NEURAL NETWORK ASSISTED SOFTWARE ENGINEERED REFRACTIVE FRINGE DIAGNOSTIC OF SPHERICAL SHOCKS

by

TREVOR KISTAN

Submitted in partial fulfilment of
the requirements for the degree of
Master of Science,
in the
Department of Computer Science,
University of Natal
1996

Durban
1996

Preface

The experimental work described in this thesis was carried out in the Department of Computer Science, University of Natal, Durban, from February 1993 to November 1996, under the supervision of Professor A. G. Sartori-Angus.

These studies represent original work by the author and have not been submitted in any form to another University. Where use was made of the work of others it has been duly acknowledged in the text.

Acknowledgements

The author wishes to thank the following people for their help, support and encouragement over the past four years.

Professor A. G. Sartori-Angus for supervising this work, for suggesting a neural network approach and for providing facilities whenever the author visited the department for research.

Professor M. M. Michaelis for access to his ray-tracer code, careful explanations, generous encouragement and, of course, for teaching him much of the underlying physics in his undergraduate days.

Mogamberi Kuppen, Nichola Lisi and Ricardo Buccellato for supplying the experimental images, for fruitful discussions and for allowing him access to the laser laboratory to get first hand experience of the experimental setup.

Doctor H. Murrell and Kevin Meyer for their genuine interest in his progress.

Alexander Polmans for introducing him to Computer Aided Software Engineering (CASE) and the Shlaer-Mellor Object Orientated Software Analysis Methodology.

Bunny Thathiah, Steven Whiting and Andrew Gillespie for helping to generate the massive training database.

Peter Olander and Trevor Tibbits for helping to proofread this document. Any remaining errors are the author's own.

Friends and colleagues for technical advice and the use of software and equipment.

His family for emotional support and encouragement.

His employers, the Naval Systems Division of Altech Defence Systems (Proprietary) Limited, in particular Mr Ian Harris and Mr Trevor Coni, for financial sponsorship, access to a CASE tool, use of the NeuralWare NeuralWorks Explorer™ software and generous amounts of study leave.

Abstract

A shock is essentially a propagating variation in the pressure or density of a medium. If the medium is transparent, such as air, and the shock radially symmetric, the refractive fringe diagnostic can be used to examine its general features. A laser beam probes the shock, the central part of the beam, refracted to different degrees by the different density features within the shock, interferes with itself and with the outer unrefracted part creating a series of coarse and fine fringes. By examining this interference pattern one can gain insight into the density profile underlying the shock.

A series of such experiments was conducted by the Plasma Physics Research Institute at the University of Natal in 1990. To model the situation computationally, they developed a ray-tracer which produced interference patterns for modified theoretical density profiles based on those predicted by Sedov. After numerous trials, an intensity pattern was produced which agreed approximately with experimental observations. Thus encouraged, the institute then sought to determine density profiles directly from the interference patterns, but a true mathematical deconvolution proved non-trivial and is still awaited.

The work presented in this thesis reconstructs the ray-tracer using software engineering techniques and achieves the desired deconvolution by training a neural network of the back-propagation type to behave as an *inverse* ray-tracer. The ray-tracer is first used to generate numerous density profile - interference pattern pairs. The neural network is trained with this *theoretical* data to provide a density profile when presented with an interference pattern. The trained network is then tested with *experimental* interference patterns extracted from captured images. The density profiles predicted by the neural network are then fed back to the ray-tracer and the resultant theoretically determined interference patterns compared to those obtained experimentally.

The shock is examined at various times after the initial explosion allowing its propagation to be tracked by its evolving density profile and interference pattern. The results obtained prove superior to those first published by the institute and confirm the neural network's promise as a research tool. Instead of lengthy trial and error sessions with the unaided ray-tracer, experimental interference patterns can be fed directly to an appropriately trained neural network to yield an initial density profile. The network, not the researcher, does the pattern association.

CONTENTS

PAGE

LIST OF FIGURES

LIST OF TABLES

PAGE

LIST OF ACRONYMS

2D/3D	Two Dimensional / Three Dimensional
AFCEA	Armed Forces Communications & Electronics Association (US)
AI	Artificial Intelligence
ARPA	Advanced Research Projects Association (formerly DARPA - Defence ARPA)
CASE	Computer Aided Software Engineering
CCTV	Closed Circuit Television
DFD	Data Flow Diagram
DETA	Diethylenetriamine
ERD	Entity Relationship Diagram
IEEE	Institute of Electrical & Electronic Engineers
LED	Light Emitting Diode
OPL	Optical Path Length
PSpec	Process Specification
RMS	Root Mean Square
STD	State Transition Diagram
US/USA	United States / United States of America
VLSI	Very Large Scale Integration

CHAPTER 1 Introduction

1.1 The Refractive Fringe Diagnostic

A shock is essentially a propagating variation in the pressure or density of a medium. If the medium is transparent, such as air, and the shock radially symmetric, the refractive fringe diagnostic can be used to examine its general features. A single laser beam probes the shock, the central part of the beam, refracted to different degrees by the different density features within the shock, interferes with itself and with the outer unrefracted part, creating a series of coarse and fine fringes. By examining this interference pattern one can gain insight into the density profile underlying the shock.

1.2 Some Perspective

A series of such experiments was performed by Michaelis, Waltham and Cunningham [Mic91] at the University of Natal's Plasma Physics Research Institute in 1990. To model the situation computationally, they developed a ray-tracer which produced interference patterns for carefully modified theoretical density profiles based on those predicted by Sedov [Sed59]. After numerous trials an intensity pattern was produced which agreed approximately with experimental observations of the number and position of the fringes, but not their intensities. Better agreement was prevented by the patchy spatial coherence of the probing beam and by a lack of mathematical sophistication in the computational ray-tracer [Mic91]. Nonetheless, their results represented the first computation of refractive fringes to bear any resemblance to experiment. Thus encouraged, the institute then sought to determine density profiles directly from the interference patterns, but a true mathematical deconvolution proved non-trivial and is still awaited [Buc94].

1.3 Neural Networks

The work presented in this thesis reconstructs the ray-tracer using software engineering techniques and achieves the desired deconvolution by training a neural network of the back-propagation type to behave as an *inverse* ray-tracer. Neural networks are computer programs that attempt to emulate the workings of biological brains at the level of neurons and synaptic connections. They exhibit some properties of biological brains such as an ability to generalise and degrade gracefully. Rather than being programmed, they are trained in much the same way that people learn. When repeatedly presented with a set of input-output pairs, the back-propagation network learns the relationship between them, eliminating the need for a formal mathematical description of the relationship. The network will behave as a pattern-associator, performing the same work and learning the same relationships that the physicists did when manipulating their ray-tracer.

1.4 Problem Approach

The ray-tracer is first used to generate numerous density profile - interference pattern pairs. The neural network is trained with this *theoretical* data to provide a density profile when presented with an interference pattern. Once trained the neural network is tested with *experimental* interference patterns extracted from images provided by the institute. The density profiles predicted by the neural network are then fed back to the ray-tracer and the resultant theoretically determined interference patterns compared to those obtained experimentally.

The shock is examined at various times after the initial explosion allowing its propagation to be tracked by its evolving density profile and interference pattern. These results, while still suffering to some extent from the problems discussed above, prove superior to those first published by the institute and confirm the neural network's promise as a research tool. Instead of lengthy trial and error sessions with the unaided ray-tracer, experimental interference patterns can be fed directly to an appropriately trained neural network to yield an initial density profile. The network, not the researcher, does the pattern association.

1.5 Applications

In later work undertaken by the Plasma Physics Research Institute the spherical shock was replicated in a synchronised, circular arrangement to produce a colliding shock lens that focuses laser beams in a similar manner to glass lenses, but without the disadvantage of cracking when the laser intensity becomes excessive [Buc93, Buc94]. Applications for such novel lenses include industrial drilling, cutting and welding, ultra-high power and all gas lasers [Mic94] and Q-switching and spatial filtering of lasers [Mic94, Lis94].

The refractive fringe diagnostic itself is useful in applications such as laser fusion research where the time scales[1] are too small to permit use of standard interferometric techniques [Mic91]. In laser fusion carefully aligned lasers target a hydrogen filled glass microballoon in the hope that the resultant *implosion* will be strong enough to cause nuclear fusion of the hydrogen nuclei into helium.

Coccorese, Martone and Morabito [Coc92, Mor93] present an interesting application of neural networks in plasma fusion research. Tokamak devices demonstrate the feasibility of controlled nuclear fusion by confining a thermonuclear plasma[2] within a magnetic field. To control the position and shape of the plasma column, the parameters defining the plasma current density profile must be estimated from magnetic measurements within milliseconds. Coccorese et al. [Coc92] tackle this difficult problem using a back-propagation neural network trained off-line on experimental data.

1.6 Thesis Outline

This thesis tackles a real-world problem and will take a multi-disciplinary approach. We shall first present sufficient theoretical physics from optics and hydrodynamics to describe the problem domain and develop the ray-tracer algorithm.

We then proceed to a discussion of neural networks and the back-propagation learning algorithm that emphasises the future directions of the field rather than its historical perspective. A comprehensive theoretical treatment of back-propagation is preceded by a biological and psychological prelude and

[1] Of the order of 50 ps.

[2] A Deuterium-Tritium mixture at 10 KeV.

followed by a review of some advanced topics, including: hybrid neural network systems with expert system, fuzzy logic and genetic algorithm components; rule extraction; alternate minimisation techniques; optical neural networks; bioelectronic networks and the biological plausibility of back-propagation.

This is followed by a review of the theoretical training data generated by the program. An outline of the experimental setup will precede the presentation of the experimental test data and related image processing considerations. The user-interface of the rewritten raytracing program will then be discussed. The architecture of the neural network used in this study will be discussed in some depth, followed by presentation and discussion of the results generated by the network - ray-tracer pair.

As the raytracing program is an application and not disposable research code, some software engineering effort has gone into its design. To aid comprehension and future maintenance, we shall present appendices containing detailed software analysis using the Shlaer-Mellor Object-Oriented Methodology [Shl88, Shl92], software analysis and code metrics, and a full source code listing.

CHAPTER 2 Theoretical Aspects of Shock Waves and Optics

2.1 The Spark Discharge in Air

Spherical shocks can be generated by discharging a spark between two closely spaced, pointed electrodes. An electrically conducting column forms in the gap between the electrodes and the temperature of the air within this column is raised to tens of thousands of degrees [Zel66] by Joule heating. The consequent increase in pressure causes the column to expand and act as a piston on the surrounding air, sending it into a cylindrical shock wave. After the shock travels a distance exceeding the column length [Zel66] it gradually becomes spherical.[3]

2.2 The Mach Number

Mach's number is the ratio of the velocity of the shock to that of sound [Pai83].

$$M = \frac{v_{shock}}{v_{sound}} \quad \textbf{(2.1)},$$

2.3 Sedov Density Curves

Michaelis, Waltham and Cunningham [Mic91] observe that little information is available on the density ahead of, and some distance behind, the shock front. The density profiles of the shock as predicted by Sedov [Sed59] and adapted by Michaelis et al. [Mic91], appear in figure (2.1) for various times after the initial explosion. These predictions make use of the Hugoniot conservation equations to relate the pressure, density and velocity of the gas ahead of and behind the shock front. The Hugoniot density equation may be expressed as [Mic91]

$$\rho_2 = \rho_1 \frac{\gamma + 1}{\gamma - 1 + 2q} \quad \textbf{(2.2)},$$

where

- ρ represents density,
- γ is the ratio of specific heats (~1.4 for air), and
- $q = 1/M^2$, where M is the Mach number defined in equation (2.1).

[3] Incidently, similar phenomena occur in storms. The shock wave produced by a lightning stroke degenerates into an acoustic wave (thunder) at large distances - a fact only appreciated in the previous century [Uma69].

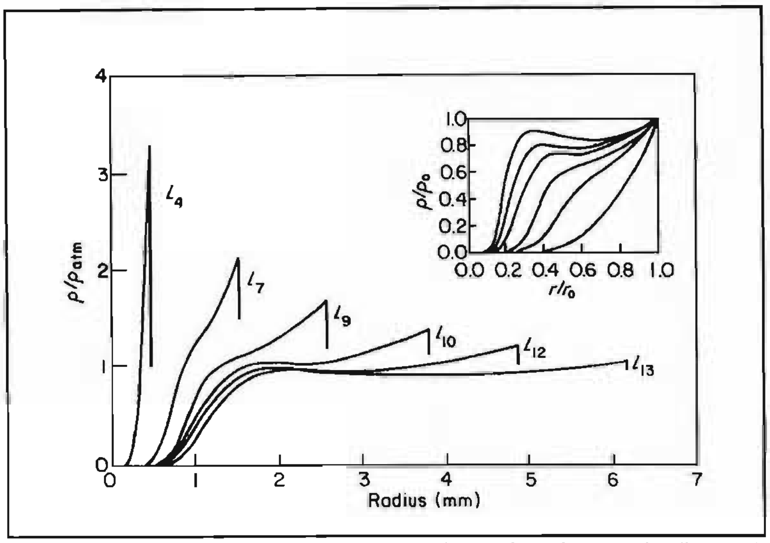


Figure 2.1 Adapted Sedov theoretical density profiles from Michaelis et al. [Mic91]. The insert shows the actual curves.

These curves can be grouped into early, middle, late and very late periods [Mic91].

Table 2.1 Temporal Classification of Adapted Sedov Density Profiles

Period	Time (μs)	Curve	Description
Early	1.0	l_4	The shock is strong with a high Mach number. The gas absorbed by the shock decompresses exponentially.
Middle	2.02 4.03	l_7 l_9	The Mach number and density drop and a small hump forms on the tail of the exponential.
Late	6.8 9.53	l_{10} l_{12}	The hump increases in density and may be preceded by a hollow curve
Very Late	13.0	l_{13}	Only the weakened shock front and the central vacuum region have densities differing from atmospheric.

Although the shock is probed at all the times listed in table (2.1), curve l_9 in the middle region forms the basis for our investigations. The work of Michaelis, Waltham and Cunningham [Mic91] shows that it is necessary to modify the gradients of the density curve (and hence from the next section, the refractive index of the shocked air) to get acceptable results from the ray-tracer.

2.4 The Refractive Index of a Shock

Ignoring the complex chemical structure of air, it is observed empirically [Mar82] that the refractive index of a gas depends on its density as follows

$$n = 1 + (n_0 - 1)\frac{\rho}{\rho_0} \qquad \textbf{(2.3),}$$

where

- ρ_0 is some constant density, and
- n_0 is the refractive index for that density.
- If we are working in units of atmospheres at sea level, then $\rho_0 = 1$.

So as a consequence of its uneven density profile a shock has a spatially varying refractive index. A ray of light traversing the shock is bent under the action of the local refractive index gradients according to Snell's law.

2.5 Snell's Law or the Law of Refraction

When a ray of light passes between optical media of differing refractive indices the ray is deflected towards the normal to the interface if the second medium has a higher refractive index, and away from the normal otherwise. In both cases the relationship (in the plane of the rays and the normal) can be expressed as [Sea87]

$$n_1 \sin\phi_1 = n_2 \sin\phi_2 \qquad \textbf{(2.4),}$$

where

- n represents the refractive index and
- ϕ is the angle between the ray and the normal.

Within each of the optical media the ray travels in a straight line. This fact, and indeed Snell's Law, are both consequences of Fermat's principle.

2.6 Fermat's Principle or the Principle of Least Time

Fermat's Principle states that the distance the ray travels - the *optical path length*, has a stationary value [Pai83]. This implies that the optical path length within a medium of constant refractive index is the shortest distance across the medium - a straight line which is also the path of minimum time. When a ray crosses an interface between media of differing refractive indices, it adjusts its path to take the minimum time to reach its end point, thus giving some physical basis to Snell's Law.

The probe beam consists of numerous rays which are refracted to different degrees within the shock. To determine what happens to the rays after they exit the shock we have to consider the wave nature of light.

2.7 Interference Patterns

2.7.1 The Principle of Linear Superposition

Interference results when waves from two or more sources overlap in space. The intensity of the light at any point is governed by the *principle of linear superposition* which states that the resultant amplitude can be obtained by adding those instantaneous amplitudes that the individual waves would have produced at the point if each were present alone [Sea87].

If waves from two sources arrive at a point in phase, their amplitudes are added in what is termed *constructive interference* or reinforcement. At the other extreme, if they arrive half a cycle out of phase, their amplitudes are "subtracted" in what is called *destructive interference* or cancellation. Thus a sequence of alternate bright and dark fringes is formed. The phase of the wave depends on the distance it travels from the source - the *optical path length* (OPL). The phase difference between two waves emanating from coherent sources is related to the difference in their optical path lengths as follows [Sea87]

$$\phi = \frac{2\pi L}{\lambda} \quad \textbf{(2.5),}$$

where

- ϕ is the phase difference,
- L the difference in optical path lengths and
- λ the wavelength.

By considering the conditions for constructive interference the bright fringe separation is given by [Mic91, Sea87, Pai83]

$$d_k = \sqrt{2k\lambda L'} \quad \textbf{(2.6),}$$

where

- d_k is the distance to the k'th fringe from the first unrefracted ray.
- k = 1,2,...
- λ is the wavelength and
- L' is the distance from the screen to the *ring focus* (see section 2.8).

This law is only approximately obeyed by the interference patterns produced by the shock [Mic91].

For an interference pattern to form it is crucial that the waves
(a) have the same phase when they leave the source,
(b) maintain a constant phase relationship, and
(c) have the same frequency and wavelength
- i.e the probe beam must be both *coherent* and *monochromatic*. These requirements are met by laser light.

2.7.2 Characteristics of Laser Light

Laser beams are intense, almost perfectly parallel, almost monochromatic and, within a given cross-section, spatially coherent [Sea87]. With reference to the previous section, these characteristics make laser light ideal for the probe beam in the refractive fringe diagnostic, but Waltham [Wal90] notes that the wavelength of the probing radiation should be selected with some care. By taking a ray optics approach in the refractive fringe diagnostic, we are assuming that the probing wavelength is much less than the scale length of the refractive index gradient within the shock. That is, after Born and Wolf [Bor65]

$$\left|\frac{\partial n}{\partial y}\right| \ll \frac{n}{\lambda_{vac}} \qquad \textbf{(2.7),}$$

where

- n is the local refractive index,
- λ is the probe wavelength in vacuum and
- y is in the direction of the density gradient normal to the ray.

2.7.3 The Intensity of Radiation

The intensity of electromagnetic radiation is the average power transmitted per unit area and is proportional to the square of the electric field of the radiation. Specifically [Sea87],

$$S = \epsilon_0 c E^2 \qquad \textbf{(2.8),}$$

where

- S is really the rate of energy flow per unit area,
- ϵ_0 is the constant, permittivity of free space,
- c is the constant, speed of light and
- E is the magnitude of the electric field.

2.8 The Refractive Fringe Diagnostic

From the previous discussion we can see that, when a shock is probed, an interference pattern forms because the refractive index gradients within the shock are steep enough to cause the incident rays to cross over when they emerge from the shock [Bha83] as indicated in figure (2.2). The point(s) at which they cross over is termed the *ring focus* (position L' of equation 2.6). If a screen is placed at this point a bright ring will be observed; if the screen is moved further away an interference pattern similar to that in figure (2.2) forms [Bha83].

We now introduce some terminology to describe the various density profile and intensity pattern regions following Michaelis, Waltham and Cunningham [Mic91]. A spherical shock has five density regions labelled 2 to 6 in figure (2.2). Moving inwards these are the shock front (2), rear (3), saddle (4), tail (5) and central vacuum (6). Region (1) is undisturbed air at atmospheric pressure ahead of the shock front.

The features of the interference pattern are labelled I to VII from the outside of the pattern in figure (2.2). We find first a set of outer fine fringes (I), followed by a dark - bright ring pair (II), then a set of inner fine fringes (III) followed by a coarse fringe (IV) and a bright - grey ring pair (V and VI) and finally undisturbed illumination in the central region (VII). It is the fringes of region (I) which obey equation (2.6) approximately [Mic91].

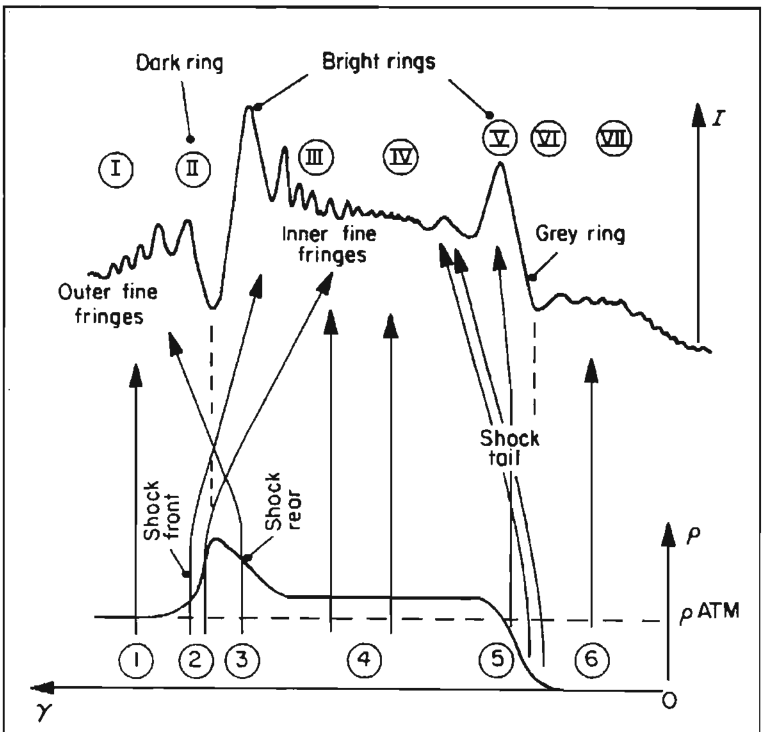


Figure 2.2 Schematic of a shock density, ray deflections & interference pattern from Michaelis et al. [Mic91].

These intensity features form as follows: (I) results from the interference of unrefracted rays from (1) and those refracted outwards from (3); (II) occurs opposite the shock peak; (III) from the interference of rays refracted inwards from (2) and unrefracted rays from (4); (IV) from rays originating on different sides of (5); (V) from rays refracted outwards by (5); (VI) is due to the fact that most of the light ends up in (V) and (VII) occurs because the rays traversing (6) are hardly deflected, as is the case with those traversing regions (1) and (4).

2.9 The Raytracing Algorithm

Given a shock density profile the ray-tracer calculates the interference pattern that would result from probing the shock with a laser beam. It does this by dividing the shock into numerous concentric density shells, each one micron thick as indicated in figure (2.3).

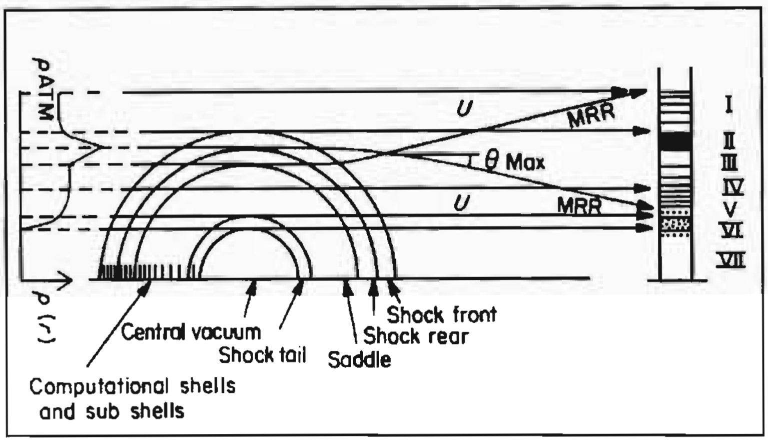


Figure 2.3 Computational subshells in the raytracer from Michaelis et al. [Mic91].

The refraction of a large number of rays, each one micron apart, is computed through each shell, allowing us to calculate the total distance or optical path length of each ray to the observation plane. The optical path length of each ray comprises three major sections: (a) the first free OPL before the ray enters the shock, (b) the OPL travelled by the refracted ray within the shock, and (c) the second free OPL from the ray's exit point to the observation plane. Each of these sections will have to be determined separately and added. The phase of each ray may be calculated from its total OPL and the interference pattern determined from the phase differences of the interfering rays.

Rays within a laser beam are virtually parallel, so incident rays will all be parallel to the x-axis. Since distance in the ray-tracer is quantised into microns, the position of each incident ray and that of each subshell is known, and we need only the refractive index in each subshell to proceed. We note from equation (2.3) that the refractive index of each subshell can be calculated from its density which is known once we select a density profile.

We pause to observe that the following simplifications are applied at various stages:

- we approximate the probe wavelength to 1λ. (A1)
- where convenient, we approximate the refractive index of air at atmospheric pressure to unity. (A2)
- we make some small angle approximations. (A3)

Finally we are in a position to present a (still somewhat simplified) calculation of the interference pattern using the physical laws and equations presented in this chapter. Consider horizontal ray OB at height Y_i incident on the first subshell which has radius R_i in figure (2.4).

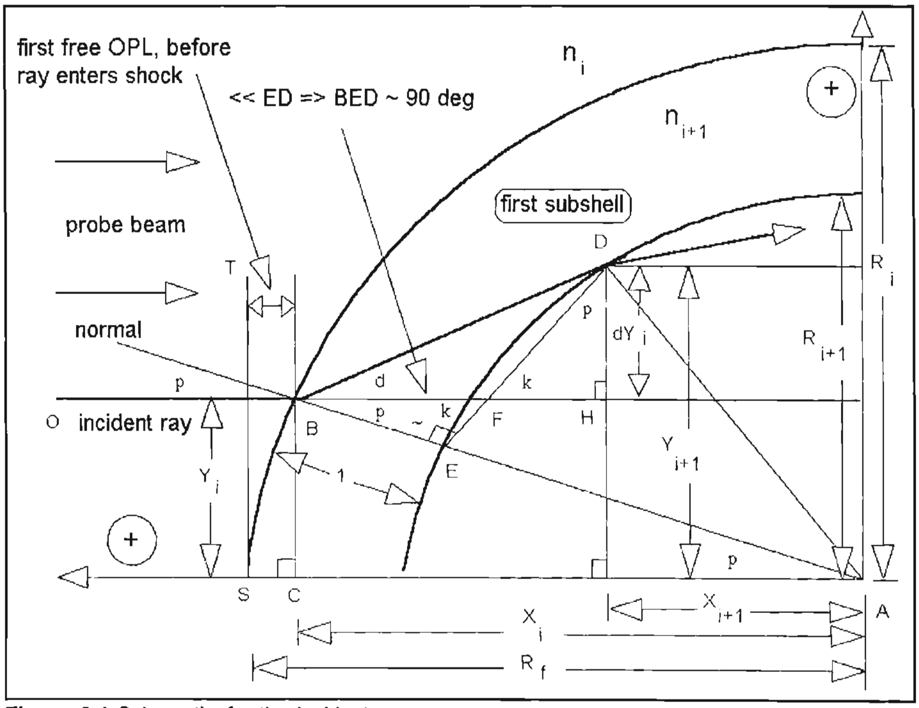


Figure 2.4 Schematic for the incident ray.

Immediately, from **Δ**ABC

$$p = \sin^{-1}\left(\frac{Y_i}{R_i}\right) \quad \textit{and} \quad X_i = \sqrt{R_i^2 - Y_i^2} \qquad \textbf{(2.9).}$$

By Snell's law, equation (2.4), we have

$$n_i \sin p = n_{i+1}\sin(p+d)$$

$$\sin(p+d) = \frac{n_i}{n_{i+1}}\sin p \qquad \textbf{(2.10).}$$

$$d = \sin^{-1}\left(\frac{n_i}{n_{i+1}}\sin p\right) - p$$

Now consider **Δ**BDE. For a small enough ED, angle BED approximates a right angle as chord ED becomes tangential. We can justify this from the scales involved and from equation (2.7). Since the subshells are 1 micron apart, BE = 1 and $R_{i+1} = R_i - 1$ and we have

$$
\begin{aligned}
tanp &= \frac{EF}{BE} = EF \\
\tan(p+d) &= \frac{ED}{BE} = ED \\
DF &= \tan(p+d) - tanp \\
\Delta Y &= DFcosp \\
&= (\tan(p+d) - tanp)cosp \\
Y_{i+1} &= Y_i + \Delta Y \\
X_{i+1} &= \sqrt{R_{i+1}^2 - Y_{i+1}^2} \\
&= \sqrt{(R_i - 1)^2 - Y_{i+1}^2} \\
OPL_{subshell} &= BD \\
&= n_{i+1}\sqrt{(X_i - X_{i+1})^2 + \Delta Y_i^2}
\end{aligned}
\qquad \textbf{(2.11),}
$$

where the factor n_{i+1} appears as within the shock we can no longer take n to be 1 as per assumption (A2). For the other subshells the equation for d becomes

$$
d_{i+1} = \sin^{-1}\left(\frac{n_i}{n_{i+1}} \sin(p_i + d_i)\right) + p_i \qquad \textbf{(2.12),}
$$

since initially d_0=0 and we adjust d as we iterate over all the subshells. Finally, summing over all the subshells we get

$$
OPL_{shock} = 2\sum_{i=1}^{N} n_{i+1}\sqrt{(X_i - X_{i+1})^2 - \Delta Y_i^2} \qquad \textbf{(2.13),}
$$

where N is the number of subshells and the factor 2 is a symmetry factor introduced because as the rays leave the shock they pass, in reverse, through the same shells that they did when they entered travelling an equal distance both ways.

The first free optical path length can also be calculated with reference to figure (2.4). For each ray this is the horizontal distance between plane TS and the shock front. For a ray at height Y_i this is simply

$$
OPL_{1st\_free} = R_f - \sqrt{R_f^2 - Y_i^2} \qquad \textbf{(2.14),}
$$

where

- R_f is the radius of the shock front and
- Y_i the height of the incident ray.

The second optical free path length may be obtained from figure (2.5).

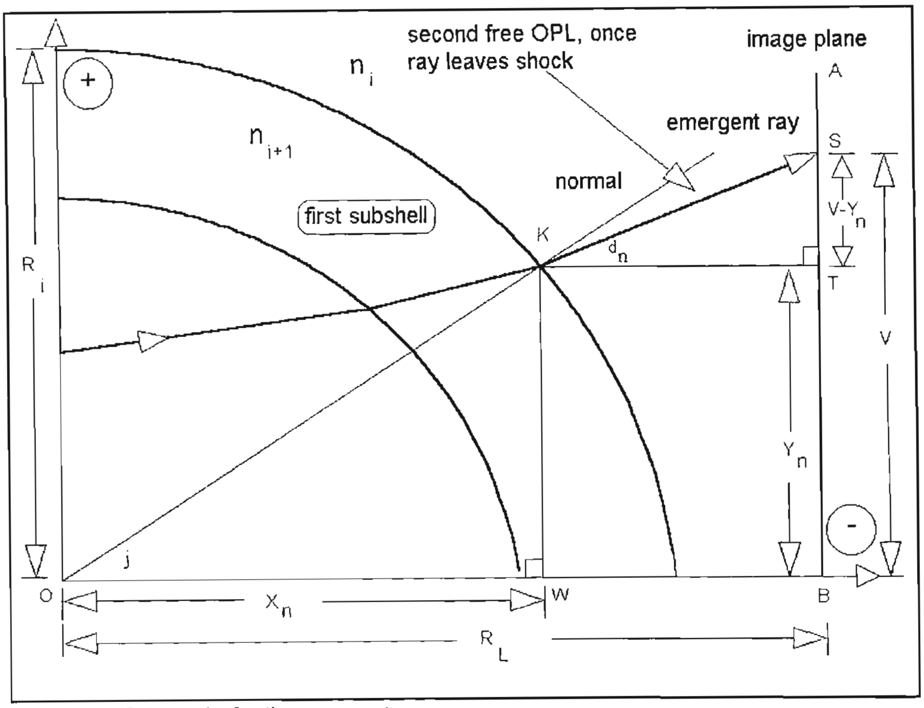


Figure 2.5 Schematic for the emergent ray.

Since the distances R_L to the image plane and R_f the shock front radius are known, we have from **Δ**KST

$$V = (R_L - R_f)\tan d_n + Y_n$$

$$OPL_{2nd\_free} = \sqrt{(V-Y_n)^2 + (R_L-R_f)^2} \qquad \textbf{(2.15).}$$

So eventually,

$$OPL_{total} = OPL_{1st\_free} + OPL_{shock} + OPL_{2nd\_free} \qquad \textbf{(2.16).}$$

Using approximation (A1) we get the phase difference ϕ by considering only the fractional part of the OPL (discarding whole wavelengths). Again using λ=1 in equation (2.5) we have

$$\phi = 2\pi\, frac(OPL_{total}) \qquad \textbf{(2.17).}$$

Using ϕ we calculate, separately, the cos and sin components of the electric field at point S in figure (2.5). We increment the cos and sin components of the electric field for each ray that lands at point S and at all other points one micron apart on the image plane. Once all the rays are traced we note from equation (2.8) that the intensity is proportional to square of the electric field and set finally for each point on the image screen

$$\begin{aligned} I &= E^2 \\ &= \left(\sqrt{E_{cos}^2 + E_{sin}^2}\right)^2 \\ &= E_{cos}^2 + E_{sin}^2 \end{aligned} \qquad \textbf{(2.18).}$$

CHAPTER 3 Neural Network Theory and the Back-propagation Algorithm

3.1 Introduction

Traditional artificial intelligence techniques have often achieved their greatest successes only by tackling problems of limited scope or artificial construction. They are generally sequential in nature and are therefore susceptible to combinatorial explosion. Everyday human activities, such as recognising familiar faces, conversing and muscular co-ordination continue to tax modern computers.

People remain smarter than machines because of the brain's massively parallel architecture. Taking typical densities of 10^{10}-10^{11} neurons, each with between 1-1000 synapses, one may deduce that no neuron is more than four synaptic connections away from the next [Rum86]. McClelland, Rumelhart and Hinton [Rum86] argue further that time constraints on the *microstructure* of human cognition exclude sequential processes in favour of parallel distributed models, many of which are stochastic. Neurons have integration times of milliseconds or tens of milliseconds, thereby excluding lengthy sequential processing [And83].

Artificial neural networks are computer programs that attempt to emulate the workings of biological brains at the level of neurons and synaptic connections. They exhibit many of the properties of real brains such as graceful degradation in the presence of noise, inherent parallelism and an ability to generalise beyond what they learnt. Rather than being programmed they learn to extract features or recognise patterns in much the same way as humans. They require no prior knowledge of the relationships inherent in the data as they extract such information empirically.

Neural networks are often applied in situations where good analytical models are unknown, complex or incomplete. Cybenko [Cyb96] describes a systems approach to neural networks. Systems are based in the real world. Models are mathematical or computational representations of systems and data are observations of those systems that are used as the basis for building models. Neural networks provide a general mechanism for building models from data.

3.2 Biological Foundations

The study of the brain uses three approaches: anatomy studies the structure of the system, neurophysiology studies its function and experimental psychology studies its behaviour [Per92]. A brief overview of these topics will provide some multi-disciplinary background and introduce terms referenced in later discussion of artificial neural network models and the biological plausibility of their learning algorithms.

Biological networks have long lifetime,[4] a massive number of components and interconnections, both spatial and temporal redundancy, error detection and learning capabilities [Moo87]. Hopfield [Hop88] observes that much of what passes for intelligence in humans is actually a massive associative memory at work. Dayhoff [Day93] suggests that neural network architectures may be improved by borrowing further from biology and by considering temporal structures in particular [Lin93]. Southwell et al.

[4] More than 70 years of continuous use.

[Sou93] attempt to capture some of the features of the mammalian cortex by considering functional columns in cortical layers and their inter-column links. Vanderbeek, Butler, El-Dinary and Strohbehn [Van93] have attempted to emulate a biological model of a cockroach leg but have had to deviate from the model to get results from their network.

3.2.1 Anatomy

Peretto [Per92] argues that because neurons are so numerous[5] and similar, we can make predictions based on ensemble averages using the concepts of statistical physics. He argues that the complexity of the human nervous system is such that the human genome is unable to control the lower levels of its hierarchy (the neurons and synaptic junctions). At higher levels however, genetic determination does step in.

3.2.1.1 General Structure

The cortex is a flat structure on the surface of the brain that serves as a "central" processor. The thalamus acts as a preprocessor, the hypothalamus controls hormonal secretions, the nigrostriate formation coordinates long lasting actions, the cerebellum sequences coordinated motion and the hippocampus is involved in long term memory.

3.2.1.2 Functional Structure of the Cortex

The cortex comprises two sheets (2 mm thick, spread over 11 dm^2) connected by an 800 million fibre bus known as the corpus callosum. There are about 40 cortical areas (defined both functionally and anatomically) in each hemisphere of the brain. The cortex has three concentric levels of organisation (parietal, receptor, temporal) and four radial systems of connection (auditory, language, visual, motion) as well as connections to limbic areas and the olfactory bulb. These inter-area connections are determined genetically. Genes promote specific tracer molecules which trigger the formation of neural growth factor proteins which control the growth of neuronal fibres during epigenesis.

3.2.1.3 Cortical Maps, Columns and Microcolumns

Signals from the sensory organs map onto specific areas of the cortex forming cortical maps composed of regions known as cortical bands. These bands are feature detectors. The cochlea in the inner ear, for example, is a frequency analyser that maps onto cortical bands sensitive to certain frequencies. Humans have large cortical areas devoted to receiving sensory input from the fingers and smaller areas for receiving those from toes, while the representations are more equal in the rhesus monkey [And83]. However, the precise structure of cortical maps is not determined genetically. The cortical bands devoted to a particular limb will disappear if that limb is amputated [Per92].

Most of the neurons in the cortex are the pyramidal neurons whose dendrites are elongated. Taken together, these extensions form a column. The axons of the neurons in a column only connect to neurons of other specific columns, sometimes in other cortical areas.

5 3×10^{10} neurons, 10^{14} synapses.

Microcolumns are sets of about 100 [Per92] strongly connected neurons which develop during epigenesis. A particular neuron in a microcolumn cannot be excited or inhibited without affecting the activities of all the other microcolumn neurons. This suggests that the microcolumn may be a basic functional unit of the cortex [Per92].

A juxtaposition of microcolumns up the size of a column is known as a band. Table (3.1) summarises the components of the human cortex.

Table 3.1 Elements of the Human Cortex (after [Per92])	
Neurons	3×10^{10}
Microcolumns	3×10^{8}
Cortical Bands	5×10^{6}
Columns	3×10^{5}
Cortical Maps	1×10^{3}
Cortical Areas	5×10^{1}

The pyramidal neurons account for over 80% [Per92] of all cortical neurons, the rest are known as interneurons. The lower pyramidal neurons are output neurons whose axons contact the thalamus. The upper pyramidal neurons connect with distant pyramidal cells in other columns. Both families of pyramidal neurons receive their input from interneurons called stellate cells. Contact between pyramidal cells within a column is excitatory and is mediated by the neurotransmitter glutamate:

```
              H    H    H
              |    |    |
OH —  C — C — C — C — C — OH .
      ||   |    |    |    ||
      O    H    H    N    O
                    / \
                   H   H
```

The interneurons provide indirect inhibitory interactions between the pyramidal neurons using the neurotransmitter gamma-aminobutiric acid:

```
              H    H    H        H
              |    |    |       /
OH —  C — C — C — C — N            .
      ||   |    |    |        \
      O    H    H    H         H
```

3.2.2 Neurophysiology

Imbalances in ionic (Na^+, K^+, Cl^-) concentration on either side of the neuronal cell membrane create an electric potential of ~70 mV. This resting potential is maintained by ionic pumps floating in the membrane and by ionic channels. Sodium ions entering the cell depolarise the membrane to ~+40 mV. After about 0.5 ms potassium ions flow out of the membrane, repolarising it to ~-80 mV. The peak of the membrane potential, about 1 ms long, is called the action potential. This is followed by a 3 to 4 ms refractory period during which the ionic channels close and the ionic pumps reset the ionic concentrations. The initial excitation occurs at the hillock zone at the root of the axon (see figure 3.1.a). When the membrane potential here exceeds a threshold of ~-30 mV, the ionic channels open and the action potential propagates down the axon as a self-regenerating wave. Because it does not change its shape this is a logical or binary signal. In contrast, diffusive potential flows in dendrites are damped and therefore carry analog signals. There is the contention that neurons are analog devices because their output is a continuous valued firing frequency [And83], but the presence or absence of an action potential does encode binary information. The propagation velocity along nerve fibres is increased to ~100 ms^{-1} by the surrounding insulating myelin sheet. This sheet is interrupted at intervals by the nodes of Ranvier and it is at these that the axon actually fires.

The branches of the axons end in synaptic buttons with vesicles which contain the neurotransmitter molecules. When the action potential reaches the synaptic buttons it precipitates fusion of the vesicle and button membranes, thus permitting the neurotransmitter molecules to flow into the synaptic cleft. This gap is only about 500 Angstrom wide [Per92] and the molecules diffuse until they attach themselves to the neuroreceptors which are large proteins situated on the membrane on the postsynaptic neuron. The neuroreceptors undergo a geometric transformation which enables ionic flow within the postsynaptic membrane. Na^+ ions indicate an excitatory synapse, Cl^- ions an inhibitory one. The postsynaptic potential diffuses along the dendrite membrane until it reaches the hillock zone, where it becomes a smooth wave. The wave's amplitude reflects the influence of the action potential of the presynapic neuron on the hillock zone of the postsynaptic neuron. The amplitude term of the wave function is known as the synaptic efficacy of the synapse connecting the two neurons. Where two neurons are connected by many synapses there is still a single effective synaptic efficacy. Peretto [Per92] observes that different links can cause the effective synapse between pyramidal neurons to be of either sign. Somatic integration sums all postsynaptic potentials algebraically at the hillock zone for comparison to a threshold. The permeability of the neuronal membrane depends on the cell's history [Per92]. This amounts to modification of the neuron's threshold. Synaptic transmission is a stochastic process - an action potential has only a 0.6 probability of triggering the opening of a vesicle [Per92]. This is the main source of noise in biological networks.

In addition to the synaptic gaps discussed above, there exist gap junctions where the membranes of presynaptic and postsynaptic neurons are in physical contact. As no chemical transmitters are involved in these electrical synapses they are very fast, but are also non-plastic (i.e. they cannot participate in any learning processes).

Peretto [Per92] notes that several mechanisms of biological neurons are retained in the construction of artificial network models. These include a standard binary transmission signal (the action potential), a refractory period which provides a standard of time, the existence of both excitatory and inhibitory synapses, local somatic integration, the threshold mechanism and the probabilistic nature of synaptic transmission.

3.2.3 Experimental Psychology

Biological networks must adapt themselves to environmental constraints by modifying their parameters, particularly the synaptic efficacies [Per92]. This implies the existence of learning dynamics in addition to the neuronal dynamics discussed in section (3.2.2). Classical conditioning experiments led to the development of learning rules such the Hebbian rule and that of Widrow and Hoff [McC88]. Operand conditioning experiments reveal the following animal learning behaviour [Per92]:

- Temporalism: efficient behaviour must be postively reinforced.
- Extinction: efficient behaviour is forgotten if not reinforced.
- Generalisation: efficient behaviour in a particular environment is adapted to similar environments.
- Discrimination: distinction between similar environments is possible if the response to one is positively reinforced and that to the other negatively reinforced.

Two forms of human memory exist - a short term memory which lasts a few seconds and a long term memory which needs a consolidation period of a few seconds to be efficient. Memorised patterns can take up to several days to organise themselves [Per92], implying global learning mechanisms.

Neurons in the cortex and hippocampus sometimes fire a regular sequence of spikes at maximum frequency. This is known as bursting and blocks synaptic modification - it may therefore control memorisation [Per92]. Attention limits brain activity to a specific region of the cortex. No memorisation is possible in diffuse waking, which is an unattentive state. Peretto notes that a dream state may be necessary to organise patterns memorised in the waking state [Per92]. The larger the memorisation task in the waking state, the longer is the dream sleep stage. During slow sleep, alpha waves block the memorisation process in the same way as bursts.

3.3 The Artificial Neuron

A generalised biological neuron, as depicted in figure (3.1.a), receives signals from neighbouring neurons via numerous dendrites at synaptic junctions, integrates these signals within the cell body and gives rise to nerve impulses which it transmits to other neurons through a single axon. The strength of an input signal at a dendrite depends on the amount of neurotransmitters that cross the synaptic junction from the axon. Learning is achieved by modifying the number or state of neuroreceptors on the receiving dendrite.

A generalised artificial neuron, as depicted in figure (3.1.b), typically multiplies each of its inputs by an associated *modifiable* weighting factor, sums the products and passes the sum through a non-linear transfer function to produce a single result which may in turn serve as input to other neurons. A neural network consists of interconnected layers of such neurons passing data through weighted connections. Learning is achieved by modifying the weights of the connections.

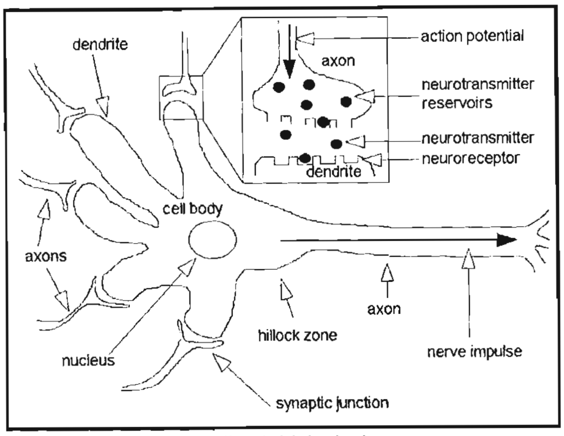


Figure 3.1.a A generalised biological neuron.

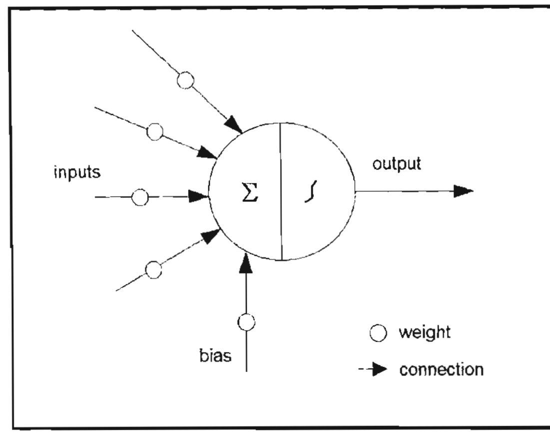


Figure 3.1.b A generalised artificial neuron.

3.4 Back-propagation Overview

The back-propagation algorithm is one specific way in which a network can be trained to learn. It has been independently presented by various authors - Bryson and Ho in 1969, Werbos in 1974, Parker in 1985 and Rumelhart et al. in 1986 [Rum86, Her91, Day90].

A back-propagation network has an input layer, an output layer and at least one hidden layer, all of which are fully interconnected. The input layer is passive and performs no computation. Neurons in the hidden layer detect feature [Hay94] or learn complex relationships (it is their introduction that first allowed networks to learn any Boolean function [Her91, Day90]). Both the hidden and output layer neurons are active and process their inputs as indicated in section (3.3). The biological analogy of section (3.3) should not be carried too far - no back-propagation mechanism has yet been identified in biological systems [Day90] and it is considered far fetched as a realistic learning mechanism [Her91]. We shall return to this discussion at the end of this chapter.

During training, an input pattern is presented to the input layer and passed to the first hidden layer over the first set of connections, each of which has a different weight. The hidden neurons sum their inputs and generate new values which they pass over the next set of weighted connections to the next hidden layer or, eventually, to the output layer. At the output layer the weighted inputs are again summed and the output compared to the corresponding target pattern to yield the error at the output layer. The derivatives of the output errors are then passed back to the hidden layer over the original weighted connections allowing each hidden neuron to calculate its contribution to the output error. This backwards propagation of errors gives the algorithm its name. Once each hidden and output neuron has calculated its error, it adjusts its weights so as to reduce the total error. After numerous presentations, the weights are altered sufficiently for the total error to reach acceptable levels. This ability to change the weight-set in response to errors is the key to the back-propagation algorithm and we shall consider it in more detail.

3.5 Adjusting the Weights using Gradient Descent

Consider a back-propagation network with one hidden layer labelled as in figure (3.2).

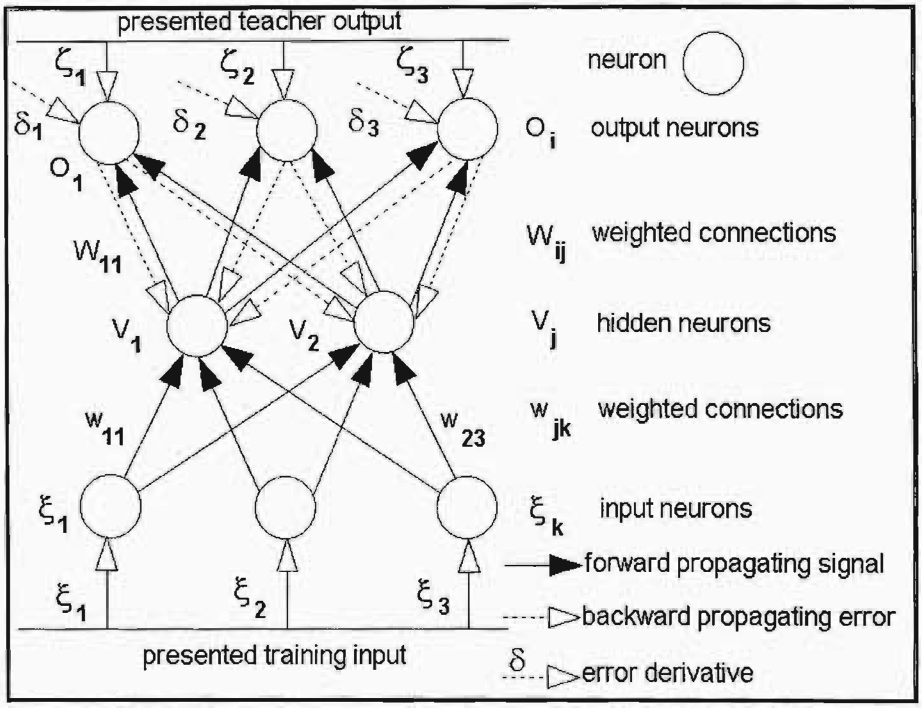


Figure 3.2 Notation for the back-propagation algorithm (after Hertz et al. [Her91]).

Following the notation of Hertz et al. [Her91] we note first that w_{ij} or W_{ij} identifies the weight connecting neuron j to neuron i and that μ refers to the μ'th training pattern. Then, from the discussion of section (3.3) hidden neuron j receives net input

$$h_j^\mu = \sum_k w_{jk}\xi_k^\mu \tag{3.1}$$

where

- ξ is the input value and
- w_{jk} is the weight of the connection from unit k to unit j,

and produces output

$$\begin{aligned} V_j^\mu &= g(h_j^\mu) \\ &= g(\sum_k w_{jk}\xi_k^\mu) \end{aligned} \tag{3.2}$$

where

- g is a continuous, differentiable *activation function* which we discuss in section (3.8).

Output neuron i then sums the previous output

$$\begin{aligned} h_i^\mu &= \sum_j W_{ij} V_j^\mu \\ &= \sum_j W_{ij} g(\sum_k w_{jk} \xi_k^\mu) \end{aligned} \qquad \textbf{(3.3)}$$

and produces as the final output

$$\begin{aligned} O_i^\mu &= g(h_i^\mu) \\ &= g(\sum_j W_{ij} V_j^\mu) \\ &= g(\sum_j W_{ij} g(\sum_k w_{jk} \xi_k^\mu)) \end{aligned} \qquad \textbf{(3.4).}$$

The usual error or cost indicator is

$$\begin{aligned} E\,[\bar{w}] &= \frac{1}{2} \sum_{\mu i} [\zeta_i^\mu - O_i^\mu]^2 \\ &= \frac{1}{2} \sum_{\mu i} [\zeta_i^\mu - g(\sum_j W_{ij} g(\sum_k w_{jk} \xi_k^\mu))]^2 \end{aligned} \qquad \textbf{(3.5)}$$

where

- the constant factor ½ is introduced for convenience in calculating the derivative later.

This function is both continuous and differentiable and may be minimised using a *gradient descent* technique. Defining

$$\delta_i^\mu = g'(h_i^\mu)[\zeta_i^\mu - O_i^\mu] \qquad \textbf{(3.6)}$$

we have for connections joining the hidden and output layers

$$\begin{aligned} \Delta W_{ij} &= -\eta \frac{\partial E}{\partial W_{ij}} \\ &= \eta \sum_\mu [\zeta_i^\mu - O_i^\mu] g'(h_i^\mu) V_j^\mu \\ &= \eta \sum_\mu \delta_i^\mu V_j^\mu \end{aligned} \qquad \textbf{(3.7)}$$

where

- η is the *learning rate* discussed in section (3.10),

and (by application of the chain rule) for those connections joining the input and hidden layers

$$\begin{aligned}\Delta w_{jk} &= -\eta \frac{\partial E}{\partial w_{jk}} \\ &= -\eta \sum_{\mu} \frac{\partial E}{\partial V_j^{\mu}} \frac{\partial V_j^{\mu}}{\partial w_{jk}} \\ &= \eta \sum_{\mu i} [\zeta_i^{\mu} - O_i^{\mu}] g'(h_i^{\mu}) W_{ij} g'(h_j^{\mu}) \xi_k^{\mu} \\ &= \eta \sum_{\mu i} \delta_i^{\mu} W_{ij} g'(h_j^{\mu}) \xi_k^{\mu} \\ &= \eta \sum_{\mu} \delta_j^{\mu} \xi_k^{\mu}\end{aligned} \qquad \textbf{(3.8)}$$

where $\boldsymbol{\delta}_j^{u}$ is now

$$\delta_j^{\mu} = g'(h_j^{\mu}) \sum_i W_{ij} \delta_i^{\mu} \qquad \textbf{(3.9).}$$

It can be shown [Her91], that for an arbitrary size network, all layers have an update rule of the form

$$\Delta w_{pq} = \eta \sum_{\mu} \delta_{output} \; x \; V_{input} \qquad \textbf{(3.10).}$$

Equation (3.9) illustrates that the $\boldsymbol{\delta}$'s for a neuron in any layer may be determined from the $\boldsymbol{\delta}$'s of neurons in the layer above it. Using the cost function at the output layer, we are able to propagate the errors ($\boldsymbol{\delta}$'s) backwards over the same connections that feed signals forward. Although it is not stated explicitly in the above discussion, neurons in a layer may receive input from neurons in any prior layer, not just the preceding one [Hay94].

3.6 Batch and Pattern Update Modes and Local Minima

An *epoch* is the complete presentation of the entire training set to the network. Network training in batch mode updates the weights after each epoch. In pattern mode the network is updated after presentation of each pattern $\boldsymbol{\mu}$. In both modes the order of presentation is randomised to prevent the network from learning any trends in the data presentation. The batch mode yields a more accurate estimate of the gradient vector and has a stronger theoretical foundation [Wei91]. The weight change for the pattern mode is only an approximation of that calculated for the batch mode [Hay94], but the pattern mode is more commonly used as it may yield better results [Wei91]. By performing a stochastic search of the weight space the pattern mode has less chance of encountering *local minima* as it performs a wider exploration of the weight space surface [Ham93]. For a geometric interpretation of gradient descent in this context consider the idealised error surface of figure (3.3).

The bottom of the bowl represents the set of weights with the smallest *root mean square* (RMS) error. The root mean square is calculated by first squaring the differences of the target pattern and output neuron values to remove negative values, then averaging over the output layer, and finally taking the square root to restore the correct magnitude. Gradient descent tries to reach the smallest RMS value by first calculating the instantaneous slope of the error surface with respect to the current set of weights and then changing the weights incrementally in the direction of the locally steepest path towards the bottom of the bowl [Ham93]. Real error surfaces have complex topologies, so by following the *locally* steepest path gradient descent can get stuck in a *local minimum* before it reaches the optimal solution.

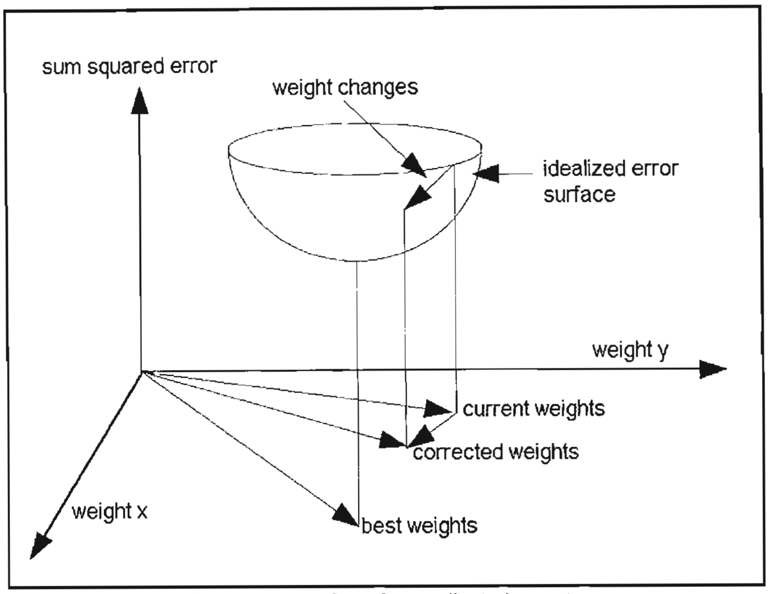


Figure 3.3 An idealized error surface for gradient descent.

In practice this does not prove to be too severe a limitation [Hay94], there is empirical evidence that such local minima occur infrequently and are not a significant problem [Wei91]. There are techniques to combat the effect and often a local minima is good enough to produce the desired results. Shallow local minima often arise when different errors compensate each other and a random presentation of the training patterns may generate sufficient noise to escape [Her91]. Alternatively noise may be explicitly added to the input when we suspect that the network is trapped [Neu91]. Local minima are more frequent when the training set is small in relation to the network size and when unfortunate choices of initial weights are made [Gor90].

3.7 The Initial Weight Space

The initial weights for all the connections are selected randomly. The wrong choice of these initial weights can lead to premature saturation where the network gets stuck at saddle points [Hay94] or local minima [Kem94a] in the error surface. If the net internal activity of an output neuron has a large magnitude, the corresponding slope of the activation function (see section 3.8) is small and the output value of the neuron is close to -1 or +1. The neuron is then in *saturation.* Initially during training both saturated and unsaturated neurons may exist in the output layer. This results in an initial rapid decrease in the RMS error followed by premature saturation [Hay94].

Some researchers put effort into sampling and preliminary adjustment of this weight-set before proceeding [Loo96]. Usually the initial network weights are uniformly distributed inside a small range. One suggestion [Loo96] for the extent of this range is

$$(-\frac{2.4}{F_i}, +\frac{2.4}{F_i}) \tag{3.11}$$

where

- F_i is the fan-in of neuron i (the number of neurons feeding into it).

Hertz, Krogh and Palmer [Her91] suggest constraining the magnitudes of the random weights to

$$w_{ij} = \frac{1}{\sqrt{F_i}} \tag{3.12}$$

where

- F_i is the *fan-in* of neuron i (the number of j neurons feeding into it).

This strategy ensures that the magnitude of typical inputs is slightly less than unity.

3.8 The Activation Function

The activation function g(h) smoothly constrains the weighted sums within a fixed range and normally saturates at its extremes. It adds a non-linearity to the network which assists in learning complex functions [Ham93]. The function has to be continuous and differentiable with g'(h) readily expressible in terms of g(h) itself. The sigmoid or logistic function is the usual choice and, as indicated in figures (3.4.a) and (3.4.b), may range over either [0,1] or [-1,1]. Sigmoidal activation functions are those that have different finite limits at $-\infty$ and $+\infty$. The logistic function has been widely used in statistics (in logistic regression for example) [Wei91].

For a range of [0,1] we have

$$g(h) = \frac{1}{1 + e^{-2\beta h}} \tag{3.13}$$

and

$$g'(h) = 2\beta g(1 - g) \tag{3.14}$$

while a range of [-1,1] requires

$$g(h) = \tanh(\beta h) \tag{3.15}$$

with

$$g'(h) = \beta(1 - g^2) \tag{3.16}$$

where

- tanh is the hyperbolic tangent, and
- ***β*** is a steepness factor that is often set to 1 or ½.

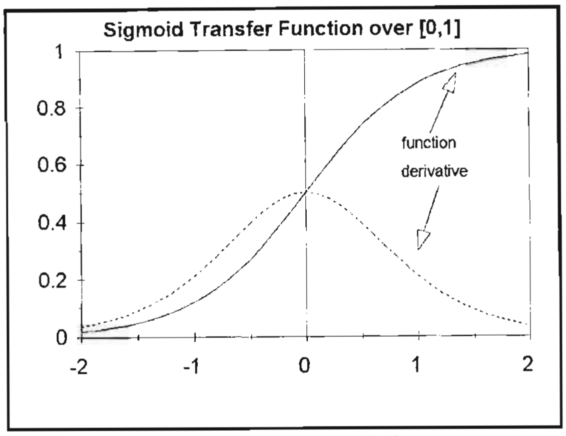


Figure 3.4.a Sigmoid function over [0,1].

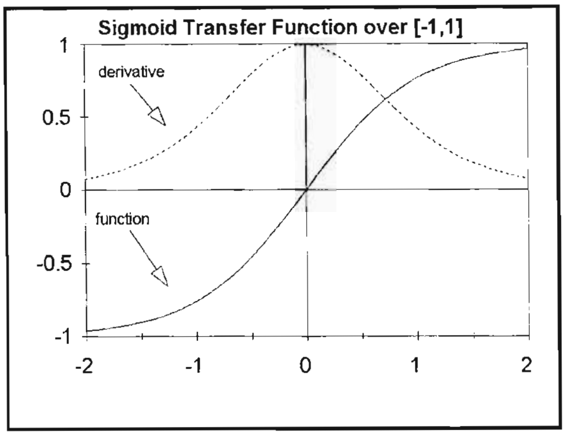


Figure 3.4.b Sigmoid function over [-1,1].

The choice of range affects the speed of training as the [-1,1] range has a larger absolute range and produces a steeper error surface directly affecting gradient descent [Kem94b]. Data presented to the network is usually *normalised* to lie within the output range of the activation function. Selection of the [-1,1] range also has implications for the bias term.

3.9 The Bias Term and the Zero Vector Problem

In the derivation of section (3.5) no explicit reference is made to the bias term as it may be considered to be just another input, clamped to a fixed value and connected to all neurons in the network. The bias term is included by default in most networks as it guarantees that neurons will receive some input during training. If we present a zero vector as input to a network whose activation range is [-1,1] and still desire a non-zero output then a bias term is necessary to stimulate the hidden and output neurons [Kem94b]. Note that a non-zero bias is equivalent to shifting the activation function left or right according to the sign of the bias [Day90].

3.10 The Learning Parameter

The factor η introduced in equation (3.7) is known as the learning rate and determines how fast the network trains. Usually η lies between 0.25 and 0.75 [Day90] - large values lead to instability and low values to excessively slow learning. The optimal value for η may not remain the same during the course of training so η is often changed, either at fixed intervals [Neu91] or according to some automatic scheme that considers the error function change [Her91]. η is often decreased during learning to prevent oscillations as we approach convergence. An alternative to an adaptive scheme is to have a different η_{ij} for each connection [Her91] where

$$\eta_{ij} \propto \frac{1}{(\textit{fan-in of neuron } i)} \qquad \textbf{(3.17).}$$

There are other advantages to making the learning rate connection dependent. Individual weights in the network can then have their values fixed by setting η_{ij} to zero for that weight, thus excluding it from further learning [Hay94]. Later layers in the network should have a smaller value for η than previous layers [Hay94]. The effective learning rate can also be affected by a momentum term.

3.11 The Momentum Term

We can increase the speed of convergence by using a momentum term. By adding to each weight change a fraction of the previous weight change we keep the changes going in the same direction, preventing oscillations and increasing the effective learning rate.

$$\Delta w_{pq}(t+1) = -\eta \frac{\partial E}{\partial w_{pq}} + \alpha \Delta w_{pq}(t) \qquad \textbf{(3.18)}$$

where

- α lies in the real range [0, 1) and usually has value 0.9 [Her91, Wei91, Hay94].

Equation (3.18) converges to

$$\Delta w_{pq} \approx -\frac{\eta}{1-\alpha} \frac{\partial E}{\partial w_{pq}} \qquad \textbf{(3.19)}$$

where

- $\eta/(1-\alpha)$ is the effective learning rate.

Note that a very large momentum term may speed training right past the global minima and out into a 'flat weight space' [Kem94a].

3.12 Tuning the Network Parameters

Typical values for the learning rate and momentum are 0.5 and 0.9 respectively [Wei91], but these may vary for the different layers and may be modified during training. Tuning the learning rate and momentum parameters in back-propagation may be problematic. Janakiraman and Honavar [Jan93] dynamically change the learning rate using heuristics that do not involve estimation of higher order derivatives and which makes the initial choice of the parameter irrelevant.

Kamiyama et al. [Kam93] present a formula to determine the parameters up front. They propose that

$$\eta = K(1-\alpha) \qquad \textbf{(3.20)}$$

where

- η is the learning rate,
- α is the momentum term,
- and K is given by

$$K = 8 \times \frac{\textit{output units}}{\textit{hidden units}} - (0.5+1.5) \qquad \textbf{(3.21).}$$

These authors suggest that the ratio of output to hidden units should lie between 0.3 and 1.0 and that α should be selected to lie within 0.3 and 0.7, whence η can be determined.

Other approaches use expert systems (section 3.23) and fuzzy logic controllers (section 3.24) to modify the parameters during training.

3.13 Network Training and Error Convergence

Network training is often an iterative, trial-and-error process. A particular configuration of network parameters is evaluated and modified until we are satisfied with the network's performance. The RMS error discussed in section (3.6) is a convenient way to measure the network's performance. As the network trains the RMS error should decrease more or less steadily. The general trend of this convergence is illustrated in figure (3.5).

Usually we stop training when the RMS error of the training-set drops below a certain level. However what we really wish to minimise is the RMS error of an independent *test-set*. Figure (3.6) shows that the RMS error of the test-set also drops at first, but then starts to rise as the network begins to memorise the data. This is known as *overtraining* and impairs the network's ability to generalise beyond its training set. The best point at which to end training is at the lowest residual test-set error. Some authors [Ham93, Hay94, Loo96] recommend alternating training and testing to avoid missing the point of lowest residual test-set error. Training and cross validation testing should be interleaved until the error of the cross validation set starts to increase [Loo96]. Learning stops when the generalisation, or testing, performance is adequate or has peaked [Hay94].

This approach assumes that an independent test-set containing both input and output (or teacher) patterns is available. During testing the test input pattern is forward-propagated to the output layer and compared to the test output pattern to determine the error - however the errors are *not* back-propagated and the network weights are *not* updated. A portion of the available training data is often reserved for testing the network, but test cases of real interest have no output or teacher patterns to match their input data. In this case the network's performance must be judged in the context of the domain in which it is applied as outputs are generated without any indication of their error.

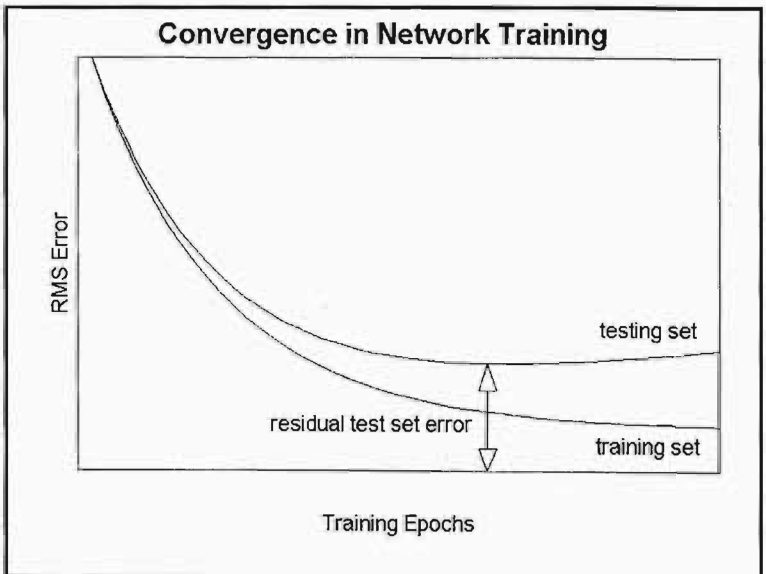


Figure 3.5 Convergence of the RMS error during network training and testing.

3.14 Some Causes of Overtraining

With too many hidden units, back-propagation approaches a direct look up table with the danger of overfitting to the training data. For two networks with the same RMS error, the simpler one will generalise better [Wei91]. As the number of hidden units increases, the training error rate generally decreases. However, as indicated in the previous section, the error rate of the test cases should decrease, flatten and then increase as the network moves from underfitting to overfitting the data [Wei91]. Such "memorisation" degrades generalisation. Looney [Loo96] suggests that overtraining is more likely to occur when the number of weights exceed the number of training sets. A worst case formula to ensure generalisation in a single hidden layer, binary output network is [Hay94]:

$$N \geq \frac{32W}{\epsilon} \cdot \ln\left(\frac{32M}{\epsilon}\right) \qquad \textbf{(3.22)}$$

where

- N is the number of training examples,
- W is the total number of weights in the network,
- M is the total number of hidden units and
- ϵ is the fraction of errors made in the training set.

Anderson [And83] provides interesting insight into the overtraining problem by quoting a study [Lur68] in which a subject with exceptional memory has trouble recognising people.

> *S. often complained that he had a poor memory for faces: "They're so changeable," he had said. "A person's expression depends on his mood and on the circumstances under which you happen to meet him. People's faces are constantly changing; it's the different shades of expression that confuse me and make it so hard to remember faces.".*
>
> *- The Mind of a Mnemonist, Luria A.R.*

3.15 Network Pruning via Weight Decay and Elimination

Reducing the complexity of neural networks prevents overtraining and loss of generalisation [Dec93, Yin93] and speeds up convergence. This can be accomplished by using any of three main methods.

- Stopping the training when the performance of the network over a separate test-set begins to deteriorate.
- Introducing penalty terms in the cost function of the network to penalise network complexity.
- Pruning irrelevant weights and neurons during training using second derivative methods or statistical analysis.

The first method is the most common. Deco, Finnoff and Zimmerman [Dec93] use the second method and introduce an unsupervised penalty term related to the *entropy* of the outputs from the hidden layer. Their term expresses the *mutual information* transferred from the input to hidden layers. A complexity penalty term identifies unreliable weights and eliminates them by setting their value to zero [Hay94]. Using the third method, Ying, Surkan and Guan [Yin93] present a weight decay mechanism that allows the weights to decay to zero in such a way that the less useful weights are preferentially removed. If all the weights to a neuron are zero, that neuron is also removed. Weights closer to the input layer are removed first as the network is less sensitive to these. When small weights are deleted, half their value is added to the threshold function of the neuron it connects to as compensation. The authors observe that the simpler network structures produced by the pruning process enhance generalisation. Looney [Loo96] and Haykin [Hay94] also observe that pruning by weight decay and weight elimination reduces network size and noise and improves generalisation. Thodberg [Tho94] uses Bayesian modelling to construct a network that automatically adjusts the weight decay parameters during training. This prunes the network and adjusts its architecture to particular problems. If the weights reduced by weight decay were not altered they could assume arbitrary values and result in poor generalisation [Hay94]. An alternative to network pruning is network growth. Here the network is grown up from a few units until satisfactory performance is achieved [Hay94].

3.16 Robustness and Convergence Considerations

Back-propagation is NP-complete [Loo96], so how do we still get results? While it cannot be proved that back-propagation converges, a theorem by Kolmogorov shows that a three layer perceptron with N(2N+1) nodes using continuously increasing non-linearities can compute any continuous function of N variables [Lip87]. According to Haykin [Hay94] the stability of back-propagation is due to the fact that, for a sigmoidal activation function, the synaptic weights which change the most are those for which the neuron's function signal is in its midrange. Hassibi, Sayed and Kailath [Has94] also provide some theoretical justification for back-propagation's robustness by proving that it is locally H^∞ optimal. This is a control theory criterion introduced to ensure robust performance when faced with model uncertainties and a lack of statistical data.

Convergence slows if the error surface is either very flat or highly curved. In the latter case the minimum may be overshot. The negative gradient vector may point away from the minimum and cause the algorithm to move in the wrong direction [Hay94]. Technically, convergence rates are linear because the Jacobian (the matrix of first order weight derivatives) is almost rank-deficient, as is the Hessian (the matrix of second order weight derivatives) [Hay94].

3.17 Multiple Hidden Layers

Multiple hidden layers may be most helpful when the number of output classes are large [Loo96]. Hidden neurons in the first hidden layer partition the feature space into convex regions which the output layer then joins up. It is not clear what role is performed by a second hidden layer [Loo96], but generalisation is better when succeeding layers of neurons are smaller than preceding ones [Loo96].

Weiss [Wei91] states that for an equal number of hidden units, two hidden layers train faster than one, but Looney [Loo96] states that there is little statistical difference between optimally trained networks with one or two hidden layers. He suggests however that networks with one hidden layer are better at generalisation while those with two are more difficult to train and are more affected by the initial weight-set and network architecture.

3.18 Some Extensions to Back-propagation

Back-propagation may be unsupervised. Buturovic' et al. [But93] present a straightforward application of back-propagation in which the target outputs are constructed during learning. The network is initially trained off-line using existing training data but then moves on to an unsupervised phase where it constructs new target outputs. Although some of the latter will be incorrect, the authors contend that if this number is small, the network will still converge to an optimal solution.

Nitta [Nit93a] has extended back-propagation to three dimensions by using three dimensional, real valued vectors for the threshold, input and output values and three dimensional orthogonal matrices for the weights. The same author has also presented a complex-numbered version of back-propagation in which the weights, threshold, input and output values are all complex [Nit93b]. Back-propagation may also be extended to non-feedforward networks [Alm89].

3.19 Rule Extraction from Neural Networks

One of the frustrations of neural networks is that they cannot explain how they arrive at their conclusions. How do humans extract rules from their neural networks? Guazzo, Rampone and Tagliaferri [Gua90] present a "dream state" hypothesis (see section 3.2.3) . They have constructed a network with three states. In the waking state the network accepts and memorises input. In the synchronised sleeping state the network rejects input, disconnects its output, and cycles to put the information into a standard form. In the desynchronised sleeping state the network stays disconnected and compacts and generalises the data to extrapolate general rules and concepts from the acquired information. Once these production rules have been acquired, the original information can be discarded to free memory.

Setiono and Lui [Set96] present their 'Neurorule' algorithm for rule extraction from a three layer, feed forward network. The algorithm uses weight decay to prune the network of irrelevant connections and units while maintaining its predictive accuracy. It then discretises the hidden unit activations by clustering and uses this to extract rules which are similar to those produced by decision trees.

3.20 Alternative Cost Functions

For completeness, we note that the quadratic error function presented in equation (3.5) is not the only possibility. We could have [Neu91]

$$E(\bar{w}) = \frac{1}{3}\sum_{\mu i} |(\zeta_i^\mu - O_i^\mu)|^3 \qquad \textbf{(3.23)}$$

yielding

$$\delta_i^\mu = g'(h_i^\mu)(\zeta_i^\mu - O_i^\mu)^2 \qquad \textbf{(3.24)}$$

or [Neu91]

$$E(\vec{w}) = \frac{1}{4}\sum_{\mu i}(\zeta_i^\mu - O_i^\mu)^4 \qquad \textbf{(3.25)}$$

yielding

$$\delta_i^\mu = g'(h_i^\mu)(\zeta_i^\mu - O_i^\mu)^3 \qquad \textbf{(3.26).}$$

Further alternatives for $\boldsymbol{\delta}$ are [Her91]

$$\delta_i^\mu = [g'(h_i^\mu)+0.1](\zeta_i^\mu - O_i^\mu) \qquad \textbf{(3.27)}$$

and

$$\delta_i^\mu = arctanh[\frac{1}{2}(\zeta_i^\mu - O_i^\mu)] \qquad \textbf{(3.28).}$$

We could even adjust the error function during training to smooth the error surface and avoid local minima [Her91]. This leads us to a discussion of optimisation techniques other than the gradient descent algorithm.

3.21 Alternative Minimisation Techniques

Various alternate optimisation techniques have been proposed, these include: steepest descent - a form of line minimisation; conjugate gradient - where new search directions are selected so as to spoil as little as possible of the optimisation achieved by the previous one; quasi-Newton - an iterative scheme; pseudo-Newton which uses both first and second derivatives; specialised stiff ordinary differential equation solvers and simulated annealing - a stochastic process controlled by a 'temperature-annealing' schedule [Her91, Car95].

3.21.1 Conjugate Gradient Methods

Back-propagation considers only the first derivatives of the error surface (its slope). Conjugate gradient methods speed up back-propagation by using second derivative information (the rate of change of the slope). Conjugate gradient descent does not proceed down the gradient but in a direction conjugate to the direction of the previous step. Gradient changes are perpendicular to the direction of the previous step and thus do not undo any previous minimisation performed by that step. An exact conjugate path involves computation of the Hessian - the matrix of second derivatives. This is computationally expensive [Loo96], so the algorithm approximates a series of conjugate steps instead of the full path [Hay94]. A quadratic error surface is assumed and close to the minima this is usually true, further away the results are less predictable [Was93]. The method requires good initial weights [Loo96] and relies on a line search to find the optimal value of the learning parameter in the cost function [Hay94]. Chen and Lai [Che93] note that conjugate gradient back-propagation is quicker than conventional back-propagation and is not susceptible to oscillations. The linear line search allows the algorithm to take the optimal step at each iteration and eliminates the choice of learning and momentum parameters.

3.21.2 Newton's Method

Unlike conjugate gradient, Newton's method calculates both the Hessian and its inverse. Not only is this computationally expensive, but the Hessian may be singular and hence non-invertible. The computational difficulty is further increased by the fact that the Hessian is almost rank-deficient [Hay94].

3.21.3 Quickprop

The Quickprop algorithm attempts to estimate second derivative information without calculating the Hessian. It assumes a parabolic error surface and an independent slope change in the weights [Was93, Loo96].

3.22 Genetic Algorithms

Caudill [Cau91] questions whether the initial state of the network (its "genetic" capability) is more important than the training it receives and notes that, as with humans, the answer may involve a combination of both. Producing neural networks is generally a trial and error activity. Genetic algorithms may be used to generate networks and optimise their weights [Fog90].

Genetic algorithms emulate the mechanisms of natural selection. They search high-dimensional spaces for solutions using robust, simple algorithms that assume no prior knowledge of the search space. Their robustness stems from the fact that they modify the characteristics of a population of solutions over many generations [Was93]. The population members (in our case they may be network weights) are coded as binary strings or chromosomes with some fitness function. The initial set is random, but successive generations are created from older ones using operators such as *mutation*, *crossover* and *reproduction* that mimic genetic transfer in biological systems with some form of 'survival of the fittest' [Par95]. An initial population (the first generation) is first produced. It may be described by an *objective function* which we seek to maximise. From the current population, reproduction produces the next generation that has, on average, a higher objective function. The number of times an element is copied to the next generation is proportional to the value of its objective function relative to those of the other elements. To produce new elements, crossover simulates the exchange of genetic material that occurs during mating by dividing elements and exchanging their bits. Mutation randomly perturbs a population's characteristics to prevent evolutionary dead-ends. Mutation rates are low due to its destructive nature. To mimic nature's diversity, genetic algorithms may be extended by operations such as diplody and dominance, inversion, sexual reproduction and uniform crossover.

Although they may appear chaotic, generic algorithms rest on a solid theoretical foundation. A *schema* S is a string constructed from the elements {0, 1, *}, where * is a wildcard. Positions in the string holding a 1 or 0 are called *definite bit positions*. A string matching a schema in all its definite bit positions *represents* it. The *order* of a schema o(S) is the number of definite bit positions, and its *defining length* δ(S), the distance between the leftmost and rightmost definite bit positions. The latter are called *specific positions*.

Mutation, crossover and reproduction may all be described by equations and their combined effect is captured in the Fundamental Theorem of Genetic Algorithms [Was93]:

$$n(S,t+1) \geq \frac{n(S,t)f(S)}{f(P)} \left(1 - P_c \frac{\delta(S)}{(l-1)} - o(S)P_m\right) \tag{3.29}$$

where

- n(S, t) is the number of representatives of schema S at time t,
- f(S) is the average objective function for the schema
- f(P) is the average objective function over the population,
- P_c is the probability of a cross-over during mating,
- $\delta(S)$ is the defining length,
- l is the string length,
- o(S) is the order and
- P_m is the probability of mutation at a given bit position.

Genetic algorithms do not require differentiable fitness functions and perform global searches of the weight space so they are impervious to local minima. They do, however suffer a computational penalty for not using gradient information. A suitable compromise might be an initial genetic search followed by a gradient method [Her91].

3.23 Hybrid Expert System - Neural Network Systems

Expert systems may be used to heuristically train a neural network. Nassersharif and Rajan [Nas93] present a real-time blackboard expert system that intelligently guides the back-propagation algorithm as it searches for a minima on the error surface.

Blackboard systems consist of several independent *knowledge sources* which communicate via one or more global *blackboards*. On the blackboard are several known facts and hypotheses. If one of the knowledge sources is able to make a contribution to the problem it writes something on the blackboard. This new information may prompt a contribution from one of the other knowledge sources and problem solving proceeds in this opportunistic manner, depending on those known and derived facts which present themselves upon the blackboard.

In Nassersharif's and Rajan's [Nas93] system the automated training blackboard system consists of several knowledge sources. An initial knowledge source decides the network architecture and initial learning parameters and weights. Several strategy knowledge sources then govern the training strategy using production rules. These match the momentum and training rates, change the sharpness of the activation function, modify the differential step or epoch size and identify and dynamically delete poorly learnt patterns. An error detection and correction knowledge source detects saturation of the sigmoid transfer function and network oscillations. A training overlord polls the knowledge sources and selects the most promising to make its contribution for a certain number of epochs, but it may select a new knowledge source at any point if the error decrease is abnormal. The overlord thus dynamically and opportunistically changes the training strategy for faster convergence. The authors claim that their scheme is faster and more robust than single parameter changes.

Communication between subsystems is an issue in hybrid systems. Hendler [Hen91] has used a back-propagation network to classify items corresponding to the leaves of a semantic network. Communication between the two networks is achieved by a marker passing mechanism.

Other authors [Lop93, Obr93, Shi93, Cec92] also present hybrid production rule or knowledge based systems with neural network components. Neural networks are not always the main focus of such hybrid systems. They may be used to research other AI fields, such as expert systems and logic.

Following Minsky's call to merge symbolic and connectionist approaches [Min91], Pinkas [Pin95] has developed a neural network that learns an extended propositional calculus and which can be extended to predicate logic as well. In essence, a connectionist inference engine learns propositional formulae and answers whether a query follows a formula either by compiling a symbolic formula or by learning it inductively. Pinkas notes that monotonic reasoning is non-adaptive, plagued by computational complexity and sensitivity to noise and argues that mapping symbolic computations onto a connectionist platform addresses these problems.

Malsburg [Mal87] presents a scheme in which a scene is represented by a set of active neurons. Each neuron encodes some elementary proposition about the scene with the activity level in the neuron proportional to the degree of confidence in the proposition. The knowledge base is encoded in the matrix of interconnections and a connection from neuron a to b is equivalent to the logical statement: proposition a implies proposition b.

The NEUROLET system developed by Quah et al. [Qua93a, Qua93b, Tan96] uses a neural network shell for rule-based reasoning. The system has the appearance of a rule-based expert system shell, but the underlying architecture is connectionist. The knowledge base contains heuristic rules, knowledge from training examples and past inference history. The inference engine, which may be run in consultation or training mode, chains active rules together into a back-propagation neural network. The system rules are fragments of the neural network which are fitted together dynamically during inferencing in a similar manner to backward or forward chaining in conventional expert systems. The authors claim the system emulates human adaptive and opportunistic behaviour.

3.24 Hybrid Fuzzy Logic - Neural Network Systems

Hybrid neural fuzzy learning systems employ an on-line fuzzy logic controller to adapt the learning rate parameters of the back-propagation algorithm to improve convergence.

The boundaries of a fuzzy set are not sharp. Membership functions map elements of the universal set to the real interval [0, 1]. The closer to 1 the mapping, the more the universal element "belongs to" the fuzzy element defined by the membership function. Set union, intersection and complement functions are also defined by membership functions. Fuzzy systems have four distinct mechanisms:

- A fuzzification mechanism that scales and transforms crisp values to fuzzy ones.
- A fuzzy rule base of if-then rules.
- A fuzzy inference mechanism.
- A de-fuzzification mechanism that scales and transforms fuzzy values to crisp ones. The centroid function is usually used for this [Ara96].

In fuzzy back-propagation, the if-then rules in the fuzzy rule base implement heuristics to control the learning rate and momentum parameters to speed up convergence. $\Delta\eta$ and $\Delta\alpha$ become fuzzy variables [Hay94].

Neural fuzzy systems is an emergent field that attempts to merge neural computing and fuzzy logic. Eklund [Ekl94] notes that no unifying paradigm has yet been developed, but offers the following table of comparisons:

Table 3.2 Comparing Neural Network and Fuzzy Logic System

Neural Network	Fuzzy Logic
Explores parameters.	Learns parameters.
Has learning rules/rates, transfer/error functions.	Has fuzzification (pre-processing) functions, inference mechanism, defuzzification functions.
Network type.	Inference mechanism type.
Hidden layers.	-
Preprocessing objective: convert data to numbers by scaling and normalisation to simplify the network's task.	Preprocessing objective: convert data to a logical form, thus extracting transformation function parameters as a basis for knowledge generation.

Funabashi et al. [Fun95] note that AI fuzzy systems are now into their third generation and classify such consolidated hybrid systems into four types. Systems using simple sequential combination; those where an integrator selects the most appropriate hybrid element to employ; fusion systems employing enhanced mathematical optimisation and associative systems with distributed architectures.

Khan [Kha93] presents a three layer neural network that represents the learning of fuzzy rules and membership functions. The first layer performs the fuzzification, the hidden layer forms the rule base and the third layer performs the rule evaluations and defuzzification process. Shann and Fu [Sha93] present a five layer neural network that acquires the fine knowledge of fuzzy neural systems. The respective layers handle input, membership functions, fuzzy AND, fuzzy OR and defuzzification. The coarse knowledge of the fuzzy system is encoded in the neural network structure and the back-propagation-like algorithm learns the fine knowledge. The authors note that their network converges faster than conventional back-propagation since it is only partially or sparsely connected.

Other authors [Zha93, Li93] also take a hybrid fuzzy - neural network approach, but the technologies need not be integrated into an application. Takagi [Tak93] has used neural networks and genetic algorithms to help design a fuzzy system by deciding the number of fuzzy rules, the shape of the membership functions and consequent parameters.

The convergence of the back-propagation algorithm may be improved by using fuzzy logic control. Hu and Hertz [Hu93] present a scheme where each neuron has its local learning rate dynamically adjusted by a fuzzy logic controller according to its output error and certain heuristic rules. The authors claim that two to three times more tests converge with their fuzzy back-propagation algorithm than with the standard version.

3.25 Hardware Implementations of Back-propagation

Many VLSI hardware implementations of neural networks are commercially available and exploit the parallel nature of neural systems. With respect to back-propagation, Erdogan and Hang [Erd93] have implemented a parallel back-propagation algorithm on a reconfigurable machine. Bofill et al. [Bof90] have implemented parallel back-propagation on a transputer network. We shall not consider any of these systems in detail, but it is worth noting that electronic components are 10^6 times faster than neurons, but 10^3 times less interconnected [Fel87] and while electronic circuits are fixed, the structure of neurobiological circuits changes every time learning occurs [Hop88].

3.26 Optical Neural Networks

It is interesting to note that some of the optical principles employed by the shock ray-tracer in chapter 2 may also be used in the construction of optical neural networks. In optical neural networks the connections between elements are not explicit - the physics of light propagation is controlled in such a way as to achieve weight connection. Such systems have inherent parallelism, high speed and dense interconnectivity. They utilise principles such as:

- Phase conjugation in photo refractive crystals and holography.
- Spatial light modulators with holographic lenslet arrays.
- Mirror array interconnections.
- Optical correlators.
- LED arrays in optic/electronic implementations.

The inherent parallelism referred to is due to the fact that light beams pass through each other without interacting [Was89]. A proper quantum-mechanical explanation of this intriguing phenomena cannot be attempted here. The observation that the non-interference of light beams is a consequence of the fact that the photons comprising the beam belong to the class of sub-atomic particles known as *bosons*, and which possess integral spin, will have to suffice. To explain the interference patterns produced by "interacting" rays of light *when they arrive at their destination* we have to consider the wave nature of light as opposed to it particle nature. Light is both a particle and a wave, but a discussion of this *wave-particle duality* is again beyond the scope of this thesis.

Vector-matrix networks use rows and columns of light emitting diodes and photodetectors and photographic film weight masks to perform vector-matrix multiplication. Such calculations are extremely rapid[6] with the speed largely independent of the matrix size, so these networks can be scaled up efficiently [Was89].

[6] < 1 nanosecond.

Synaptic or weight information can be stored in high density holograms[7] [Was89] and can be modified during operation. Optical correlators store reference images in a hologram. When a noisy or incomplete input image is applied to the system, it is simultaneously optically correlated with all of the stored reference images. The correlations are fed back to the input where the strongest reinforces the input image. The enhanced image passes through this loop repeatedly, resembling the stored image more closely on each pass [Was89].

More advanced architectures involve volume holograms in crystals and the use of Fourier techniques to produce an optical interconnect matrix [Was89]. Keller and Gmitro [Kel93] claim that holograms fabricated with electron-beam lithography have the capacity to connect 6700 input to 6700 output neurons at rates of 45 - 720 trillion connections per second.

Steck, Skinner and Behrman [Ste93] present an interesting all-optical (as opposed to electro-optical) back-propagation neural network. The refractive index in Kerr-type optical materials is described by a non-linear function. Thin layers of this material serve as processing layers of neurons and are separated by thick layers of linear material (air) which act as connecting layers of weights. Each non-linear layer acts as a plane continuum of neurons connected by a plane continuum of optical paths - both of which are limited only by the optical resolution of the system. The inputs and weights are a distribution of irradiance in 2D space. The weights vary the refractive index profile of the non-linear media. This varies the phase profile of an applied optical plane wave producing interference patterns as the forward signal plane wave propagates through the device in the third spatial dimension. The interference pattern, and hence output pattern, depend on the optical pattern input at the first layer and on successive weight layers. The error at the output is a continuous distribution whose direction is reversed and propagated backwards through the linear and non-linear optical layers to produce optical error signals that modify the light applied to the non-linear 'weight' layers. This back-propagation scheme is implemented in the *same physical device* as the forward calculations.

In simpler hybrid optical/digital neural networks learning is performed off-line on a digital neural network. The weights are then downloaded to an optical neural network to perform on-line, real-time classification [Cas92a].

Jeon et al. [Jeo92] consider the power requirements of very large neural networks. Between two NxN sheets of neurons there are N^4 connections. The authors give the interconnection limits of various technologies as follows:

Table 3.3 Interconnection Limitations

Technology	N^4 Interconnections
Electronics	$\approx 10^4$
Biology	$\approx 10^8$
Optics	$\approx 10^{12}$

[7] $\sim 10^{12}$ bits/cm^3.

They calculate that a fixed optical hologram would use 10^6 times less energy than an electronic computer of similar complexity, if such could be built.

Casasent [Cas92b] argues that new algorithms may be necessary to take full advantage of optical hardware and presents optimising and adaptive algorithms for an optical correlator.

Non-linear optical materials like gallium arsenide may be used to construct optical threshold devices [Ale90]. Barnes et al. [Bar92] have developed an opto-electronic 'neurochip' which has optical modulators and detectors fabricated on the same substrate as electronic components. The semiconductor material (indium phosphide) is transparent at the operating wavelength (1.5 μm). Wells and barriers divide the modulators into planar arrays and the chip carrier is drilled to allow optical transmission.

Optical devices are extremely promising, but they do have their own physical characteristics which do not always match up well with the requirements of artificial neural networks [Was89].

3.27 Further Inspiration from Physics

As this thesis has a strong physics bias, it is interesting to note that the extent to which neural network models borrow from physics extends beyond statistical mechanics. The non-linear Schrödinger wave equation of quantum mechanics[8] has been used to construct a time-varying probability density function applicable in radar and sonar tracking systems utilising spatio-temporal neural network architectures [Hay94].

Quantum neurodynamics has also been used to describe the biological interaction between glial cells[9] and neurons in the cognitive processing of dynamic information streams [Hay94].

Schempp [Sch93] has paraphrased Heisenberg's uncertainty principle[10] as follows:

Theorem: *It is impossible to assign precise space-time coordinates to neural events.*
Corollary: *The information content of neural holograms will be completely erased when the precise space-time coordinates of neural events are known.*

Schempp uses this formulation to postulate that the brain cannot completely explain itself! We shall not pursue this line of philosophical speculation any further.

---

[8] The equation gives the probability of locating a particle at a position in space.

[9] Small cells that do not generate active electrical signals like neurons.

[10] This principle notes the impossibility of simultaneously knowing both the exact position and exact momentum (or other such pairs) of a particle. It states an upper bound on the precision of such information and attributes the uncertainty to the intrusive nature of experimental observations at the atomic and sub-atomic levels - the very act of observing influences the observations.

3.28 Bioelectronics

A report [Rob94a, Rob94b] on an ARPA funded US bioelectronics consortium's progress in developing a bioelectronic computer is perhaps Orwellian. Tissue taken from the hippocampi of rat embryos was mechanically dissociated into single cell suspensions and grown onto an artificial substrate as a self-assembled monolayer. Lithographic patterns imprinted on the substrate material control the growth of the neurons into precise, functional, user-defined networks or circuits as shown in figure (3.6). A diethylenetriamine (DETA) substrate promotes cell survival, adhesion and growth, particularly of output axons, so controlling the polarity and direction of the circuit. The cells can survive in the substrate for several months. Their support environment is considered analogous to the cryogenics of modern supercomputers. Electric signals will be transmitted to, and read from the biological network via solid state transducers. Initial applications are in military sensor technology, but the researchers hope that the organic circuits will provide insight into more efficient learning algorithms for artificial neural networks.

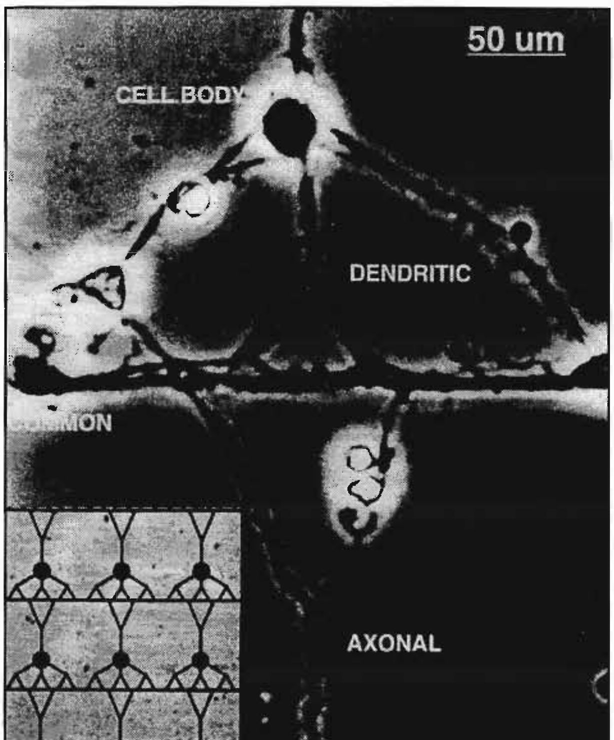


Figure 3.6 Embryonic rat hippocampal neurons cultivated on an artificial substrate. The insert shows the ultraviolet lithographic mask controlling cell growth. Reproduced from [Rob94b].

3.29 Biological Plausibility of Back-propagation

Although the neural network models introduced in this chapter were inspired by biological systems, we should consider the biological plausibility of the back-propagation algorithm more soberly before we move on, as no known biological system actually exhibits back-propagation [Was87].

Haykin [Hay94] summarises some of the main arguments against the biological plausibility of the back-propagation algorithm:

- Neurons in the brain are either excitatory or inhibitory, but not both.
- Back-propagation is subject to the *locality constraint* - the computations of artificial neurons are influenced only by those neurons with which they are in direct contact. In the brain however, hormonal and other global controls are critical for functions like learning.
- The modification of synaptic weights by presynaptic activity and error signals *independent of* any postsynaptic activity seems contrary to neurobiological evidence.
- Back-propagation would require rapid transmission of information backwards along axons. This appears unlikely in the brain.
- Back-propagation requires a teacher. The existence of some novel set of neurons in the brain to act as such is implausible.
- Back-propagation forgets what it has learnt when presented with new training patterns. This temporal instability is biologically implausible in a continuously changing environment [Was89].

There is however, legitimate AI interest in neural networks for their computational ability alone. When considering the correspondence of such models with the nervous system Durbin [Dur89] suggests that we should ask: "How weak can the correspondence be and still give worthwhile information?". He argues that research into artificial neural networks may suggest basic brain mechanisms, identify critical physiological properties or lead to new psychological models.

Miall [Mia89] observes that real neurons are anatomically distinct, unlike identical artificial neural network units. Biological tissue that is specialised for learning (e.g. the cerebellar cortex) is however more uniform and more comparable to artificial neural networks. He notes that real neurons are constrained by their evolutionary history and that we need not include every known detail about them in our models. Like Haykin [Hay94] he also notes that, contrary to earlier belief, neurons with excitatory outputs and neurons with inhibitory outputs both exist. Section (3.2.2) describes a mechanism by which "effective synapses" could have weights of either sign.

Barto [Bar89] suggests that we modify our assumption that the processing units in artificial networks are simple and that nearly everything of interest is the result of the interaction of many units. The network should be viewed as a "confederation of units that face special difficulties in achieving their local goals because of their spatial distribution, limited means of communication and lack of access to centralised control information".

Despite reservations about back-propagation's teacher, animal learning theorists agree that learning is governed by the difference between what is required and what is currently known - i.e. some error correcting rule is required [McL89].

Gardner [Gar93] acknowledges that the retrosynaptic transfer of error information in back-propagation weakens analogies to networks of biological neurons, but presents a *hypothetical* neuromorphic implementation of back-propagation. The error signal is a retrosynaptic chemical neurotransmitter released by the post-synaptic neuron to incident pre-synaptic receptors, of which there are two types. One forms the pre-synaptic error and the other modifies the weights of the local synapses. Gardner [Gar93] notes that there is some evidence to suggest that a Ca^{++} dependent process in the post-synaptic cells of the mammalian hippocampus synthesises NO which diffuses retrosynaptically to act on recently active pre-synaptic cells.

Mitchison [Mit89] expresses concern that the long convergence times of the back-propagation algorithm leads to problems when scaling up, but notes that relaxing the learning requirement to allow a proportion of errors reduces the scaling effect dramatically. Sejnowski [Sej87] explains that slow learning in back-propagation allows the hidden units to build up efficient internal representations. He notes that it takes long practice for humans to excel in a particular field. Haykin [Hay94] suggests that breaking networks into modular components may help combat the scaling problem.

Mitchison [Mit89] proposes that error correcting is a separate phase for learning in neuronal systems with a reverse pathway dedicated to credit assignment that fires occasionally, perhaps during sleep. In presenting a hypothetical neurophysical implementation of back-propagation, Hecht-Nielsen [Hec92] assumes, like Mitchison [Mit89], that the forward pass is active all the time and the backward pass is triggered only occasionally. The trigger may be some "mismatch detection", "search light" or "important event detector" that Hecht-Nielsen assumes arises in the thalamus and signals the cortex. His hypothetical model learns mappings between nearby regions of the cortex interconnected by axons of shallow pyramid neurons. Deep pyramidical neurons integrate the inputs, while the shallow pyramid and stellate cells carry out both the forward and backward passes and the application of the sigmoid transfer function. He explains that synapses in the brain that are not needed by his model may be statistically irrelevant or used in some other network circuit at other times.

In an earlier work [Hec88] Hecht-Nielsen observes that while artificial neural networks do not duplicate the brain, they offer some valuable, specialised, brain-like capabilities that are probably beyond the reach of algorithmic programming.

CHAPTER 4 Theoretical Data

We return now to the computational ray-tracer to consider the generation of the theoretical data on which the neural network is trained.

4.1 Simplifying the Density Profile

To capture both the gradients of the main shock features, as well as their spatial distribution, we wish to pass the position and density co-ordinates of the start of the shock front, rear, saddle, tail and central vacuum regions as parameters to the neural network. It is not possible to interpolate a Sedov-type curve over just these five points. To avoid introducing parameters which have no physical significance we seek to simplify our representation of the density profile.

Michaelis, Waltham and Cunningham [Mic91] report that the intensity pattern is strongly influenced by the *gradients* of the different components of the shock density profile. Their computational experience shows that, generally, the inner and outer fine fringes are determined almost entirely by the gradients of the shock front and rear respectively, while the coarse fringes depend strongly on the gradient of the shock tail [Mic91]. These shock features are approximately linear; the shock saddle (which has little bearing on the intensity pattern) being the most curvaceous section of the density profile which they use [Mic91]. Using this as a basis, we simplify the density profile to a series of straight lines. It is observed that the ray-tracer continues to produce reasonable results with this approximation. The (position, density) coordinates of the junctions of the straight lines parametrise the density profile.

4.2 Generating the Data-set

An initial density profile was selected, based on that published in paper [Mic91], and corresponding to a time 4 μs after the explosion. As indicated in figure (4.1) a grid was superimposed at the junctions of each of the density profile's five regions. New density profiles are selected by moving the base pattern's junction points to different cells in the grids. The grids at the start of the shock front and vacuum regions are of dimension 3x1 as the density at these points is fixed at 1 and 0 atmospheres respectively. The grids at the other points are of dimension 3x3. Some grid overlap on the position axis also limits the possible permutations as the shock regions are strictly sequential.

The density profiles are limited to a [0, 3000] micron position range and a [0.0, 2.5] atmosphere density range. The intensity patterns are limited to a [0, 4500] micron position range and a [0.0, 10.0] absolute intensity range. The intensity patterns produced need to be parametrised. This is done by extracting the positions and intensities of all extrema (peaks and troughs) in the intensity profile up to a maximum of 60 extrema.

A point is considered to be one of these extrema if:
(a) it is above an intensity threshold[11], and
(b) its neighbours are both below, or both above it.

[11] 0.0005 absolute intensity.

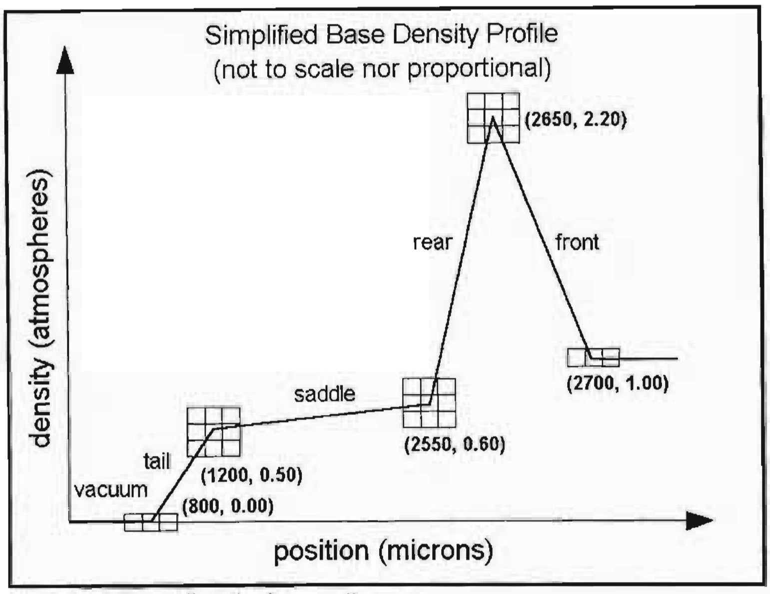


Figure 4.1 Expanding the base pattern.

4.3 Pruning the Data-set

Two series of grids were used independently of each other - a fine grid with resolution 50 microns and 0.05 atmospheres and a coarse grid with resolution 100 microns and 0.1 atmospheres. Over 7000 different density profiles (representing all valid permutations with both grid types) were run through the ray-tracer. Those intensity patterns generated from profiles permuted from the coarse grid proved too dissimilar to the base pattern and were discarded. Profiles with shallow gradients did not produce good results and the data-set was further restricted to profiles with steep gradients by limiting the displacement between consecutive junction points as follows:

Table 4.1 Data-set Pruning Criteria

Consecutive junction points (start of region)	Maximum Delta (μm)
Vacuum & Tail	450
Saddle & Rear	150
Rear & Front	100

The number of fringes present in each intensity pattern varied considerably and the data-set was again restricted to those patterns in the central region indicated in figure (4.2). Patterns outside this region are quite dissimilar to those within it and would confuse the network during training.

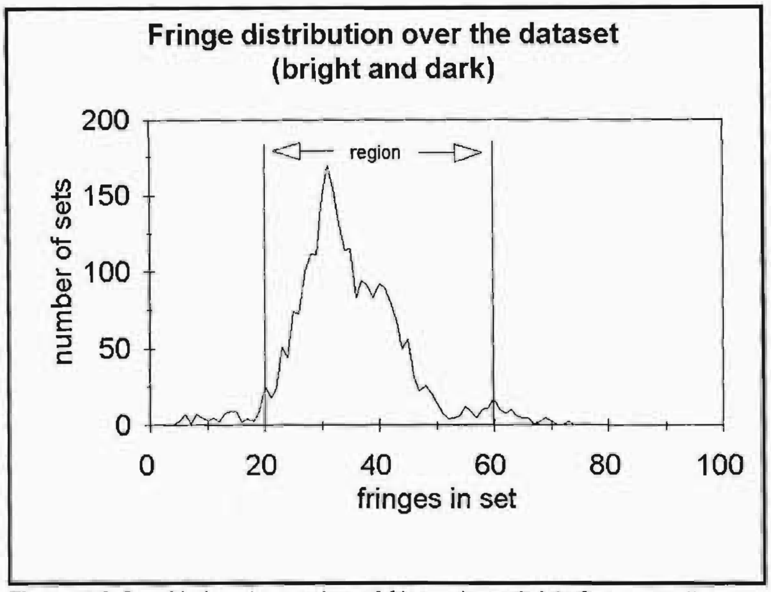


Figure 4.2 Considering the number of fringes in each interference pattern.

The pruning process outlined above resulted in a training set of 2400 density profile - interference pattern pairs. A further 56 pairs were reserved as a test-set of theoretical data distinct from the experimentally obtained interference patterns. The test-set thus represents 2.3 percent of the total theoretical data available. Despite pruning there is considerable variation of the intensity patterns of both the test and train sets as the ray-tracer proved quite sensitive to the shock gradients.

CHAPTER 5 Experimental Data

5.1 The Experimental Setup

Figure (5.1) shows the experimental setup. A spherical shock is generated in air by discharging a 1.1 pF, 15 kV capacitor across a pair of pointed electrodes 0.9 mm apart. The shock persists for several hundred microseconds during which it is probed by a pulsed ruby laser of 0.8 μm wavelength and 30 ns duration. The refracted beam produces an interference pattern that is captured and digitised by a CCTV camera and an Oculus frame grabbing card. The camera focuses on an image plane 16 cm distant from the centre of the shock. The firing of the laser and the generation of the shock are synchronised by a personal computer with the time delay between the two measured by a photodiode, search coil and digital storage adapter arrangement. Note the neutral density filter situated before the camera lens; its function is discussed in section (5.3).

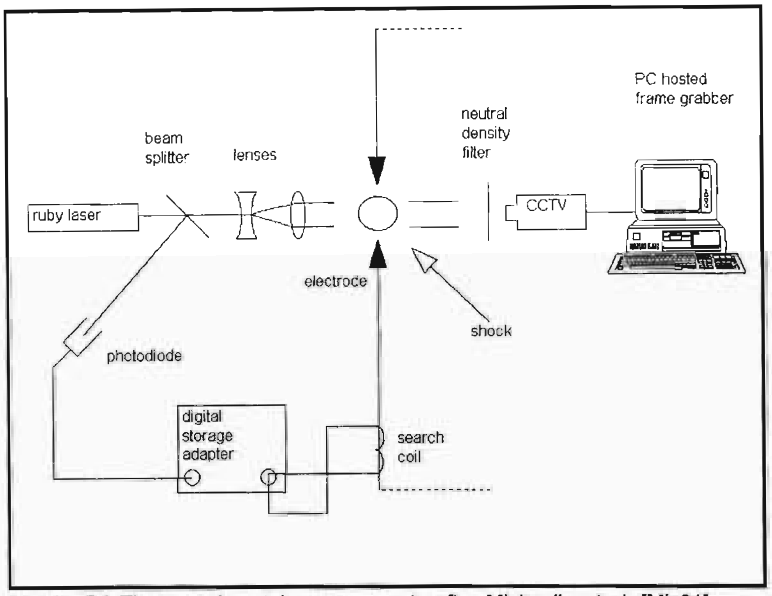


Figure 5.1 The experimental arrangement, after Michaelis et al. [Mic91].

5.2 Image Acquisition

The experimental images supplied by the Plasma Physics Research Institute at the University of Natal were captured by an Hitachi HV 720E CCTV camera attached to an Oculus 200 frame grabbing card. The Oculus card digitises video images into an array of 480x512 pixels [Ocu87], with each pixel assigned an integer value between 0 and 127. This value is proportional to the average light intensity incident on the corresponding unit of the TV tube, but the relationship is not linear - the camera response is logarithmic. A further complication is that the image consists of two, interleaved, fields referred to as the *bright* and *dark* fields and clearly visible in figure (5.3.a). The bright field has a higher average pixel value. Note that there was not enough laser light to illuminate the entire interference pattern - only the left hand side and a portion of the right were captured. Since the pattern is radially symmetric, this does not pose any problems. The black columns at the extreme left of the image are the so-called 'dead-pixels' added by the card to give the image a uniform resolution of 512x512 pixels. Figure (5.3.b) shows the bright field extracted from figure (5.3.a).

5.3 Linearising the Image

Waltham [Wal90] describes the Institute's video analysis system and we follow his discussion of the calibration technique employed. Under dark background conditions the ruby laser was pulsed through a diffusing screen and various calibrated neutral density filters. The transmitted light was digitised and the average pixel number for each filter noted, allowing the response curve of the video system to pulsed ruby radiation to be determined. Two response curves were generated in this manner, one for each field of the digitised image. These curves originally related pixel values to the relative logarithm of intensity (or relative density), but are presented in figure (5.2) as a straight mapping between pixel values and light intensity. This data was also supplied by the Plasma Physics Research Institute.

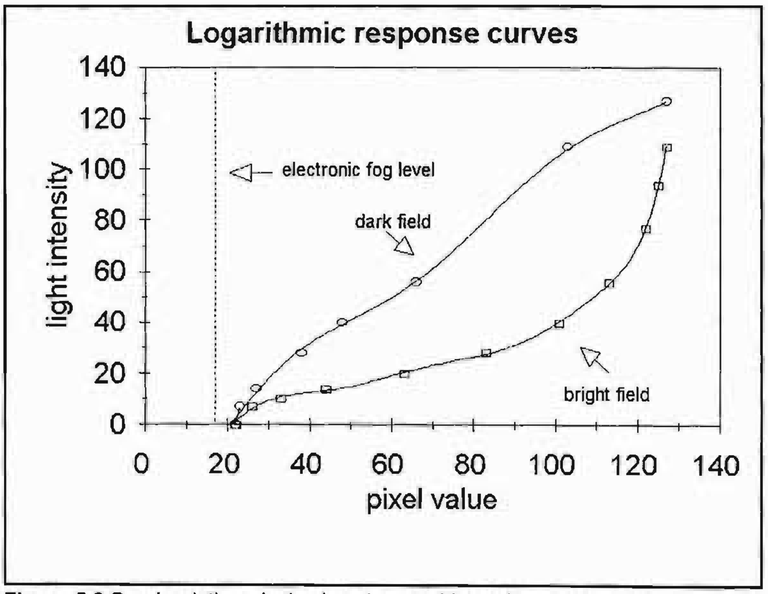


Figure 5.2 Graph relating pixel values to actual intensities, data points represent measured values.

Note that the bright field response curve has a larger and more uniform distribution of measured data points. We shall consider only the bright field of the captured images.

5.4 Scaling the Image

No scaling information was supplied with the images, but we do know that the spark discharge gap between the electrodes is 900 microns. The electrodes cast clear shadows in the captured images (particularly Image a) and the gap can be measured as approximately 60 pixels. This yields a scale of 1 pixel : 15 microns which, because of the 1:1 aspect ratio, we use for the horizontal direction as well.

5.5 Extracting the Intensity Pattern

The bright field of each image was first extracted and saved. The horizontal row of pixels lying midway between the shadow of the electrodes was extracted, linearised using the curve for the bright field, and saved. Using a single row proved to be the safest approach. It is not desirable to average rows from the different fields together, and a window for a single field represents just half the actual height i.e. a r_{i-1}, r_i, r_{i+1} window for a field actually captures rows r_{i-2}, r_i, r_{i+2} of the complete image. Since the image is radially symmetric, averaging horizontal rows even a small distance apart can lead to slight inaccuracies. Table (5.1) lists the extracted rows of each image.

Table 5.1 - Bright Field of Captured Images

Image[12]	Time (μs)	Central X Pixel	Central Y Pixel
a	0.0	395	117
b	1.0	395	114
c	1.6	395	117
f	4.0	395	113
e	6.8	395	114
d	10.0	395	115
g	12.0	395	113
h	160.0	395	113
i	330.0	395	112

5.6 Smoothing the Intensity Pattern

The raw extracted intensity patterns are quite uneven and are smoothed for graphical presentation using a simplified least squares technique where each data point is recalculated as the weighted average of its original value and the surrounding data points. A convolution kernel of size 7 is used with the weights of table (5.2).

Table 5.2 - Convolution Kernel

i-3	i-2	i-1	i	i+1	i+1	i+2
-2	3	6	7	6	3	-2

[12] The naming of images 'f', 'e', 'd', and their extracted intensities, in tables (5.1), (5.4), (8.7) and (8.8) is deliberately out of sequence to maintain consistency with the Plasma Physics Research Institute's original labelling. The time column of table (5.1) orders the images chronologically.

The processes described in sections (5.5) and (5.6) are illustrated in table (5.3) for image (f), and summarised in table (5.4) for all the captured images. The printed images presented here have not been linearised. Instead, to make the fringes more visible, they have been histogram-equalised and stretched from 128 to 256 greyscales. Histogram equalisation averages the image brightness and so compensates for the poor spatial coherence of the incident laser light (which is discussed in section 5.7). This technique appears to improve the image contrast as well. The zero of position for the intensity graphs is situated between the two electrodes, at the centre of the spherical shock.

Table 5.3 - Extraction, Linearisation & Smoothing of Image Intensities

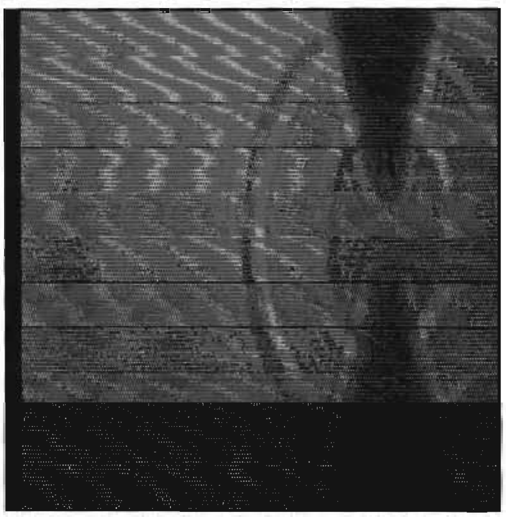


Figure 5.3.a Interleaved dark and bright fields in Image (f).

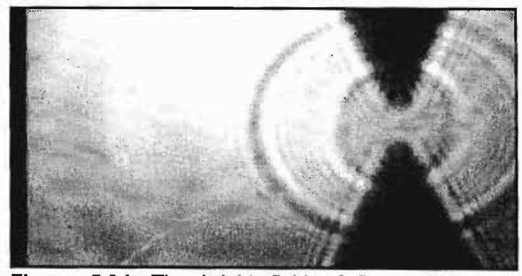


Figure 5.3.b The bright field of figure (5.3.a) is extracted.

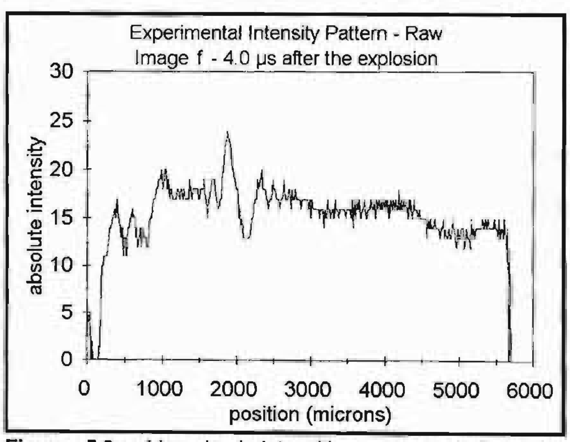


Figure 5.3.c Linearised intensities extracted from the central row of figure (5.3.b).

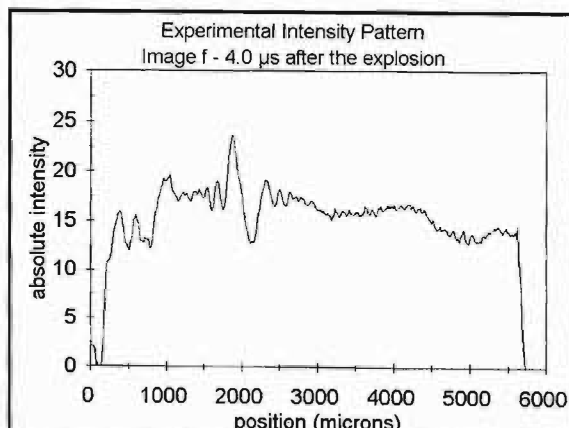


Figure 5.3.d Smoothed Linearised Intensities of figure (5.3.c).

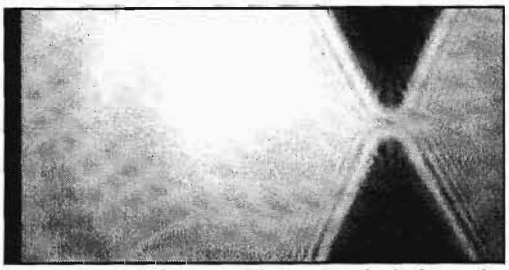
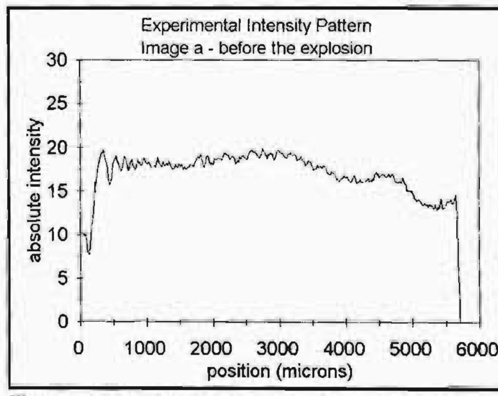
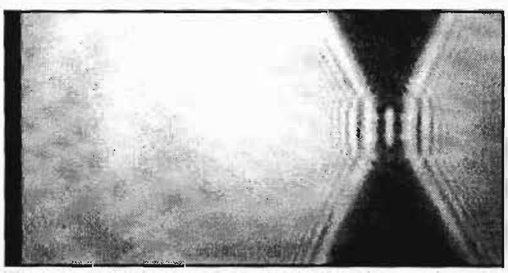
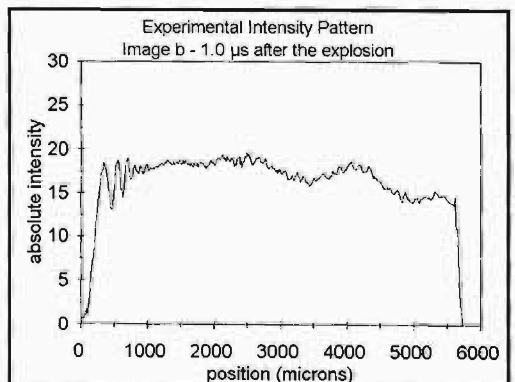
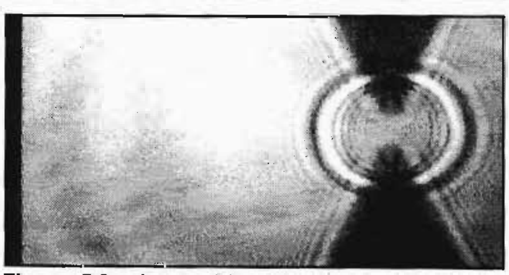
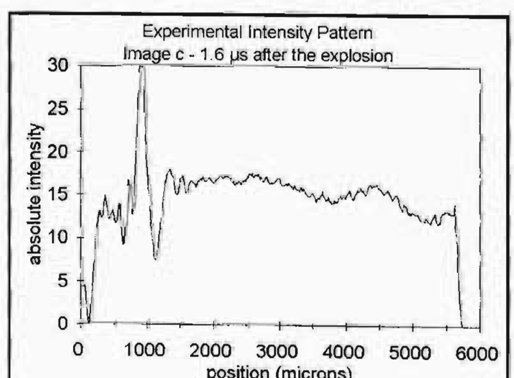
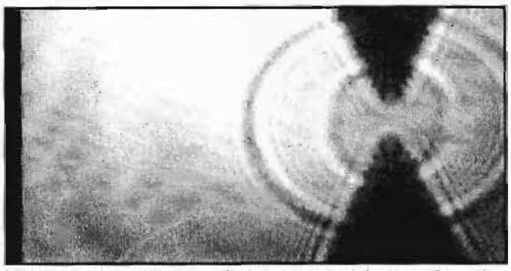
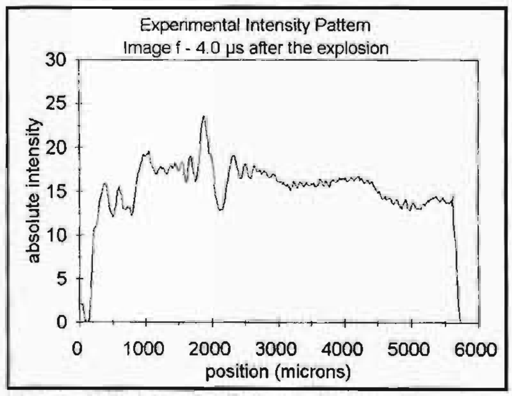
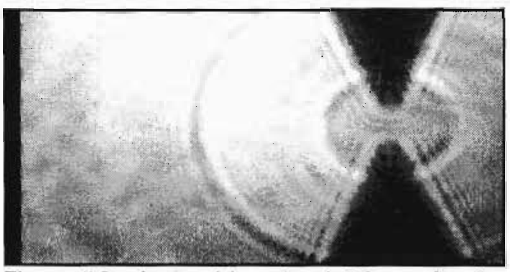
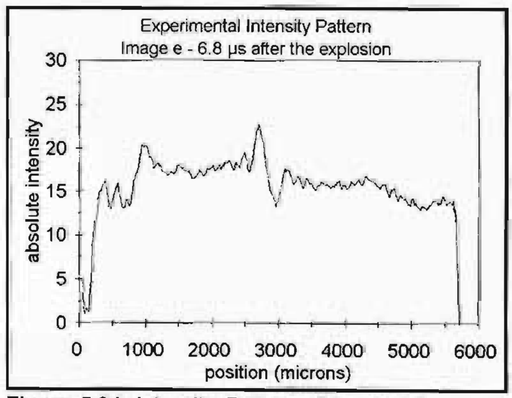
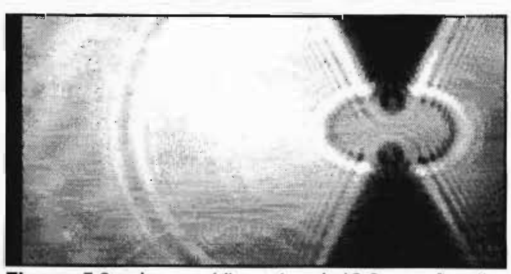
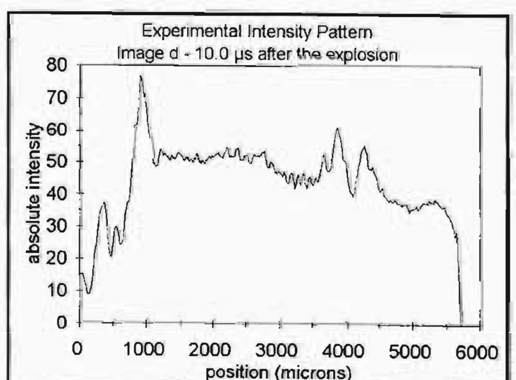
Table 5.4 - Experimental Data	
Image	**Extracted Intensity Pattern**
 **Figure 5.4.a** Image (a) captured before the discharge (0 μs) - the electrodes cast clear shadows and noise is evident towards the sides of the image. Note the linear Fresnel fringes around the electrodes. These are due to diffraction effects, not interference.	 **Figure 5.4.b** Intensity Pattern of Image (a).
 **Figure 5.5.a** Image (b) captured 1.0 μs after the discharge - the shock is still cylindrical.	 **Figure 5.5.b** Intensity Pattern of Image (b).
 **Figure 5.6.a** Image (c) captured 1.6 μs after the discharge - the shock starts to assume spherical proportions.	 **Figure 5.6.b** Intensity Pattern of Image (c).

Table 5.4 - Experimental Data

Image	Extracted Intensity Pattern
 **Figure 5.7.a** Image (f) captured 4.0 μs after the discharge - the shock is spherical. Our theoretical investigations concentrate on this time.	 **Figure 5.7.b** Intensity Pattern of Image (f).
 **Figure 5.8.a** Image (e) captured 6.8 μs after the discharge - the shock continues to expand.	 **Figure 5.8.b** Intensity Pattern of Image (e).
 **Figure 5.9.a** Image (d) captured 10.0 μs after the discharge - the shock continues to expand.	 **Figure 5.9.b** Intensity Pattern of Image (d).

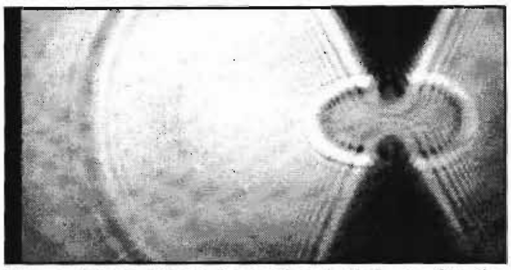
<table>
<tr><th colspan="2">Table 5.4 - Experimental Data</th></tr>
<tr><th>Image</th><th>Extracted Intensity Pattern</th></tr>
<tr>
<td>


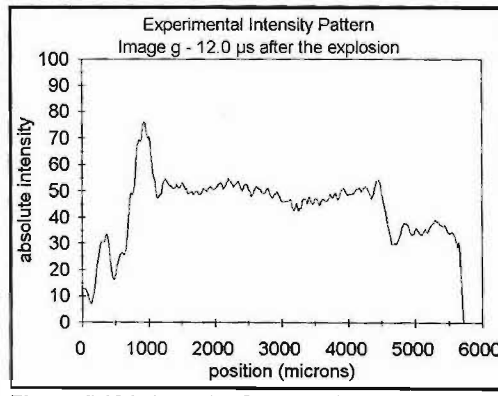
Figure 5.10.a Image (g) captured 12.0 μs after the discharge - the shock is weakened but the shock tail bright ring has remained virtually stationary in the last few images.
</td>
<td>


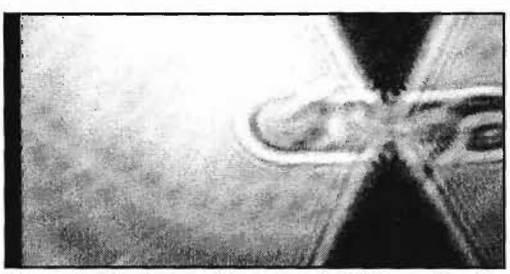
Figure 5.10.b Intensity Pattern of Image (g).
</td>
</tr>
<tr>
<td>


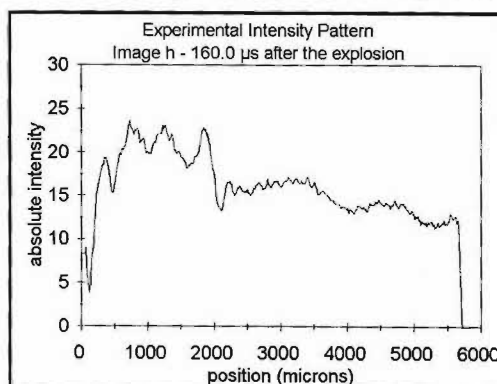
Figure 5.11.a Image (h) captured 160.0 μs after the discharge - the shock front has left the field of view and the shape of the central ring defining the vacuum region has become toroidal.
</td>
<td>


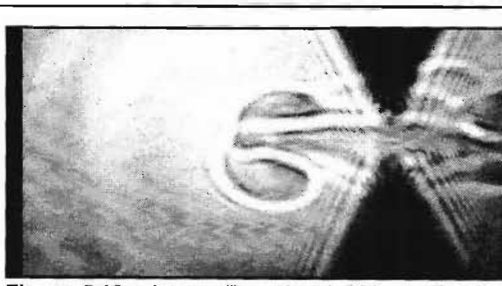
Figure 5.11.b Intensity Pattern of Image (h).
</td>
</tr>
<tr>
<td>


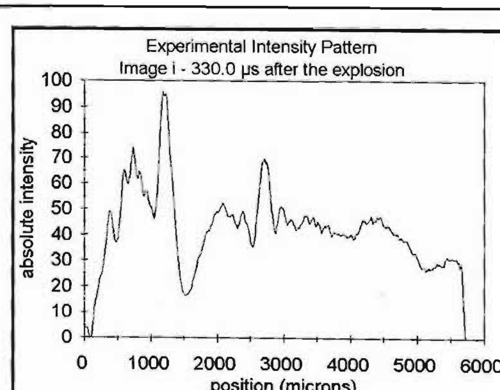
Figure 5.12.a Image (i) captured 330 μs after the discharge - it is now extremely late and turbulent behaviour is observed as gas flows through the rarefaction wave, which is now stationary.
</td>
<td>


Figure 5.12.b Intensity Pattern of Image (i).
</td>
</tr>
</table>

5.7 Determining the Fringe Locations

Michaelis, Waltham and Cunningham [Mic91] concede that the spatial coherence of the laser beam used in the experiment was patchy. This is evident in image (a), captured before the spark discharge. The pixel intensities diminish away from the centre of the image and a fair amount of "noise" is evident in the plot of the extracted central row. This noise is similar to that present in the other images, particularly towards the outside of the image, so we can use it to help determine where the fringes end. The exact fringe locations and intensities are measured from the unsmoothed intensity pattern, but reference to both the smoothed pattern, and a histogram-equalised version of the image, help to determine the fringe locations. This is necessary as the pixel intensity variation is slight for some of the fringes and masked to some extent by the "noise" introduced by the patchy spatial coherence of the probing laser beam.

CHAPTER 6 Using the Raytracing Program

6.1 Introduction

The ray-tracer is a DOS based text and graphics application with a mouse-driven menu and dialog box user-interface for interactive use. It can also be run in batch mode using a macro file, the format of which is discussed in Appendix B. For robust batch processing, it has proved expedient to construct a version of the ray-tracer, stripped of its user interface, that accepts command line parameters from a DOS batch file. Numeric errors that may cause the program to return to the operating system do not halt processing of the rest of the batch file. The format of this DOS batch file will also be presented in Appendix B.

The ray-tracer program requires at least a VGA graphics monitor and (despite the use of overlays) makes large demands on conventional memory. Network and other device drivers may need to be unloaded or relocated to the upper memory blocks for the ray-tracer to work correctly. A heap-size indicator on the *status line* at the bottom of the screen shows the available memory. Operations requiring more memory are safely ignored. Figure (6.1) shows the *menu bar*, *status line* and expanded *menu tree*. The various **Save** and **Load** options open a common file *dialog box* that displays and prompts for a filename. All dialogs, including message, confirmation and warning dialogs may be dismissed by clicking on the **Cancel** *button* or **Close** (**[*]**) *icon*. The macro facility mentioned earlier is invoked by selecting the **Run Macro File** menu option.

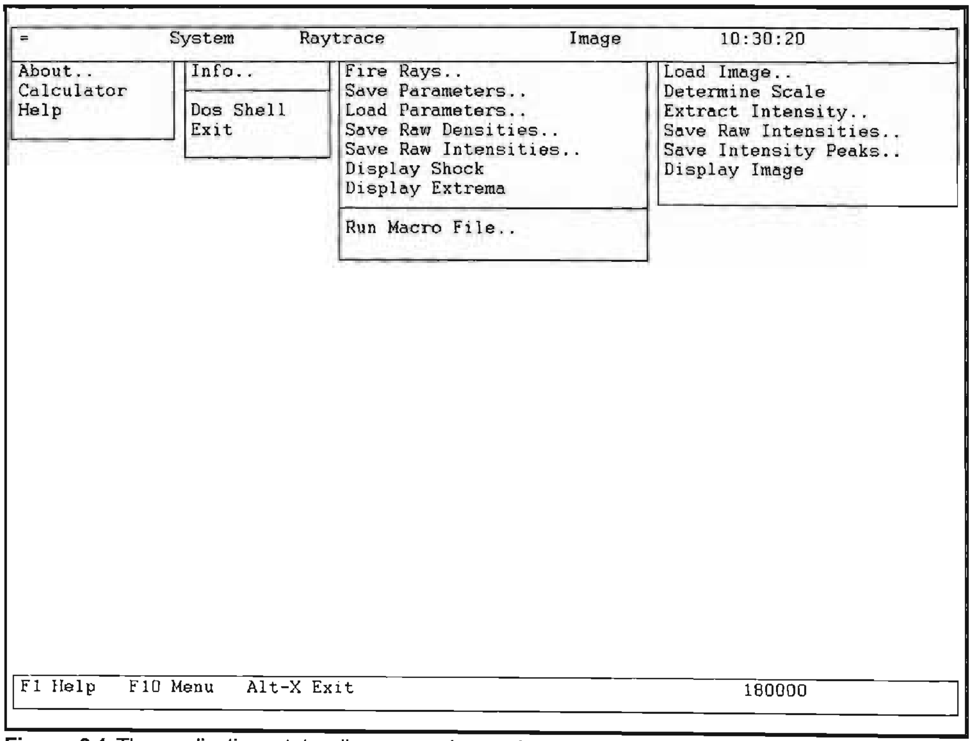


Figure 6.1 The application status line, menu bar and expanded menu tree.

6.2 Setting the Density Profile

A typical raytracing session would start with the selection of the **Fire Rays** menu option which opens the **Set Density Profile** dialog box of figure (6.2). The position and density co-ordinates of the end of each shock region is entered into the appropriate *numeric input lines* and a method selected from the **Interpolation** *radio-box cluster* to interpolate the five points.

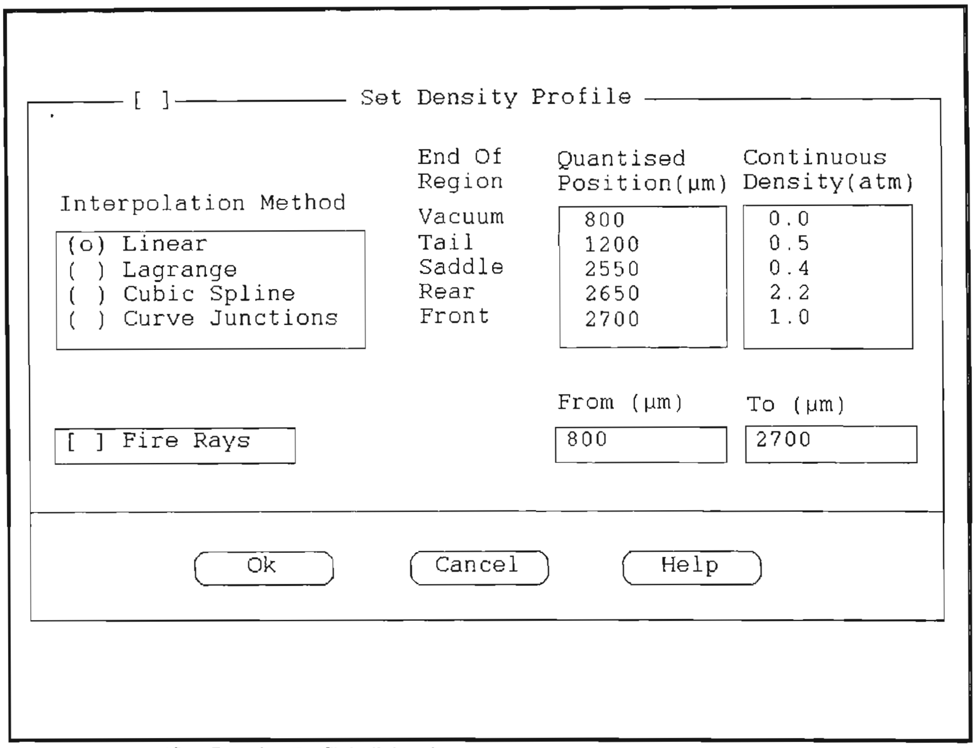


Figure 6.2 The 'Set Density Profile' dialog box.

Clicking on the **Ok** button at this point merely displays the interpolated density profile. When satisfied with this profile, the user fires the rays by reinvoking the dialog (the entered data is persistent), entering the start and end limits for the raytracing in the **From** and **To** input lines, checking the **Fire Rays** *checkbox* and then clicking the **Ok** button. Raytracing may take several minutes to complete[13], during which time the density profile, its parameters and a progress indicator are displayed.

[13] Approximately 4 minutes on a 100 MHz 486 DX4 PC.

6.3 Firing the Rays

The intensity pattern is displayed once it is generated. Any mouse click or keypress returns the user to the menu system from where the new data may be saved, either in raw format or parametrised for use by the neural network. The density profile is parametrised using the five points entered and the intensity pattern parametrised by locating its extrema (peaks and troughs). These may be interpolated and displayed for comparison with the original pattern by selecting the appropriate menu options. When an existing parameter file is loaded into the ray-tracer the density profile is interpolated using the saved interpolation method and the intensity pattern interpolated for display purposes using a cubic spline method. A screen display of such a situation is shown in figure (6.3).

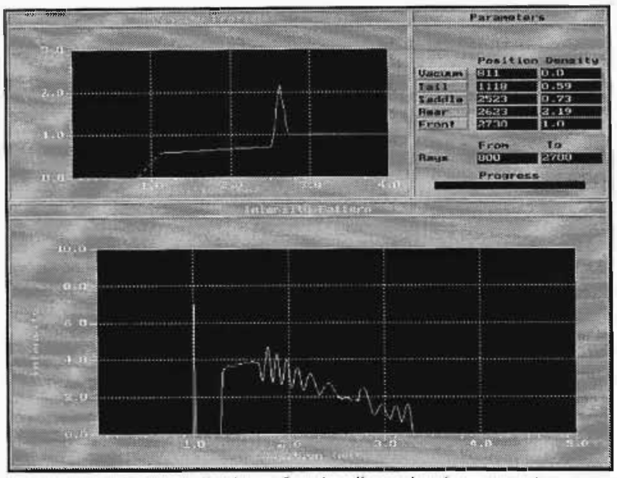


Figure 6.3 Screen display after loading shock parameters.

6.4 Extracting the Image Intensities

To extract an intensity cross-section from an experimentally captured image of an interference pattern, an image file is first loaded and then scaled. The **Determine Scale** menu option displays the selected image and a vertical scale bar. This bar is moved and sized (using the arrow, page, shift and enter keys) to measure the distance between the shadows cast by the two electrodes which are a known distance apart. A pixel-micron scale is thus established and the intensities may be extracted using the **Extract Intensities** options. This option opens the dialog box shown in figure (6.4). This dialog permits selection of the size of the horizontal data **Window** which scans the image, as well as the **Linearisation** response curve (if any) to apply to the pixels within the data window. Clearing the **Clear Previous Plots** checkbox allows the results of several trials to be superimposed for comparison.

When the **Ok** button is clicked the image is displayed, together with a horizontal scan bar which is manoeuvred in a similar manner to the scale bar. When the enter key is pressed the data window is positioned around the scan bar and the image intensities extracted and plotted as indicated in figure (6.5). Further trials may be attempted until the user is satisfied. The escape key returns to the menu system from where the intensities may be saved, either in raw format or as located peaks and troughs.

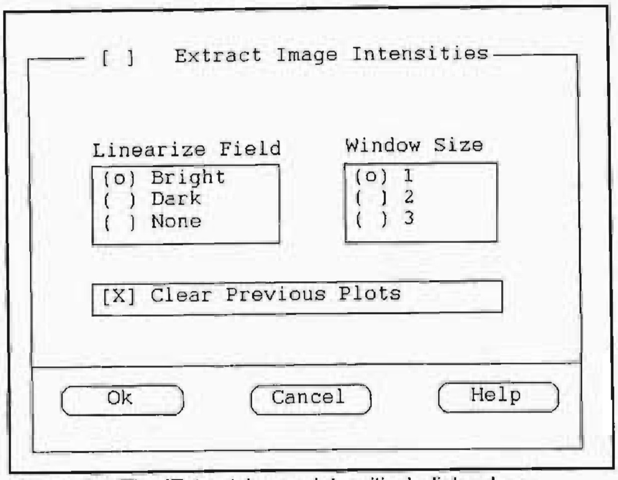


Figure 6.4 The 'Extract Image Intensities' dialog box.

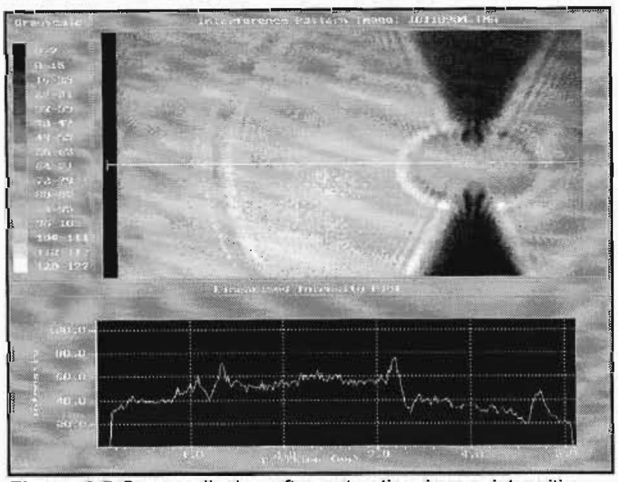


Figure 6.5 Screen display after extracting image intensities.

6.5 Software Engineering

The ray-tracer code has been properly analysed on a CASE (Computer-Aided Software Engineering) tool using the Shlaer-Mellor Object Orientated Methodology and concepts such as entity relationships, data dictionaries, state machines, event lists, data and control flows and processes. The results of this exercise and software metrics for both the analysis (DeMarco's Bang) and code (Halstead, McCabe, etc.) are presented in Appendix D, and a full source listing given in Appendix E. The analysis and metrics are situated in an appendix as they digress from the main thrust of this thesis. Their relocation should not, however, be allowed to diminish their importance.

CHAPTER 7 Neural Network Architecture

The NeuralWorks Explorer[14] package was used to create a standard back-propagation network that would learn to behave as an *inverse* ray-tracer. When presented with an interference pattern, the network must try to determine the density profile underlying the shock that gave rise to it.

7.1 Normalisation

To normalise the data, all values of position, density and intensity fed to the network were scaled to lie within the range [0.0, 1.0]. Network responses are also within this range and have to be scaled to the problem domain. The ranges of the problem domain are tabulated in section (4.2).

7.2 The Input Layer

The input layer represents the intensity pattern. Each bright or dark fringe (peak or trough) in the pattern is represented by two sequential neurons, the first representing its position in microns, the second its absolute intensity. 120 neurons are necessary to represent the maximum of 60 peaks. If there are fewer than 60 peaks, the extra neurons have their values set to zero. The number and positions of the fringes will vary, but we expect that the major bright and dark ring features will always be present and will act as landmarks in the pattern.

7.3 The Output Layer

The output layer represents the density profile. Each junction in the profile is represented by two sequential neurons representing the point's position (in microns) and density (in atmospheres). As the shock propagates away from its centre, we consider each shock region to start from the end towards the outermost section of the shock. Ten neurons thus encode the start of the following shock regions - the central vacuum, tail, saddle, rear and front. During training each neuron will always have a valid value.

7.4 The Hidden Layer

Although back-propagation networks may possess multiple hidden layers, a single layer suffices for most applications and only single and double hidden layer configurations were investigated. Selecting the number of neurons in the hidden layer is a common problem. The outer layers interface to the problem domain and their characteristics are largely determined by application-specific details. In general, the more complicated the relation between the input and output patterns, the more neurons are necessary to learn that relationship. However, the greater the number of hidden neurons, the longer it takes the network to converge, so the number of neurons in the hidden layer is a parameter which we seek to minimise once convergence is established. Usually the number of neurons in the hidden layer lies somewhere between that in the input and output layers, but we can refer to the literature for more quantitative estimates.

[14] Version 4.0 from Neuralware Inc., Pittsburgh, USA. The Explorer version of the software is restricted to 150 neurons and lacks some of the more advanced tools and features (such as network pruning) of the full-blown version.

7.4.1 The Neuralworks Estimate for the Number of Hidden Units

The Neuralworks manual [Neu91] presents a rule of thumb for setting an upper bound on the number of neurons in the hidden layer based on the amount of training data available

$$hidden = \frac{N}{\beta \times (output + input)} \qquad \textbf{(7.1)}$$

where

- N is the number of training sets and
- ß ranges from 2 (clean data) to 100 (noisy data), but is usually within 5-10.

Since our theoretical data is noise-free, we set ß=2, and using N=2400, input=120 and output=10 we arrive at an estimate of 9 neurons in the hidden layer.

7.4.2 The Overdetermined Approximation for the Number of Hidden Units

Carpenter and Hoffman [Car95] consider the problems of overtraining and generalisation in a neural network. If too many neurons are present in the hidden layer, the network's ability to generalise beyond what it has learnt is impaired. They argue that such underdetermined networks yield non-unique approximations and are susceptible to overtraining when too many training iterations are run. With analogy to polynomial approximations, these authors present an equation that limits the number of neurons in the hidden layer with regard to the amount of training data available and the degree of overdetermination required.[15] The undetermined parameters associated with a neural network are all its weights and biases, so the number of such parameters, P, for a network with one hidden layer is simply

$$P = hidden(input + 1) + output(hidden + 1) \qquad \textbf{(7.2).}$$

Now setting N=αP, we obtain

$$hidden = \frac{\frac{N}{\alpha} - output}{input + output + 1} \qquad \textbf{(7.3)}$$

where

- N is the number of training sets and
- α determines the degree of overdetermination (e.g. α=1.5 yields a 50% overdetermination).

Using the recommended value of α=2 (100% overdetermination) in equation (7.3), and setting N=2400, input=120 and output=10, determines the number of neurons in the hidden layer to be 9, the same as that determined from equation (7.1). This is less than the number of neurons in the output layer, so we set α=1.5 (50% overdetermination and a value described as reasonable in paper [Car95]) to obtain a value of 13 neurons. Thus the number of neurons in the hidden layer is set to 13.

---

15 Note that the constant α is erroneously presented as a numerator in the original paper.

7.5 The Convergence Criteria

A common way of determining when to stop training is to do so when the RMS error (refer to section 3.6) drops below a certain threshold. Deciding what this threshold should be is complicated in our case by the fact that the output neurons represent two different values (position and density) with different ranges in the problem domain. Nonetheless, if we scale a dimensionless RMS value of 0.001, say, to equivalent error values in the problem domain, we can expect errors of the order of 3 microns for position and 0.0025 atmospheres for density. This is more than acceptable, particularly when we recall that position is quantised into microns by the ray-tracer. It is best to select an optimistic value for the RMS convergence threshold and see if it is achieved, so 0.001 is the value we settle on.

7.6 Network Parameters

Table (7.1) lists the parameters of the network finally used for testing.

Table 7.1 - Network Parameters

Transfer Function	Sigmoid
Learning Rule	Delta
Bias Unit	Connected
Momentum	0.9
Learning Coefficient	0.6
1st Transfer Pt	None
RMS Convergence Threshold	0.001
Randomise Learning	Enabled
Epoch Size	2400
Input Range	0.0..1.0
Output Range	0.1..0.9
Input Neurons	120
Hidden Neurons	13
Output Neurons	10

Note that the epoch size is usually set equal to the number of patterns in the training file to ensure that each pattern is presented at least once, on average, during each epoch or training cycle. To prevent the network from learning any trends in the data ordering, rather than the data relationships itself, the order of pattern presentation is randomised on each cycle. To avoid saturation of the sigmoid transfer function at its endpoints, it is usual to set the network ranges for the output units to values close to, but not equal to the endpoints - hence 0.1 and 0.9. The momentum and learning co-efficients hold for both the hidden and output layers and remain constant throughout training. Usually, to prevent oscillations, these parameters are reduced to slow down learning as the network approaches convergence; however oscillations did not prove to be a problem with the network.

In additional experiments the epoch size was changed to 1 to test pattern (as opposed to batch) updating, and to 240 to search for 'resonance frequencies' within the training data-set. A second hidden layer of 7 neurons[16] was also tried. Separate learning rates were tried for the hidden and output layers, and a learning schedule that changed the momentum and learning rate at fixed intervals was also attempted.

[16] All that remained of the Explorer software's original complement of 150.

CHAPTER 8 Results

8.1 Convergence Performance

As suggested in section (3.13), network training and testing are alternated to prevent overtraining and consequent loss of generalisation. The RMS error of both the training and testing sets are plotted in figures (8.1) to (8.4) for various network configurations.

Table 8.1 - Network Training and Testing RMS Error Convergence

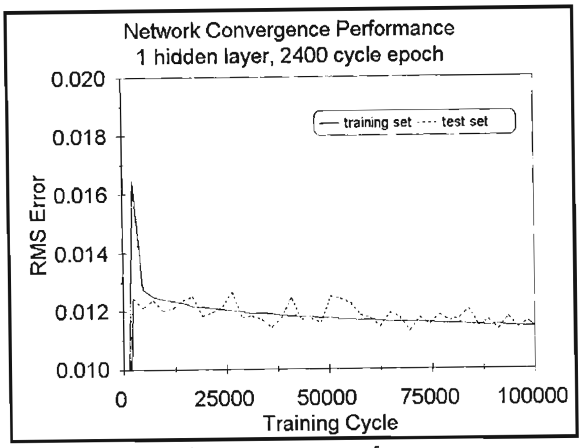


Figure 8.1 Network convergence performance.

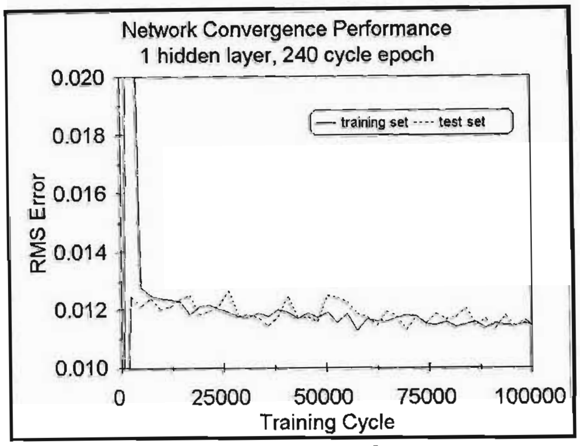


Figure 8.2 Network convergence performance.

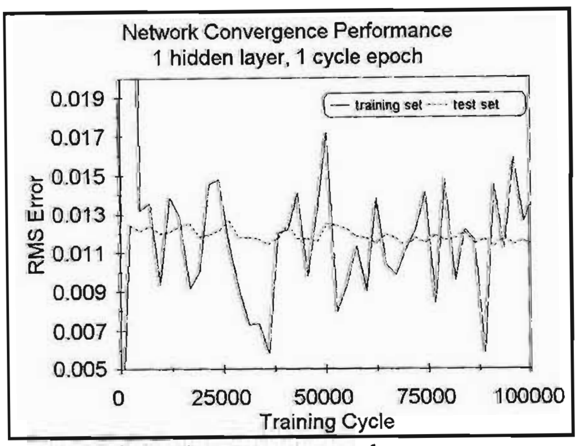


Figure 8.3 Network convergence performance.

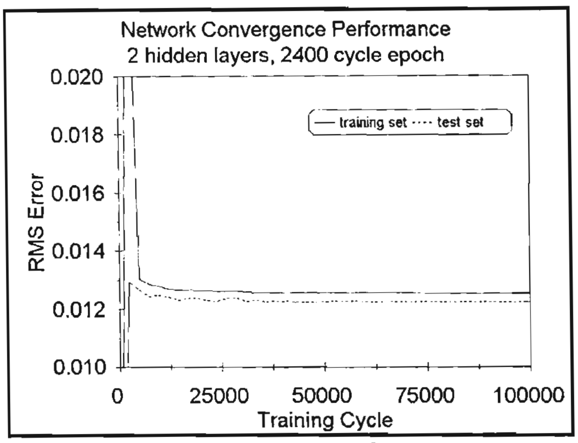


Figure 8.4 Network convergence performance.

Note the increasing oscillation in the training RMS error with decreasing epoch size as the path of the back-propagation algorithm through the weight space becomes stochastic. Learning is much smoother with an additional hidden layer. None of the additional networks show any improvement in either the training or testing RMS error over the original single hidden layer, 2400 cycle epoch network. Various changes to the learning rate and momentum parameters, as discussed in section (7.6), also produced no improvement in convergence.

Analysis of the weight distribution at the output layer, summed over an epoch, indicate that the weights do not alter much after their initial movement away from their random initialisation allocation around zero. Figures (8.5) to (8.8) show the weight distribution at various times for all the network variants considered above.

Table 8.2 - Network Output Layer Weight Distribution

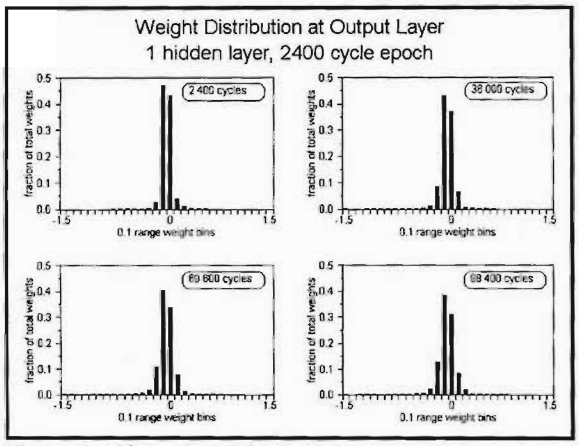


Figure 8.5 Network weight distribution.

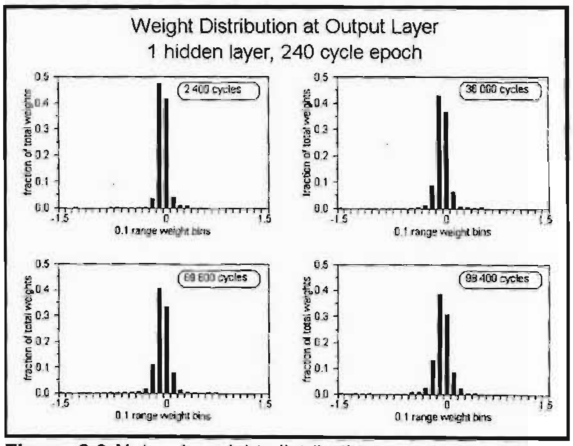


Figure 8.6 Network weight distribution.

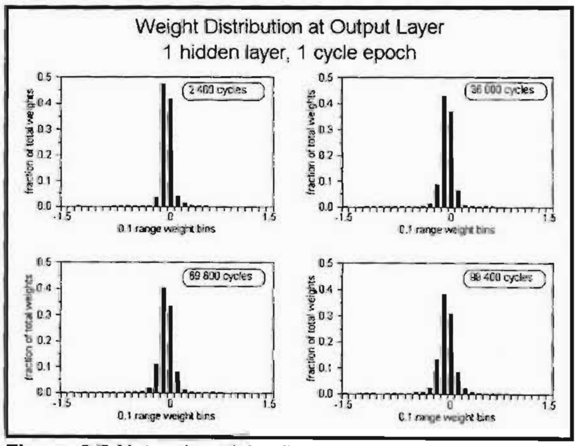


Figure 8.7 Network weight distribution.

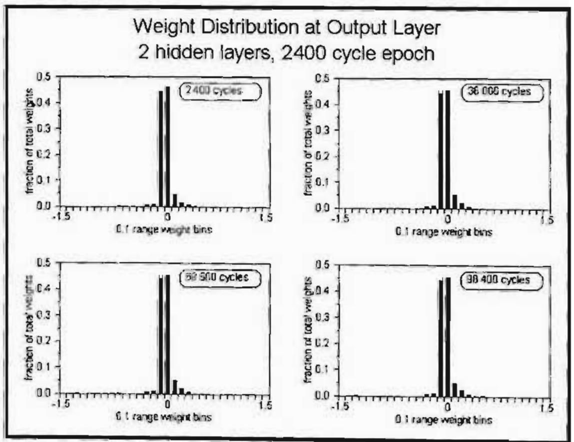


Figure 8.8 Network weight distribution.

Note that there is no increase in the RMS error for the test-set, so the network is not overtrained. There is, in fact, further extremely slow convergence of both the train and test RMS errors as indicated in table (8.3).

Table 8.3 - Extended Network Training, 1 hidden layer & 2400 cycle epoch	
Cycles	**RMS Training Error**
151 769	0.011254
852 189	0.010999

The slowness in convergence (measured per cycle) may be attributable, in part, to the large epoch size. From the discussion of back-propagation theory in sections (3.13) and (3.14) we may also speculate that a bad choice of initial weights led to both unsaturated and incorrectly saturated neurons in the output layer. This would result in the observed initial reduction in the RMS error, followed by network saturation. However, various random initialisations of the network weights failed to produce any convergence improvement. Small perturbations of the weights during training, to encourage the network to escape from any local minima, also failed to improve convergence. The networks are still an order of magnitude above the optimistic convergence criteria set in section (7.5) and there is no likelihood of them achieving that goal within any reasonable timescale. The single hidden layer, 2400 cycle epoch network, trained for 151 769 cycles is selected for testing as a compromise between too little and possibly too much training. The RMS training convergence and weight distribution for this network are shown in the screen dump of figure (8.9).

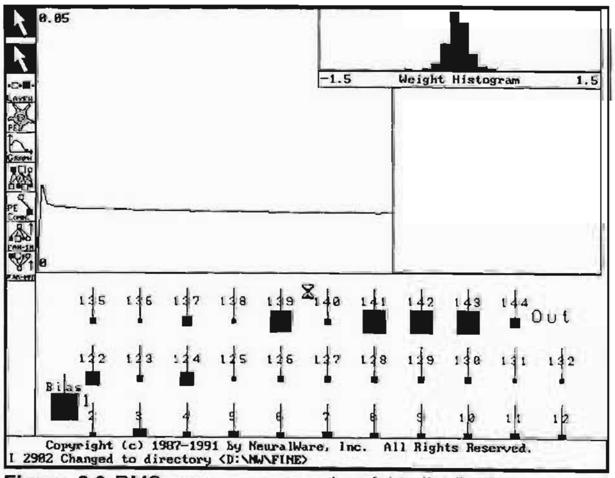


Figure 8.9 RMS convergence and weight distribution.

8.2 Theoretical Test-set Results

Table (8.4) tabulates the detailed results of the 56 member reserved test-set for the final network selected at the end of section (8.1). This is the test-set whose RMS error performance is plotted in the graphs of table (8.1). If the network had achieved convergence to an RMS error of 0.001 as desired in section (7.5), then we would expect errors in position of around 3 microns and errors in density of about 0.0025 atmospheres in table (8.4). The targets are fairly ambitious and we shall show in this section that the higher convergence indicated in table (8.3) is quite adequate.

Table 8.4 - Theoretical Test-set Results

	vacuum		tail		saddle		rear		front		vacuum		tail		saddle		rear		front	
	pos	den	pos	den	pos	den	pos	den	pos	den	pos	den	pos	den	pos	den	pos	den	pos	den
	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm
test	808	0.00	1187	0.49	2534	0.59	2652	2.21	2702	1.00	810	0.00	1176	0.50	2510	0.57	2655	2.20	2708	1.00
target	850	0.00	1250	0.45	2550	0.65	2700	2.25	2750	1.00	850	0.00	1250	0.50	2500	0.60	2650	2.15	2700	1.00
delta	-42	0.00	-63	0.04	-16	-0.06	-48	-0.04	-48	0.00	-40	0.00	-74	0.00	10	-0.03	5	0.05	8	0.00
test	807	0.00	1190	0.47	2526	0.59	2617	2.21	2685	1.00	805	0.01	1177	0.52	2584	0.61	2630	2.20	2699	1.00
target	850	0.00	1250	0.50	2550	0.60	2650	2.25	2700	1.00	850	0.00	1250	0.50	2600	0.65	2650	2.25	2750	1.00
delta	-43	0.00	-60	-0.03	-24	-0.01	-33	-0.04	-15	0.00	-45	0.01	-73	0.02	-16	-0.04	-20	-0.05	-51	0.00
test	807	0.00	1164	0.53	2528	0.58	2622	2.18	2720	1.00	800	0.00	1161	0.53	2538	0.60	2632	2.19	2728	1.00
target	850	0.00	1250	0.55	2500	0.60	2600	2.25	2700	1.00	850	0.00	1250	0.55	2550	0.60	2650	2.25	2750	1.00
delta	-43	0.00	-86	-0.02	28	-0.02	22	-0.07	20	0.00	-50	0.00	-89	-0.02	-12	-0.00	-18	-0.06	-22	0.00
test	793	0.00	1173	0.53	2552	0.63	2665	2.21	2740	1.00	808	0.00	1187	0.46	2506	0.56	2619	2.20	2675	1.00
target	850	0.00	1250	0.55	2600	0.65	2700	2.15	2750	1.00	850	0.00	1200	0.45	2550	0.55	2650	2.20	2700	1.00
delta	-57	0.00	-77	-0.02	-48	-0.02	-35	0.06	-10	0.00	-42	0.00	-13	0.01	-44	0.01	-31	0.00	-25	0.00
test	808	0.00	1189	0.49	2522	0.60	2626	2.20	2683	1.00	804	0.00	1170	0.50	2537	0.61	2638	2.20	2731	1.00
target	850	0.00	1200	0.50	2500	0.55	2600	2.15	2650	1.00	850	0.00	1200	0.50	2550	0.60	2650	2.25	2750	1.00
delta	-42	0.00	-11	-0.01	22	0.05	26	0.05	33	0.00	-46	0.00	-30	0.00	-13	0.01	-12	-0.05	-19	-0.00
test	804	0.00	1183	0.47	2519	0.59	2635	2.20	2696	1.00	789	0.00	1174	0.53	2544	0.63	2668	2.22	2742	1.00
target	850	0.00	1200	0.55	2500	0.60	2600	2.25	2650	1.00	850	0.00	1200	0.55	2550	0.65	2650	2.15	2700	1.00
delta	-46	0.00	-17	-0.08	19	-0.01	35	-0.05	46	0.00	-61	0.00	-26	-0.02	-6	-0.02	18	0.07	42	0.00
test	798	0.00	1176	0.48	2535	0.61	2630	2.21	2719	1.00	810	0.00	1181	0.51	2525	0.60	2655	2.20	2707	1.00
target	750	0.00	1150	0.45	2500	0.60	2600	2.25	2700	1.00	750	0.00	1150	0.45	2550	0.65	2700	2.15	2750	1.00
delta	48	0.00	26	0.03	35	0.01	30	-0.04	19	-0.00	60	0.00	31	0.06	-25	-0.05	-45	0.05	-43	0.00
test	810	0.00	1176	0.52	2524	0.61	2659	2.20	2719	1.00	810	0.00	1153	0.54	2561	0.57	2662	2.16	2718	1.00
target	750	0.00	1150	0.50	2500	0.65	2650	2.25	2700	1.00	750	0.00	1150	0.50	2600	0.60	2700	2.20	2750	1.00
delta	60	0.00	26	0.02	24	-0.04	9	-0.05	19	0.00	60	0.00	3	0.04	-39	-0.03	-38	-0.04	-32	-0.00
test	800	0.00	1168	0.52	2535	0.59	2634	2.20	2729	1.00	795	0.00	1185	0.49	2524	0.60	2658	2.21	2740	1.00
target	750	0.00	1150	0.55	2550	0.60	2650	2.15	2750	1.00	750	0.00	1200	0.45	2500	0.55	2650	2.25	2750	1.00
delta	50	0.00	18	-0.03	-15	-0.01	-16	0.05	-21	-0.00	45	0.00	-15	0.04	24	0.05	8	-0.04	-10	-0.00
test	807	0.00	1181	0.49	2528	0.58	2643	2.20	2703	1.00	802	0.00	1175	0.50	2510	0.59	2637	2.19	2696	1.00
target	750	0.00	1200	0.45	2550	0.65	2650	2.15	2700	1.00	750	0.00	1200	0.50	2500	0.65	2600	2.20	2650	1.00
delta	57	0.00	-19	0.04	-22	-0.07	-7	0.05	3	0.00	52	0.00	-25	-0.00	10	-0.06	37	-0.01	46	0.00
test	813	0.00	1194	0.47	2572	0.59	2621	2.22	2670	1.00	806	0.00	1189	0.50	2578	0.60	2629	2.21	2687	1.00
target	750	0.00	1200	0.50	2600	0.55	2650	2.25	2700	1.00	750	0.00	1200	0.55	2550	0.55	2600	2.20	2650	1.00
delta	63	0.00	-6	-0.03	-28	0.04	-29	-0.03	-30	0.00	56	0.00	-11	-0.05	28	0.05	29	0.01	37	0.00
test	802	0.00	1175	0.52	2568	0.62	2643	2.20	2714	1.00	804	0.00	1183	0.48	2527	0.58	2640	2.20	2696	1.00
target	750	0.00	1200	0.55	2600	0.65	2650	2.15	2700	1.00	800	0.00	1150	0.45	2550	0.55	2700	2.15	2750	1.00
delta	52	0.00	-25	-0.03	-32	-0.03	-7	0.05	14	0.00	4	0.00	33	0.03	-23	0.03	-60	0.05	-54	0.00
test	805	0.00	1168	0.52	2508	0.57	2653	2.19	2725	1.00	801	0.00	1184	0.52	2525	0.63	2653	2.23	2742	1.00
target	800	0.00	1150	0.50	2500	0.55	2650	2.25	2750	1.00	800	0.00	1150	0.50	2550	0.65	2650	2.20	2750	1.00
delta	5	0.00	18	0.02	8	0.02	3	-0.06	-25	-0.00	1	0.00	34	0.02	-25	-0.02	3	0.03	-8	-0.00
test	795	0.00	1176	0.52	2529	0.62	2661	2.20	2722	1.00	805	0.00	1177	0.52	2536	0.61	2643	2.19	2712	1.00
target	800	0.00	1150	0.55	2500	0.65	2600	2.20	2650	1.00	800	0.00	1150	0.55	2550	0.65	2650	2.15	2700	1.00
delta	-5	0.00	26	-0.03	29	-0.03	61	-0.00	72	0.00	5	0.00	27	-0.03	-14	-0.04	-7	0.04	12	0.00
test	800	0.00	1179	0.49	2516	0.59	2638	2.21	2728	1.00	789	0.00	1175	0.49	2532	0.61	2627	2.21	2724	1.00
target	800	0.00	1200	0.45	2500	0.55	2650	2.25	2750	1.00	800	0.00	1200	0.45	2550	0.60	2650	2.15	2750	1.00
delta	-0	0.00	-21	0.04	16	0.04	-12	-0.04	-22	0.00	-11	0.00	-25	0.04	-18	0.01	-23	0.06	-26	-0.00
test	806	0.00	1184	0.51	2581	0.61	2637	2.21	2703	1.00	810	0.00	1186	0.49	2578	0.57	2634	2.22	2727	1.00
target	800	0.00	1200	0.45	2600	0.65	2650	2.20	2750	1.00	800	0.00	1200	0.50	2550	0.55	2600	2.15	2700	1.00
delta	6	0.00	-16	0.06	-19	-0.04	-13	0.01	-47	0.00	10	0.00	-14	-0.01	28	0.02	34	0.07	27	-0.00
test	807	0.00	1173	0.54	2591	0.60	2631	2.20	2682	1.00	806	0.00	1182	0.48	2519	0.59	2637	2.20	2699	1.00
target	800	0.00	1200	0.50	2600	0.55	2650	2.15	2700	1.00	800	0.00	1200	0.55	2500	0.60	2600	2.25	2650	1.00
delta	7	0.00	-27	0.04	-9	0.05	-19	0.05	-18	0.00	6	0.00	-18	-0.07	19	-0.01	37	-0.05	49	0.00
test	802	0.00	1179	0.51	2550	0.61	2650	2.21	2715	1.00	801	0.00	1173	0.52	2539	0.61	2641	2.19	2716	1.00
target	800	0.00	1200	0.55	2550	0.60	2650	2.25	2700	1.00	800	0.00	1200	0.55	2600	0.65	2700	2.15	2750	1.00
delta	2	0.00	-21	-0.04	0	0.01	0	-0.04	15	0.00	1	0.00	-27	-0.03	-61	-0.04	-59	0.04	-34	0.00
test	801	0.00	1163	0.50	2518	0.56	2626	2.18	2717	1.00	811	0.00	1192	0.47	2577	0.58	2631	2.22	2680	1.00
target	800	0.00	1250	0.45	2550	0.55	2650	2.15	2750	1.00	800	0.00	1250	0.45	2600	0.65	2650	2.20	2700	1.00
delta	1	0.00	-87	0.05	-32	0.01	-24	0.03	-33	-0.00	11	0.00	-58	0.02	-23	-0.07	-19	0.02	-20	0.00
test	807	0.00	1177	0.49	2536	0.59	2648	2.20	2707	1.00	809	0.00	1186	0.50	2517	0.60	2656	2.20	2705	1.00
target	800	0.00	1250	0.50	2550	0.60	2650	2.15	2700	1.00	800	0.00	1250	0.55	2500	0.55	2650	2.20	2700	1.00
delta	7	0.00	-73	-0.01	-14	-0.01	-2	0.05	7	0.00	9	0.00	-64	-0.05	17	0.05	6	-0.00	5	0.00

Table 8.4 - Theoretical Test-set Results

	vacuum		tail		saddle		rear		front		vacuum		tail		saddle		rear		front	
	pos	den	pos	den	pos	den	pos	den	pos	den	pos	den	pos	den	pos	den	pos	den	pos	den
	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm	µm	atm
test	812	0.01	1182	0.52	2550	0.63	2616	2.20	2707	1.00	813	0.00	1181	0.48	2529	0.58	2647	2.20	2695	1.00
target	800	0.00	1250	0.55	2550	0.65	2600	2.15	2700	1.00	850	0.00	1150	0.45	2500	0.60	2650	2.25	2700	1.00
delta	12	0.01	-68	-0.03	0	-0.02	16	0.05	7	0.00	-37	0.00	31	0.03	29	-0.02	-3	-0.05	-5	0.00
test	814	0.00	1182	0.48	2568	0.56	2604	2.20	2700	1.00	808	0.00	1187	0.48	2593	0.56	2626	2.23	2716	1.00
target	850	0.00	1150	0.45	2600	0.55	2650	2.20	2750	1.00	850	0.00	1150	0.50	2550	0.55	2600	2.25	2700	1.00
delta	-36	0.00	32	0.03	-32	0.01	-46	-0.00	-50	0.00	-42	0.00	37	-0.02	43	0.01	26	-0.02	16	-0.00
test	805	0.00	1176	0.50	2537	0.57	2632	2.20	2727	1.00	795	0.00	1172	0.46	2530	0.59	2636	2.21	2715	1.00
target	850	0.00	1150	0.55	2500	0.55	2600	2.20	2700	1.00	750	0.00	1200	0.45	2500	0.60	2600	2.15	2700	1.00
delta	-45	0.00	26	-0.05	37	0.02	32	-0.00	27	0.00	45	0.00	-28	0.01	30	-0.01	36	0.06	15	-0.00
test	800	0.00	1190	0.47	2569	0.58	2673	2.22	2719	1.00	801	0.00	1177	0.52	2531	0.62	2645	2.21	2734	1.00
target	800	0.00	1150	0.45	2550	0.60	2650	2.15	2700	1.00	750	0.00	1150	0.50	2500	0.60	2650	2.20	2750	1.00
delta	-0	0.00	40	0.02	19	-0.02	23	0.07	19	-0.00	51	0.00	27	0.02	31	0.02	-5	0.01	-16	0.00
test	812	0.01	1176	0.51	2588	0.58	2616	2.19	2714	1.00	802	0.00	1187	0.48	2568	0.59	2667	2.22	2710	1.00
target	800	0.00	1250	0.50	2600	0.55	2650	2.15	2750	1.00	850	0.00	1150	0.45	2500	0.55	2650	2.20	2700	1.00
delta	12	0.01	-74	0.01	-12	0.03	-34	0.04	-36	0.00	-48	0.00	37	0.03	68	0.04	17	0.02	10	-0.00
test	803	0.00	1183	0.51	2566	0.61	2660	2.21	2723	1.00	804	0.00	1181	0.49	2542	0.61	2643	2.21	2724	1.00
target	750	0.00	1150	0.50	2600	0.65	2700	2.15	2750	1.00	850	0.00	1200	0.45	2500	0.65	2600	2.25	2700	1.00
delta	53	0.00	33	0.01	-34	-0.04	-40	0.06	-27	-0.00	-46	0.00	-19	0.04	42	-0.04	43	-0.04	24	-0.00
test	795	0.00	1177	0.50	2586	0.59	2665	2.22	2721	1.00	806	0.00	1192	0.45	2546	0.56	2630	2.22	2704	1.00
target	850	0.00	1200	0.50	2600	0.55	2700	2.20	2750	1.00	750	0.00	1200	0.45	2500	0.55	2600	2.15	2700	1.00
delta	-55	0.00	-23	0.00	-14	0.04	-35	0.02	-29	-0.00	56	0.00	-8	0.00	46	0.01	30	0.07	4	-0.00
test	800	0.00	1177	0.50	2564	0.59	2661	2.22	2723	1.00	805	0.00	1182	0.49	2532	0.60	2646	2.20	2702	1.00
target	750	0.00	1200	0.50	2550	0.55	2700	2.15	2750	1.00	800	0.00	1200	0.50	2550	0.60	2650	2.20	2700	1.00
delta	50	0.00	-23	-0.00	14	0.04	-39	0.07	-27	0.00	5	0.00	-18	-0.01	-18	-0.00	-4	-0.00	2	0.00

Some elementary analysis of the figures in table (8.4) reveals the following:

Table 8.5 - Theoretical Test-set Error Analysis

Statistic	Minimum Error		Maximum Error		Mean Error		Standard Deviation	
Shock Region	Position (µm)	Density (atm)	Position (µm)	Density (atm)	Position (µm)	Density (atm)	Position (µm)	Density (atm)
Vacuum	0	0.00	63	0.01	33	0.00	21.942	0.002
Tail	3	0.00	89	0.08	35	0.03	22.755	0.018
Saddle	0	0.00	68	0.07	24	0.03	13.438	0.018
Rear	0	0.00	61	0.07	24	0.04	15.637	0.021
Front	2	0.00	72	0.00	25	0.00	15.353	0.000
TOTAL	0	0.00	89	0.08	28	0.02	18.774	0.022

It is interesting to compare the total mean error of the test-set with the errors expected from the RMS_{test} error values. With an RMS_{test} of 0.011254 from table (8.3) we would expect errors in position of around 34 microns and errors in density of about 0.03 atmospheres. The actual mean error performance is slightly better than this at 28 microns and 0.02 atmospheres respectively. Note however that the application of the RMS error measure is complicated by the presence of two types of neurons in the output layer - those encoding position and those encoding density. The calculation of a single RMS error does not distinguish between these neurons, even though section (4.2) indicates that there is a difference of three orders of magnitude in their respective range of values.

Recall from section (4.2) that the fine grid superimposed on the base density profile in order to generate both the test and train sets by permutation, has a resolution of 50 microns and 0.05 atmospheres. The mean network error tabulated in table (8.5) is within this granularity and is therefore deemed acceptable. Figures (8.10) and (8.11) are graphical representations of the position and density error distributions tabulated in table (8.4).

Table 8.6 - Test-set Position and Density Error Distribution

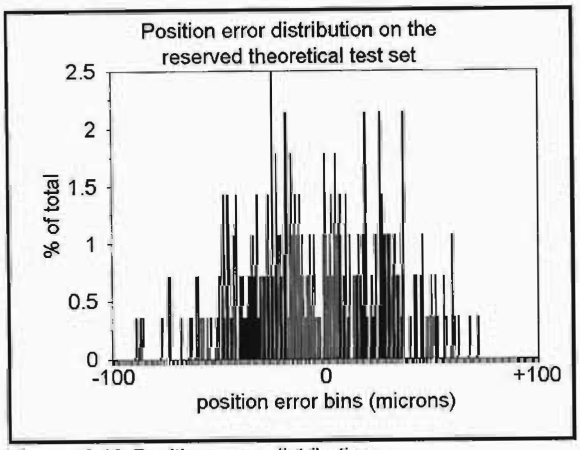


Figure 8.10 Position error distribution.

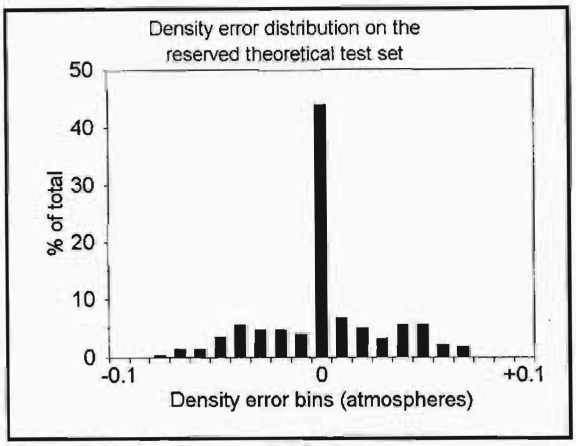


Figure 8.11 Density error distribution.

8.3 Experimental Test-set Results

Recall again from section (4.2) that the perturbations applied to the theoretical base density profile to generate the training and testing sets are relatively minor. This means that the network is trained exclusively on theoretical data generated for a shock at a time around 4.0 μs after the discharge. Training a single network to cover a large time scale would be difficult as the theoretical intensity patterns are so sensitive to the selected density profile. Since the shock front propagates so rapidly, the network should only be tested at times around the 4.0 μs mark. Nonetheless, it shall be interesting to see how the network attempts to accommodate shocks at different times so we shall test experimental images captured from 1.6 to 12.0 μs after the discharge. Intensity fringes are extracted from the images according to the procedure in section (5.7) and these experimental intensity patterns are fed to the network.

Table (8.7) shows the intensity patterns extracted from the captured images, as well the density profiles that the network has determined underlies the shocks giving rise to the intensity pattern. Note that the first few peaks and troughs in the experimental images presented in tables (8.7) and (8.8) may be almost entirely due to diffraction effects around and between the electrodes. The linear structures surrounding the electrodes in the images in table (5.4) are Fresnel diffraction fringes. The ray-tracer computes only fringes due to refraction and interference and does not reproduce the Fresnel fringes.

Table 8.7 - Experimental Intensity Patterns and their Postulated Density Profiles

Experimental Intensity Pattern	Network Postulated Density Profile

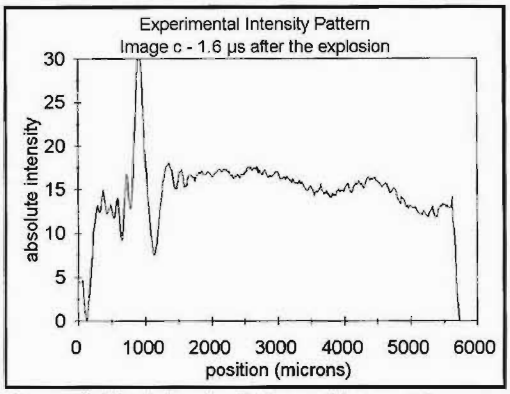


Figure 8.12.a Intensity Pattern of Image (c).

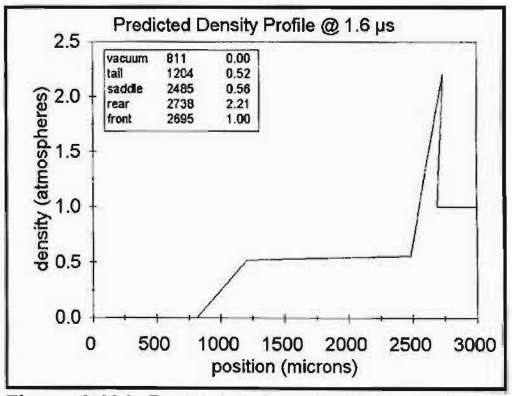


Figure 8.12.b Postulated Density Profile for Image (c).

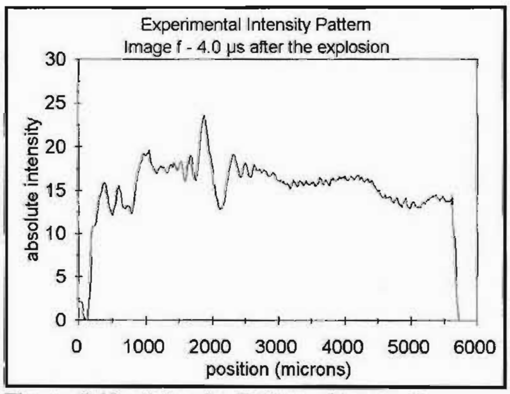


Figure 8.13.a Intensity Pattern of Image (f).

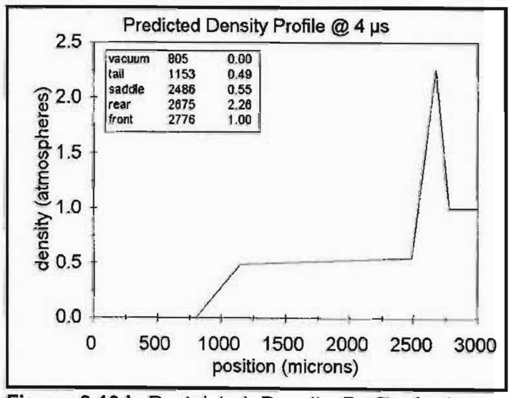


Figure 8.13.b Postulated Density Profile for Image (f).

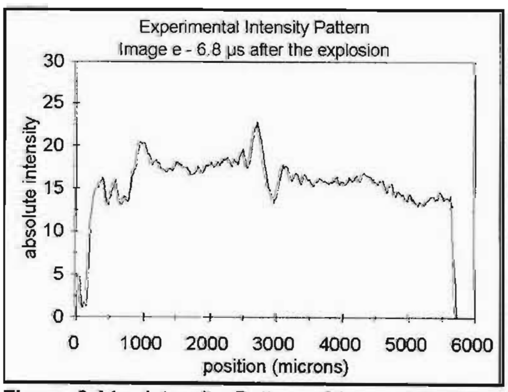


Figure 8.14.a Intensity Pattern of Image (e).

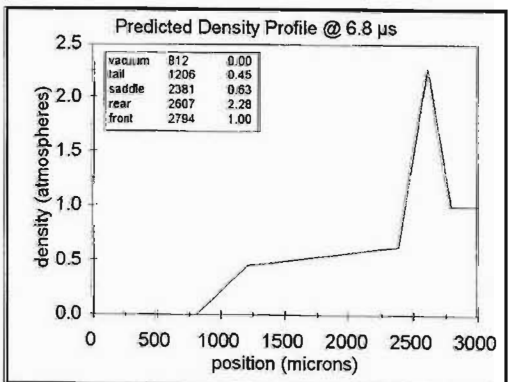
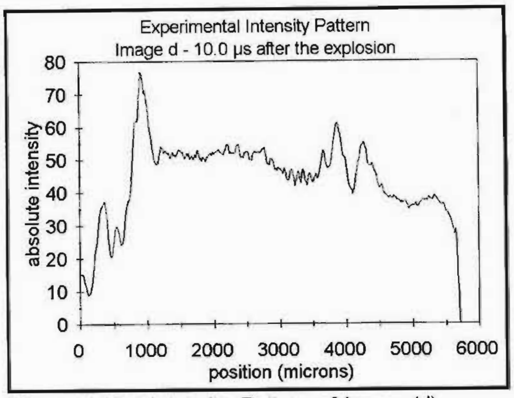
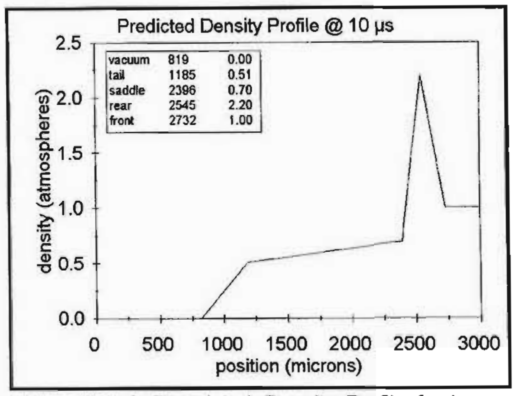
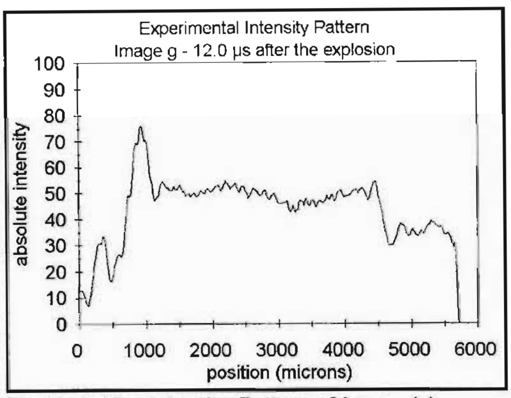
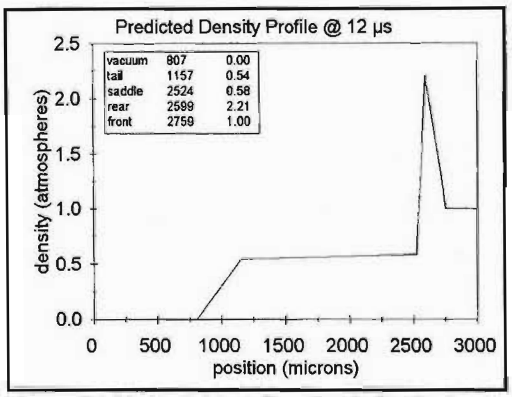


Figure 8.14.b Postulated Density Profile for Image (e).

Table 8.7 - Experimental Intensity Patterns and their Postulated Density Profiles

Experimental Intensity Pattern	Network Postulated Density Profile
 **Figure 8.15.a** Intensity Pattern of Image (d).	 **Figure 8.15.b** Postulated Density Profile for Image (d).
 **Figure 5.16.a** Intensity Pattern of Image (g).	 **Figure 8.16.b** Postulated Density Profile for Image (g).

As expected, the positions of the shock density regions do not stray too far from those in the training set - they certainly do not move far enough to model the physical situation. Compare these density profiles with the adapted Sedov theoretical curves in figure (2.1), but remember that the Sedov curves are also theoretical and do not define the density profile underlying the shock exactly. The network is so confused for the very early pattern 1.6 μs after the discharge that the shock peak is predicted to occur before the shock front. It is also disappointing that the peak of the density profile does not drop over time as indicated in figure (2.1). The gradients of the shock features are more plausible. Note how the saddle region is almost flat for early (1.6 μs) and late (12.0 μs) times (weak or no shock) and how the gradient of the shock rear changes. For the main pattern (image (f) at 4.0 μs) the density profile predicted by the network has a steeper shock front and rear than the corresponding modified Sedov curve in figure (2.1), but the saddle density and tail gradient are similar to those in the modified Sedov curve.

Poor performance for times on either side of the 4.0 μs timemark is entirely expected as we are attempting to extrapolate the network's performance well outside its trained ranges. Generalisation usually implies interpolation. The generalisation performance we seek is generalisation from a set of theoretical interference patterns around 4.0 μs to an experimental one at the same time. To test this we feed the density profile determined by the network back to the mathematical ray-tracer to produce theoretical intensity patterns. Table (8.8) shows the experimental intensity patterns next to those determined by the ray-tracer. Remember that image (f), captured 4.0 μs after the discharge, is the main test.

Table 8.8 - Experimental and Theoretical Intensity Patterns

Experimental Intensity Pattern	Theoretical Intensity Pattern

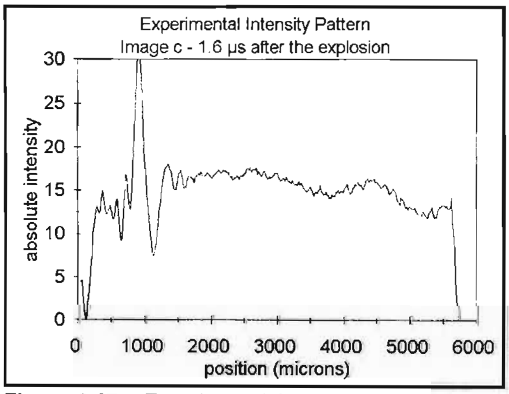


Figure 8.17.a Experimental Intensity Pattern Image (c).

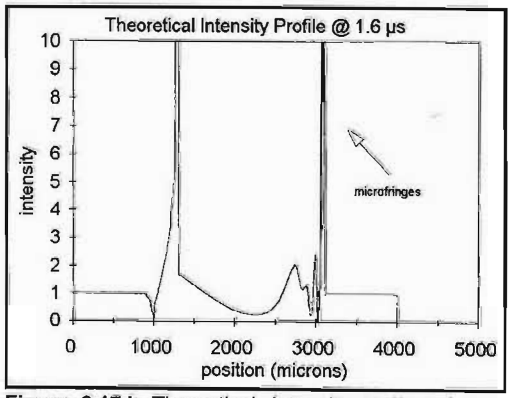


Figure 8.17.b Theoretical Intensity Pattern Image (c).

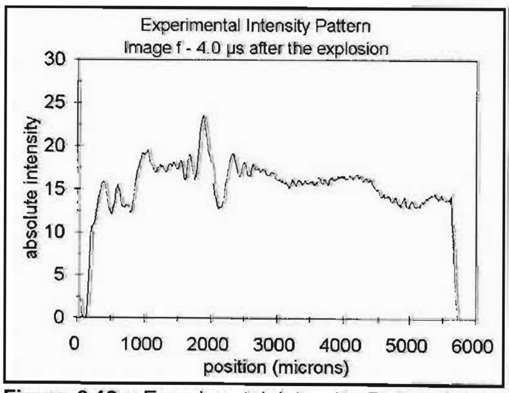


Figure 8.18.a Experimental Intensity Pattern Image (f).

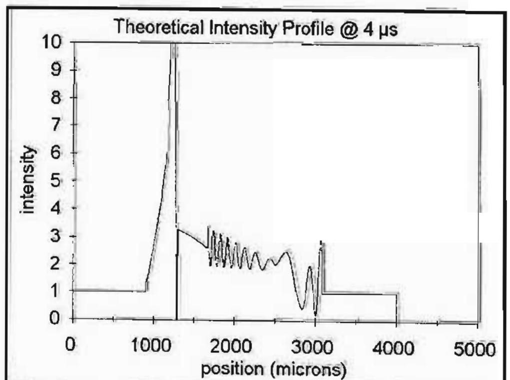
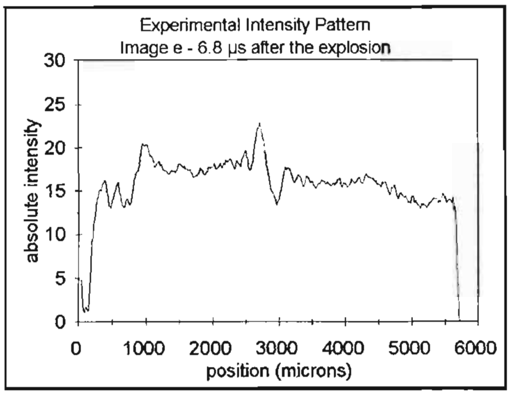
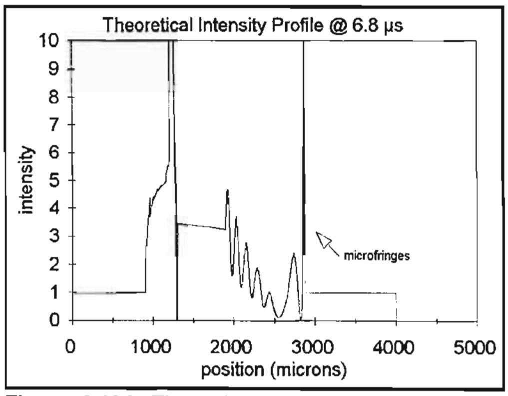
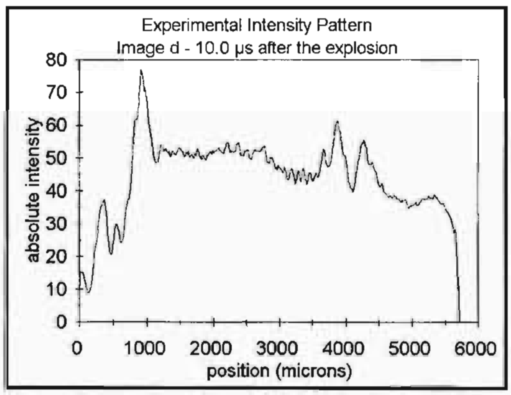
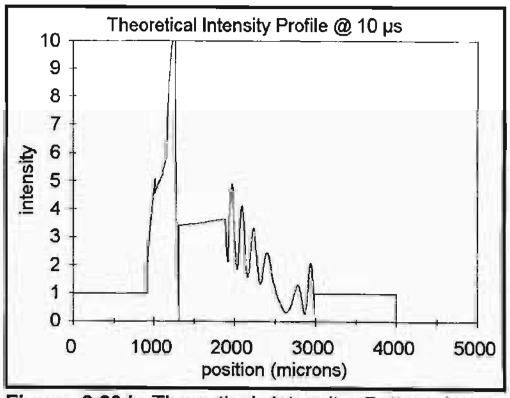
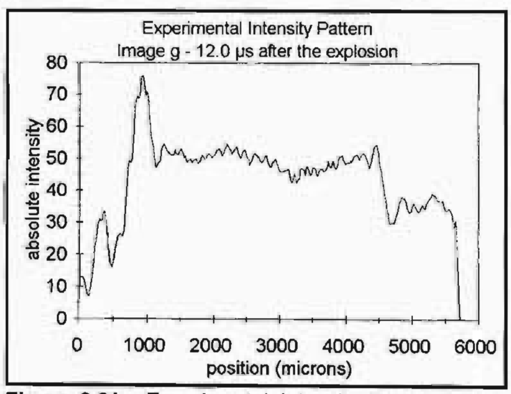
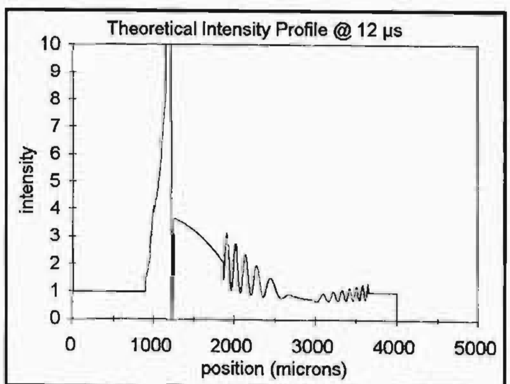


Figure 8.18.b Theoretical Intensity Pattern Image (f).

Table 8.8 - Experimental and Theoretical Intensity Patterns

Experimental Intensity Pattern	Theoretical Intensity Pattern
 **Figure 8.19.a** Experimental Intensity Pattern Image (e).	 **Figure 8.19.b** Theoretical Intensity Pattern Image (e).
 **Figure 8.20.a** Experimental Intensity Pattern Image (d).	 **Figure 8.20.b** Theoretical Intensity Pattern Image (d).
 **Figure 8.21.a** Experimental Intensity Pattern Image (g).	 **Figure 8.21.b** Theoretical Intensity Pattern Image (g).

We can make similar remarks about the dangers of extrapolation for table (8.8) as well. There is obvious confusion for the early shock at 1.6 μs with spurious strong peaks and microfringes at 3000 microns. The peaks for the main pattern at 4.0 μs have spread out about 1000 microns more than they should have, but are otherwise reasonable in relative distribution and relative (not absolute) intensity. The spreading may be due to incorrect scaling of the image. A more detailed analysis of this pattern will be given later. At 6.8 μs the locations of the peaks are more plausible, but their relative intensities are incorrect as they drop towards zero. Like the 1.0 μs situation, this pattern ends in microfringes leading up to a strong peak. The situation is similar at 10.0 μs, but the microfringes and strong peaks have disappeared. The last two peaks and troughs correspond well to the experimental pattern, although they are about 1000 microns too far in. These final extrema disappear in the experimental pattern for the situation at 12.0 μs and are absent in the theoretical pattern as well. Indeed the network has tracked supposed extrema (actually considerable noise) between 3000 and 4000 microns quite well in both position and relative intensity. The peaks in this region could be the 'microfringes' of the 6.8 μs situation 'spread out'. These latter results are surprisingly good for such a late time.

The microfringes exhibited by figures (8.17.b) and (8.19.b) are more easily observable in the expanded figures (8.22) and (8.23). These are most probably purely computational features with no physical significance.

Table 8.9 - Microfringes in the Theoretical Intensity Patterns

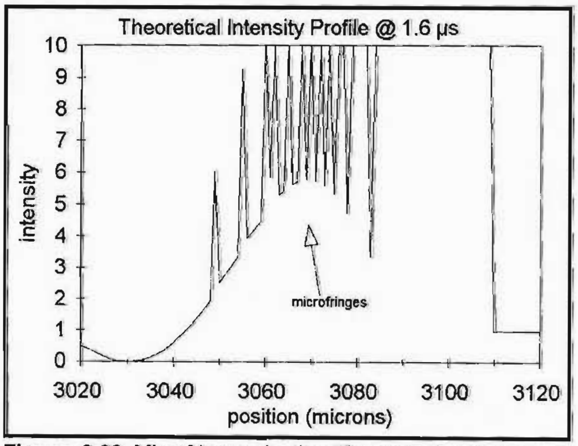


Figure 8.22 Microfringes in the Theoretical Interference Pattern for Image c.

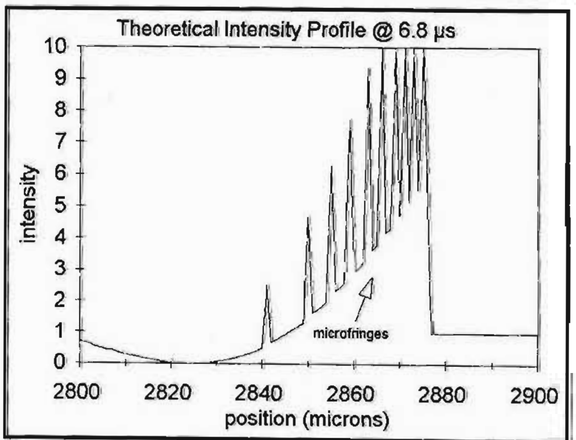


Figure 8.23 Microfringes in the Theoretical Interference Pattern for Image e.

Finally, in table (8.10), we compare our results for a time 4.0 μs after the discharge to that originally published by the Plasma Physics Research Institute for the same time.

Table 8.10 - Comparison with Original Published Results

Experimental Intensity Pattern	**Theoretical Intensity Pattern**

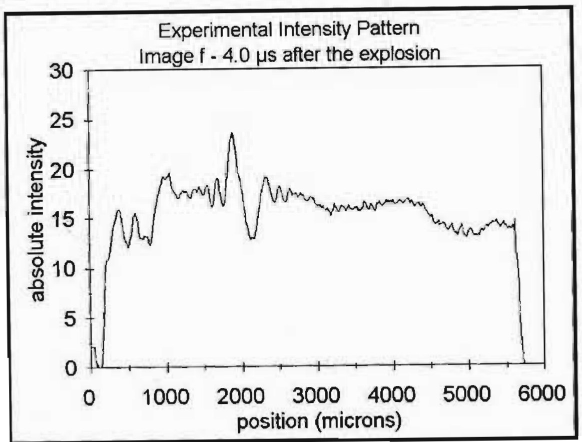


Figure 8.24.a Experimental Intensity Pattern Image (f).

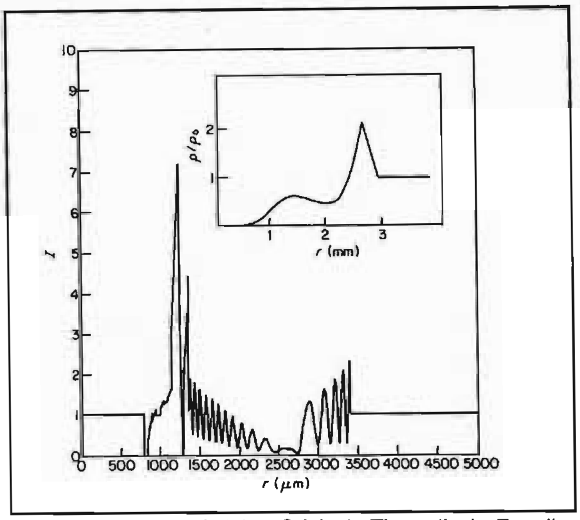


Figure 8.24.b Institute's Original Theoretical Results. Reproduced from Michaelis et al. [Mic91].

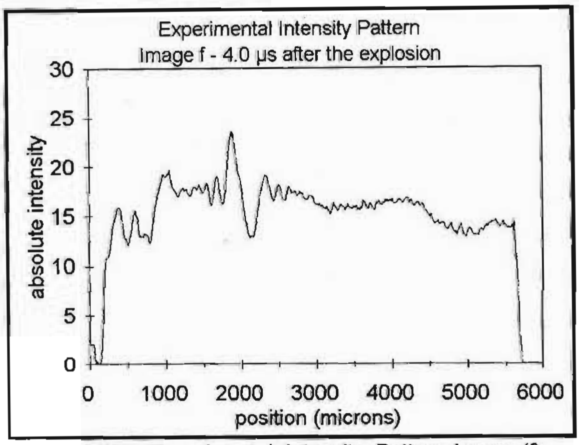


Figure 8.25.a Experimental Intensity Pattern Image (f).

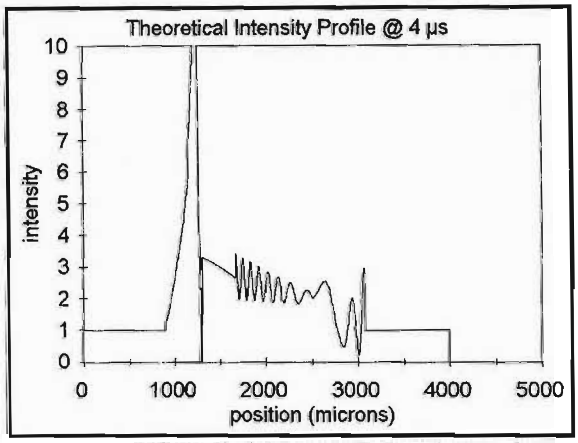


Figure 8.25.b Theoretical Intensity Pattern Image (f).

The number and *relative* size of the peaks and troughs in the new results are more accurate than the original theoretical results, although the positions of the peaks and troughs in the original theoretical results have a slightly better correspondence to the experimental results than do the theoretical results of the network/ray-tracer pair. Although the absolute intensities of the peaks and troughs are still erroneous for both sets of theoretical results, the network/ray-tracer pair has produced a far more accurate representation of the intensities. The central intensity peaks and troughs of the new results are located much higher than before and the peaks and troughs of the central region are no longer so artificially superimposed and damped on a slope that goes to zero. Note how the network/ray-tracer pair latches onto the prominent outermost two peaks and troughs in all the theoretical results. The outer fringes are predicted more accurately by the network/ray-tracer pair than in the original results. Other prominent features, such as the bright rings, dark ring and grey ring identified in section (2.8) have also been reproduced.

8.4 Conclusions

The shock was tracked over a large time frame to follow its evolution. The network is not able to accurately extrapolate its theoretical predictions to the early or late stages of the shock, but it does make a reasonable attempt. Large temporal variations in the evolving density profiles and intensity patterns mean that the network must be trained and tested on small time windows. Extrapolation is however not our main aim, instead we seek generalisation, or interpolation from a theoretically generated data-set for a particular time, to an experimentally captured pattern at that same time. The results for this latter exercise are an improvement over the theoretical results originally published by the Plasma Physics Research Institute. They are also a reasonable approximation of the actual experimental results given the limitations of the experimental equipment, in particular the poor spatial coherence of the illuminating laser - (section 5.7), the relative mathematical simplicity of the computational ray-tracer (section 1.2) and the limited (though adequate) convergence of the neural network (sections 8.1 and 8.2).

Neural networks do expand the range of computational tools available to research scientists. Objective consideration of the experiences of chapter 8 suggest that, for the application tackled by this thesis, any time saved by letting the neural network do the pattern association instead of the scientist must be balanced against the time taken to construct, tune and train the neural network. Once trained however, no modifications should be necessary to process similar patterns. In must be noted that ultimately all the neural network does is optimise the performance of the mathematical ray-tracer by selecting more plausible input patterns. The performance of the network/ray-tracer pair is bounded by the accuracy of the ray-tracer.

The Plasma Physics Research Institute has since investigated two colliding shocks, (figure 8.26), and a circular arrangement of 8 colliding shocks which produce a colliding shock lens capable of focusing intense laser light. Industrial applications for their research have already been presented in the introduction to this thesis.

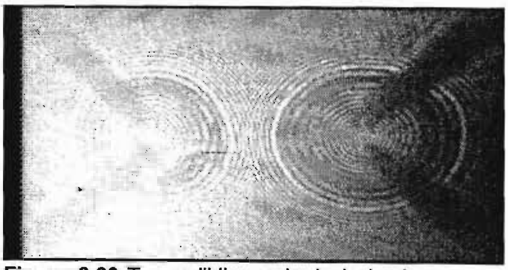


Figure 8.26 Two colliding spherical shocks.

9 References

[Ale90] Aleksander I. & Morton H.
An Introduction To Neural Computing, 1'st edition, 68.
Chappman and Hill, London, 1990.

[Alm89] Almeida L.B.
"Backpropagation in non-feedforward networks", in
Neural Computing Architectures - The Design of Brain-Like Machines, (1989) 74, edited by Aleksander I., North Oxford Academic Press, London, 1989.

[And83] Anderson J.A.
"Cognitive and Psychological Computation with Neural Models", in
IEEE Transactions on Systems, Man and Cybernetics 13 (5) (Sept-Oct 1983) 799-815.

[Ara96] Arabshahi P., Choi J.J., Marks R.J. & Caudell T.P.
"Fuzzy Parameter Adaption in Optimization: Some Neural Net Training Examples", in
IEEE Computational Science and Engineering 3 (1) (Spring 1996) 36.

[Bar89] Barto A.
"From Chemotaxis to Cooperativity: Abstract Exercises in Neuronal Learning Strategies", in
The Computing Neuron (1989) 73, edited by Durbin R., Miall C. and Mitchison G., Addison-Wesley, New York, 1989.

[Bar92] Barnes N.
"An Opto-Electronic Neural Network Processor", in
Neural Networks for Vision, Speech and Language, (1992) 331-347, edited by Linggard R., Meyer D.J. and Nightengale C., Chapman and Hall, London, 1992.

[Bha83] Bhagwandin N.
M.Sc. Thesis: Laser Diagnostics of Plasmas and Flames, 87.
University of Natal, Durban, 1983.

[Bof90] Bofill P., Manyer J., Milla'n J.R. & Salvans V.
"A Systolic Algorithm for Back-Propagation: Mapping onto a Transputer Network", (1990) in
Symbols versus Neurons?, edited by Stender J. & Addis T, IOS Press, Amsterdam, 1990.

[Bor65] Born M. & Wolf E.
Principles of Optics, 3rd edition
Pergamon Press, London, 1965.

[Buc93] Buccellato R., Lisi N. & Michaelis M.M.
"Colliding shock lens", in
Optics Communications 101 (1993) 350-355.

[Buc94] Buccellato R.
Ph.D. Thesis: Refractive Effects in Phase Objects and Associated Phenomena, 122-135, 145.
University of Natal, Durban (1994).

[But93] Buturovic' L., Markovic' M.Z. & Citkusev L.T.
"Unsupervised Backpropagation for Pattern Classification", in
Proceedings: *World Congress on Neural Networks* II (1993) 439.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Car93] Carpenter G.A. & Tan A.H.
"Rule Extraction, Fuzzy ARTMAP and Medical Databases", in
Proceedings: *World Congress on Neural Networks* I (1993) 501.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Car95] Carpenter W.C. & Hoffman M.E.
"Training Backprop Neural Networks", in
AI Expert Magazine (March 1995) 30.

[Cas92a] Casasent D.
"Hybrid Neural Networks and Algorithms", in
Neural Networks for Perception Computation, Learning and Architectures, 2 (1992) 253, edited by Wechsler H., Academic Press, Boston 1992.

[Cas92b] Casasent D.
"Optics and Neural Nets", in
Neural Networks for Vision and Image Processing, (1992) 437-488, edited by Carpenter G.A. & Grossberg S., MIT Press, Cambridge (1992).

[Cau91] Caudill M.
"Evolutionary Neural Networks", in
AI Expert Magazine (March 1991) 28.

[Cec92] Ceccarelli M., DiSarno E., Iovine E. & Maffeo G.
"SEANET: A connectionist Expert System for Fault-Alarm Classification", in
Proceedings: *Fifth Italian Workshop* 5 (1992) 343, published as
Neural Nets WIRN VIETRI-92 Fifth Italian Workshop, edited by Caianiello E.R., World Publishing Company, Singapore 1992.

[Che93] Chen C.H. & Lai H.
"A Comparative Study of the Gradient Descent and the Conjugate Gradient Backpropagation Neural Networks", in
Proceedings: *World Congress on Neural Networks* III (1993) 401.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Coc92] Coccorese E., Martone R. & Morabito C.
"On Line Plasma Shape Identification in a Tokamak reactor via neural network", in
Proceedings: *Fifth Italian Workshop* 5 (1992) 349, published as
Neural Nets WIRN VIETRI-92 Fifth Italian Workshop, edited by Caianiello E.R.,
World Publishing Company, Singapore 1992.

[Cyb96] Cybenko G.
"Neural Networks in Computational Science and Engineering", in
IEEE Computational Science and Engineering 3 (1) (Spring 1996) 36.

[Day90] Dayhoff E.J.
Neural Network Architectures An Introduction, 1'st edition, 58-81.
Van Nostrand Reinhold, New York, 1990.

[Day93] Dayhoff E.J.
"Temporal and Molecular Structures in Neural Systems", in
Proceedings: *World Congress on Neural Networks*, I (1993) 105.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Dec93] Deco G., Finnoff W. & Zimmermann H.G.
"Unsupervised Elimination of Overtraining by a Mutual Information Criterion", in
Proceedings: *World Congress on Neural Networks* III (1993) 420.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Dur89] Durbin R.
"On the Correspondence Between Network Models and the Nervous System", in
The Computing Neuron (1989) 1, edited by Durbin R., Miall C. and Mitchison G.,
Addison-Wesley, New York 1989.

[Ekl94] Eklund P.
"Network Size vs Preprocessing", in
Fuzzy Sets, Neural Networks and Soft Computing, 250-264, edited by Yager R.R. & Zadeh L.A., Van Nostrand Reinhold, New York (1994).

[Erd93] Erdogan S.S. & Hong T.H.
"Massively Parallel Computation of Back-Propagation Algorithm using the Reconfigurable Machine", in
Proceedings: *World Congress on Neural Networks* IV (1993) 861.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Fel87] Feldman J.A.
"Structured Neural Networks in Nature and in Computer Science", in
Proceedings: *NATO Advanced Research Workshop on Neural Computers*, (1987) 17, published as *Neural Computers*, edited by Eckmiller R. & Malsburg C., Springer-Verlag, Berlin, 1987.

[Fog90] Fogarty T.C.
"Using Genetic Algorithms to adapt Intelligent Systems", in
Symbols versus Neurons, edited by Stender J. & Addis T., IOS Press, Amsterdam (1990).

[Fun95] Funabashi M., Maeda A., Morooka Y., Mori K. & Works O.
"Fuzzy and Neural Hybrid Expert Systems: Synergetic AI", in
IEEE Expert 10 (4) (August 1995) 32.

[Gar93] Gardner D.
"Back-Propagation and Neuromorphic Plausibility", in
Proceedings: *World Congress on Neural Networks* II (1993) 590.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Gor90] Gori M. & Tesi A.
"Some Examples of Local Minima during Learning with Back Propagation", in
Proceedings: *Third Italian Workshop on Parallel Architectures and Neural Networks* 3 (1990) 87, edited by Caianiello E.R., World Scientific, Singapore 1990.

[Gua90] Guazzo G.M., Rampone S. & Tagliaferri R.
"'State-Dependent' Learning Processes in Neural Nets", in
Proceedings: *Third Italian Workshop on Parallel Architectures and Neural Networks* 3 (1990) 95, edited by Caianiello E.R., World Scientific, Singapore 1990.

[Ham93] Hammerstrom D.
"Working with Neural Networks", in
IEEE Spectrum (July 1993) 46-53.

[Has94] Hassibi B., Sayed A.H. & Kailath T.
"H^∞ Optimality Criteria for LMS and Backpropagation", in
Advances in Neural Information Processing Systems 6 (1994) 351, edited by Cowan J.D., Tesauro G. and Alspector J., Morgan Kaufmann Publishers, San Francisco 1994.

[Hay94] Haykin S.
Neural Networks A Comprehensive Foundation, 1'st edition, 138-220, 585-589.
MacMillian College Publishing Company, New York (1994).

[Hec88] Hecht-Nielsen R.
"Neurocomputing: picking the human brain", in
IEEE Spectrum, 25 (3) (March 1988) 36-41.

[Hec92] Hecht-Nielson R.
"Theory of the Backpropagation Neural Network", in
Neural Networks for Perception Computation, Learning and Architectures, 2 (1992) 65, edited by Wechsler H., Academic Press, Boston, 1992.

[Hen91] Hendler J.
"Developing Hybrid Symbolic / Connectionist Models", in
Advances in Connectionist and Neural Computation Theory : High Level Connectionist Models, 1 (1991) 165, edited by Barden J.A. & Pollack J.B., Ablex Publishing Corporation, Norwood NJ, 1991.

[Her91] Hertz J., Krogh A. & Palmer R.G.
Introduction to the Theory of Neural Computation, 115-130.
Addison-Wesley, Massachusetts, 1991.

[Hop88] Hopfield J.J.
"Artificial Neural Networks", in
IEEE Circuits and Devices Magazine (Sept 1988) 3-10.

[Hu93] Hu Q. & Hertz D.B.
"Improving the Convergence of Backpropagation Using Fuzzy Logic Control", in
Proceedings: *World Congress on Neural Networks* II (1993) 47.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Jan93] Janakiraman J. & Honavar V.
"Adaptive Learning Rate for increasing learning speed in Backpropagation networks", in
Proceedings: *World Congress on Neural Networks* IV (1993) 378.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Jeo92] Jeon H., Shamir J., Johnson R.B., Caulfield H.J., Kinser J., Hester C. & Temmen M.
"The Use of Fixed Holograms for Massively-Interconnected, Low-Power Neural Networks", in
Neural Networks for Perception Computation, Learning and Architectures 2 (1992) 282, edited by Wechsler H., Academic Press, Boston 1992.

[Kam93] Kamiyama N., Taguchi A., Watanabe H., Pechranin N., Yoshida Y & Sone M.
"Tuning of Learning Rate and Momentum in Back-Propagation", in
Proceedings: *World Congress on Neural Networks* IV (1993) 119.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Kel93] Keller P.E. & Gmitro A.F.
"Operational Parameters of an Opto-Electronic Neural Network Employing Fixed Planar Holographic Interconnects", in
Proceedings: *World Congress on Neural Networks* IV (1993) 799.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Kem94a] Kempka A.A.
"Activating Neural Networks: Part 1", in
AI Expert Magazine (June 1994) 33.

[Kem94b] Kempka A.A.
"Activating Neural Networks: Part 2", in
AI Expert Magazine (August 1994) 42.

[Kha93] Khan E.
"Neural Network Based Algorithms For Rule Evaluation and Defuzzification in Fuzzy Logic Design", in
Proceedings: *World Congress on Neural Networks* II (1993) 31.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Li93] Li C. & Wu C.H.
"A Real-Time Neural Fuzzy System-Based Fault Classifier in a Power Controller", in
Proceedings: *World Congress on Neural Networks* II (1993) 39.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Lin93] Lin D.T., Ligomenides P.A. & Dayhoff J.E.
"Learning Spatiotemporal Topology using an Adaptive Time Delay Neural Network", in
Proceedings: *World Congress on Neural Networks* I (1993) 291.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Lis94] Lisi N., Michaelis M.M., Buccellato R., Kuppen M. & Prause A.
"The Colliding Shock Lens as an Intracavity Q-Switch Element", in
Applied Optics (1994).

[Lip87] Lippmann R.P.
"An Introduction to Computing with Neural Nets", in
IEEE ASSP (April 1987) 4-22.

[Loo96] Looney C.G.
"Advances in Feedforward Neural Networks: Demystifying Knowledge Acquiring Black Boxes", in
IEEE Transactions on Knowledge and Data Engineering 8 (2) (April 1996) p211-226.

[Lop93] Lopez-Moliner J. & Sopena J.M.
"Learning to Induce Conditional Rules using Sequential Networks", in
Proceedings: *World Congress on Neural Networks* I (1993) 573.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Lur68] Luria A.R.
The Mind of a Mnemonist: a little book about a vast memory
Penguin Books, Harmondsworth, Middx, 1975.

[Mal87] Malsburg v.d. M.
"Goal and Architecture of Neural Computers" in
Proceedings: *NATO Advanced Research Workshop on Neural Computers* (1987) 23, published as *Neural Computers*, edited by Eckmiller R. & Malsburg C., Springer-Verlag, Berlin, 1987.

[Mar82] Marcuse D.
Light Transmission Optics, 2'nd edition, 146,
Van Nostrand Reinhold, New York, 1982.

[McC] McClelland J. L. & Rumelhart D. E.
Explorations in Parallel Distributed Processing - A Handbook of Models, Programs, and Exercises, 1'st edition, 84-99,
MIT Press, Cambridge, 1988.

[McL89] McLaren I.
"The Computational Unit as an Assembly of Neurones: an Implementation of an Error Learning Algorithm", in
The Computing Neuron (1989) 160, edited by Durbin R., Miall C. and Mitchison G.,
Addison-Wesley, New York 1989.

[Mia89] Miall C.
"The Diversity of Neuronal Properties", in
The Computing Neuron (1989) 11, edited by Durbin R., Miall C. and Mitchison G.,
Addison-Wesley, New York, 1989.

[Mic91] Michaelis M.M, Waltham J.A. & Cunningham P.F.
"Refractive Fringe Diagnostic of Spherical Shocks", in
Optics and Laser Technology 23 (1991) 283.

[Mic94] Michaelis M.M., Lisi N., Kuppen R., Buccellato R. & Prause A.
"Applications of the colliding shock lens", in
Laser and Particle Beams 12 (3) (1994) 1-8.

[Min91] Minsky M.
"Logic vs analogical, symbolic vs connectionist, neat vs scruffy", in
AI Magazine 12 (2) (1991).

[Mit89] Mitchison G.
"Learning Algorithms and Networks of Neurons", in
The Computing Neuron (1989) 35, edited by Durbin R., Miall C. and Mitchison G.,
Addison-Wesley, New York, 1989.

[Moo87] Moore W.R.
"Conventional Fault Tolerance and Neural Networks", in
Proceedings: *NATO Advanced Research Workshop on Neural Computers* (1987) 29, published as *Neural Computers* edited by Eckmiller R. & Malsburg C., Springer-Verlag, Berlin, 1987.

[Mor93] Morabito F.C.
"Multi-layer Neural Network for identification of non-linear Electromagnetic systems", in
Proceedings: *World Congress on Neural Networks* I (1993) 428.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Nas93] Nassersharif B. & Rajan V.
"Automated Training of Backpropagation Neural Networks Using a Real Time Expert System", in
Proceedings: *World Congress on Neural Networks* I (1993) 536.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Neu91] *NeuralWorks Professional II/PLUS and NeuralWorks Explorer Reference Guide* 51, 95, 199-204
NeuralWare Inc., Pittsburgh (1991).

[Nit93a] Nitta T.
"A Three-dimensional Back-propagation", in
Proceedings: *World Congress on Neural Networks* III (1993) 572.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Nit93b] Nitta T.
"A Complex Numbered Version of the Back-propagation Algorithm", in
Proceedings: *World Congress on Neural Networks* III (1993) 576.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Obr93] Obradovic' Z. & Fletcher J.
"Integration of Knowledge-Based and Constructive Learning Neural Networks", in
Proceedings: *World Congress on Neural Networks* I (1993) 589.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Ocu87] *Oculus 200 Frame Grabber Manual*, 1.
Coreco Inc., 1987.

[Pai83] Pain H.J.
The Physics of Vibrations and Waves, 3rd edition, 280
John Wiley and Sons, New York, 1983.

[Per92] Peretto P.
An Introduction to the Modelling of Neural Networks, 13-55,
University Press, Cambridge, 1992.

[Pin95] Pinkas G.
"Reasoning non monotonicity and learning in connectionist networks that capture propositional knowledge", in
Artificial Intelligence 77 (2) (1995) 203-247, edited by Bobrow D. G. & Brady M., Elsevier Science, Amsterdam, 1995.

[Por95] Porto V.W., Fogel D.B. & Fogel L.J.
"Alternative Neural Network Training Methods", in
IEEE Expert (June 1995) 16.

[Qua93a] Quah T.S. & Tan C.L.
"A Connectionist Shell for Rule-based Reasoning", in
Proceedings: *World Congress on Neural Networks* I (1993) 543.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Qua93b] Quah T.S., Tan C.L., Teh H.H. & Shen Z.
"An Enhanced NEURONET for Rule-based Reasoning", in
Proceedings: *World Congress on Neural Networks* I (1993) 605.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Rob94a] Robinson C.A. (reporter)
"Bioelectronics Computer Era Merges Organic, Solid State", in
Signal Magazine (official publication of AFCEA), (February 1994) 15-17, edited by Robinson C.A.

[Rob94b] Robinson C.A. (reporter)
"Selective Biological Cells Grow as Neural Networks"
Signal Magazine (official publication of AFCEA), (February 1994) 19-20, edited by Robinson C.A.

[Rum86] Rumelhart D.E., McClelland J.L. et al.
Parallel Distributed Processing: Explorations in the Microstructure of Cognition Volume 1, 12, 131-133,
MIT Press, Massachusetts, 1986.

[Sch93] Schempp W.
"A Paradigm Shift in the Neurodynamics of Spatio-Temporal Neural Networks: The Coherent Wavelet Revolution in Linking Neural Network Architectures, Quantum Holography and Chaos", in
Proceedings: *World Congress on Neural Networks* IV (1993) 443.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Sea87] Sears F.W., Zemansky M.W. & Young H.D.
University Physics, 7th edition, 842, 913-921.
Addison-Wesley, Massachusetts, 1987.

[Sed59] Sedov L. I.
Methods of Similarity and Dimensional Analysis in Mechanics, 3rd edition
Academic Press, New York, 1959.

[Sej87] Sejnowski, T.J.
"Neural Network Learning Algorithms", in
Proceedings: *NATO Advanced Research Workshop on Neural Computers* (1987) 292, published as *Neural Computers*, edited by Eckmiller R. & Malsburg C., Springer-Verlag, Berlin, 1987.

[Set96] Setiono R. & Lui H.
"Symbolic Representation of Neural Networks", in
IEEE Computer 29 (3) (March 1996) 71-77.

[Sha93] Shann J.J. & Fu H.C.
"Backpropagation Learning for Acquiring Fine Knowledge of Fuzzy Neural Networks", in
Proceedings: *World Congress on Neural Networks* II (1993) 66.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Shi93] Shishkov S.I. & Trifonov R.I.
"A Hybrid Neural Production System", in
Proceedings: *World Congress on Neural Networks* I (1993) 593.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Shl88] Shlaer S. & Mellor S.J.
Object-Orientated Systems Analysis: Modeling the World in Data
Prentice Hall, Englewood Cliffs, N.J. (1988).

[Shl92] Shlaer S. & Mellor S.J.
Object Lifecycles: Modeling the World in States
Prentice Hall, Englewood Cliffs, N.J. (1992).

[Sou93] Southwell D.T. & Allinson N.M.
"Temporal pre-synaptic effects in Columnar Array Networks", in
Proceedings: *World Congress on Neural Networks* I (1993) 511.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Ste93] Steck J.E., Skinner S.R. & Behram E.C.
"Using χ^3 Nonlinear Optics for Neural Network Implementations", in
Proceedings: *World Congress on Neural Networks* IV (1993) 808.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Tak93] Takagi H.
"Neural Network and Genetic Algorithm techniques for Fuzzy Systems", in
Proceedings: *World Congress on Neural Networks* II (1993) 631.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Tan96] Tan C.L., Quah T.S. & Teh H.H.
"An Artificial Neural Network that Models Human Decision Making", in
IEEE Computer 29 (3) (March 1996) 64.

[Tho94] Thodberg H.H.
"Bayesian Backprop in Action: Pruning, Committees, Error Bars and an Application to Spectroscopy", in
Advances in Neural Information Processing Systems 6 (1994) 208, edited by Cowan J.D., Tesauro G. and Alspector J., Morgan Kaufmann Publishers, San Francisco, 1994.

[Uma69] Uman M.A.
Lightning Advanced Physics Monograph Series, 181-182,
McGraw-Hill, New York, 1969.

[Van93] Vanderbeek R., Butler M., El-Dinary A. & Strohbehn K.
"The Design of an Artificial Leg Controlled with Silicon Neurons", in
Proceedings: *World Congress on Neural Networks* III (1993) 149.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Wal90] Waltham J.A.R.
Ph.D. Thesis: Investigations of Laser-Produced Plasmas and Associated Phenomena, 74, 195.
University of Natal, Durban, 1990.

[Was87] Wasserman P.D. & Schwartz T.
"Neural Networks Part 2", in
IEEE Expert (Spring 1988) 10-15.

[Was89] Wasserman P.D.
Neural Computing - Theory and Practice, 1'st edition, 43-59, 151-165.
Van Nostrand Reinhold, New York, 1989.

[Was93] Wasserman P.D.
Advanced Methods in Neural Computing, 74-115.
Van Nostrand Reinhold, New York, 1993.

[Wei91] Weiss S.M. & Kulikowski C.A.
Computer Systems That Learn - Classification and Prediction Models from Statistics, Neural Nets, Machine Learning and Expert Systems, 1'st edition, 95.
Morgan Kaufmann Publishers, San Mateo, 1991.

[Yag94] Yager R.R. & Zadeh L.A.
Fuzzy Sets, Neural Networks and Soft Computing, 126, 233, 345.
Van Nostrand Reinhold, New York, 1994.

[Yin93] Ying X., Surkan A.J. and Guan Q.
"Simplifying Neural Networks by Pruning Alternated with Backpropagation Training", in Proceedings: *World Congress on Neural Networks* III (1993) 364.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

[Zel66] Zel'dovich Ya. B. & Raizer Yu. P.
Physics of Shock Waves and High Temperature Hydrodynamic Phenomena, Volume 2 636.
Academic Press, New York (1966).

[Zha93] Zhang D., Kamel M. & Elmasry M.I.
"Fuzzy Clustering Neural Network (FCNN) Using Fuzzy Competitive Learning", in Proceedings: *World Congress on Neural Networks* II (1993) 22.
Lawrence Erlbaum Associates, Hillsdale N.J., 1993.

Appendix A Sample Training Data

Table A.1 Sample Training Data

Density Profile	Interference Pattern

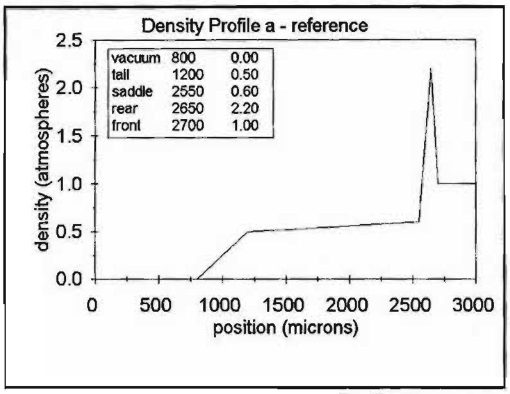


Figure A.1.a Theoretical Density Profile a.

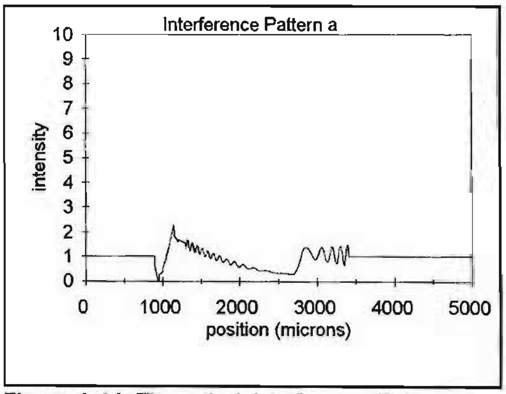


Figure A.1.b Theoretical Interference Pattern a.

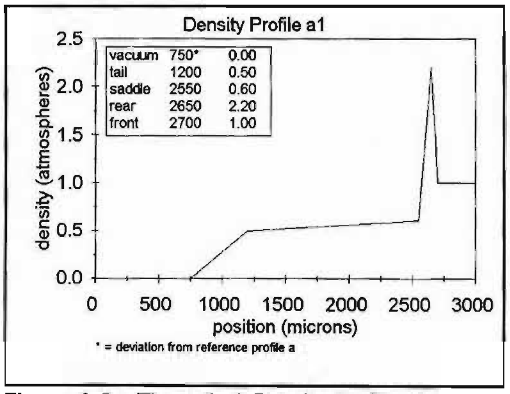


Figure A.2.a Theoretical Density Profile a1.

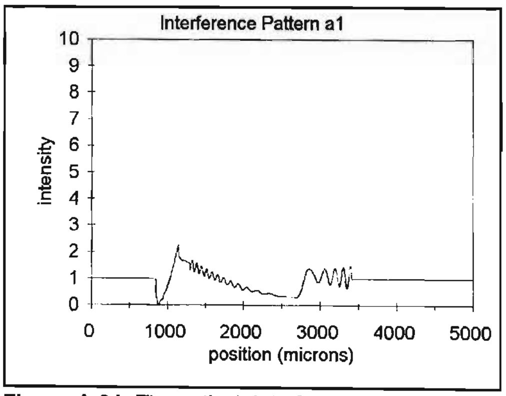


Figure A.2.b Theoretical Interference Pattern a1.

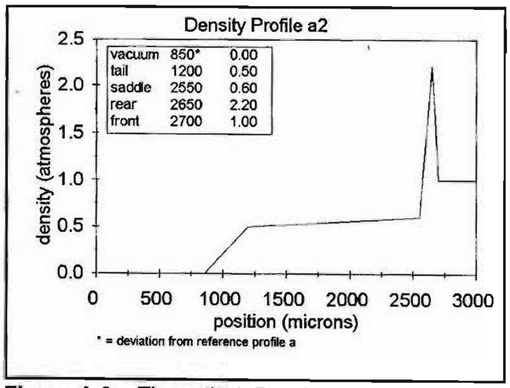


Figure A.3.a Theoretical Density Profile a2.

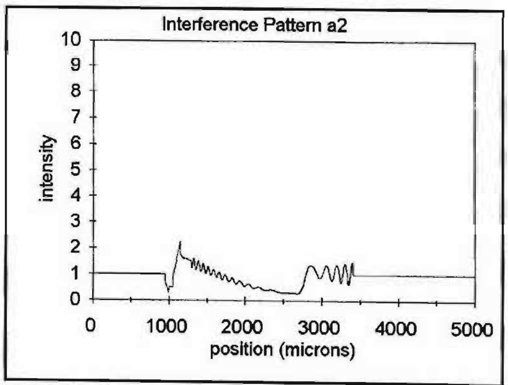


Figure A.3.b Theoretical Interference Pattern a2.

Table A.1 Sample Training Data

Density Profile | **Interference Pattern**

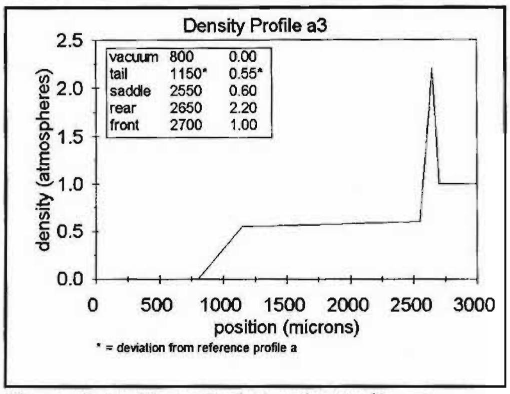


Figure A.4.a Theoretical Density Profile a3.

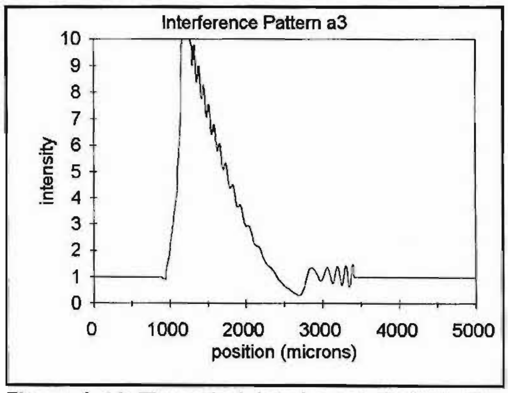


Figure A.4.b Theoretical Interference Pattern a3.

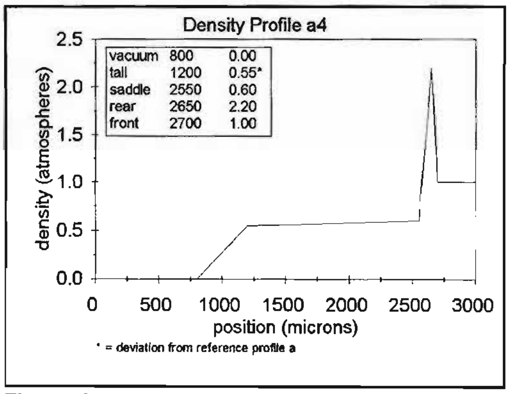


Figure A.5.a Theoretical Density Profile a4.

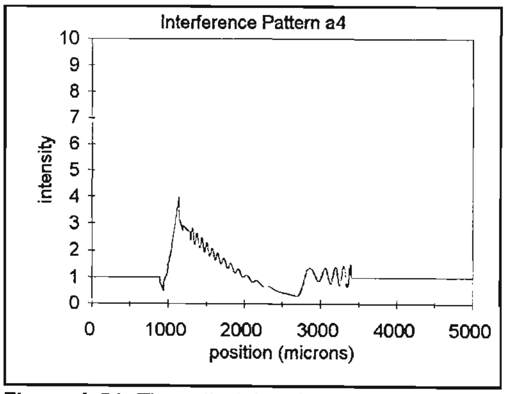


Figure A.5.b Theoretical Interference Pattern a4.

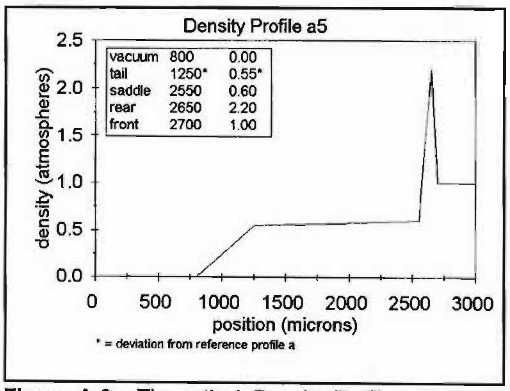


Figure A.6.a Theoretical Density Profile a5.

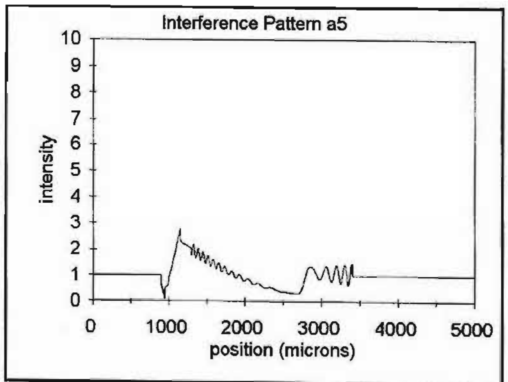


Figure A.6.b Theoretical Interference Pattern a5.

Table A.1 Sample Training Data

Density Profile | **Interference Pattern**

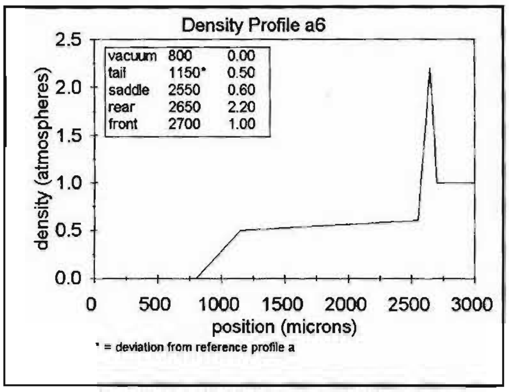


Figure A.7.a Theoretical Density Profile a6.

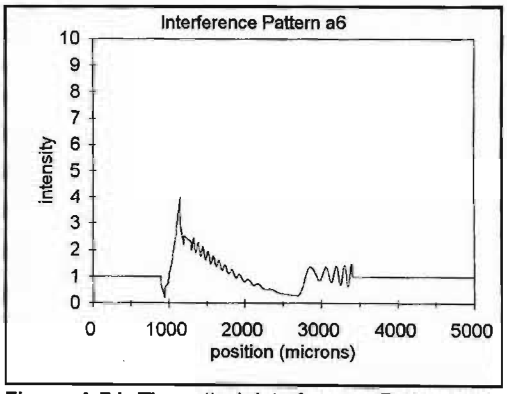


Figure A.7.b Theoretical Interference Pattern a6.

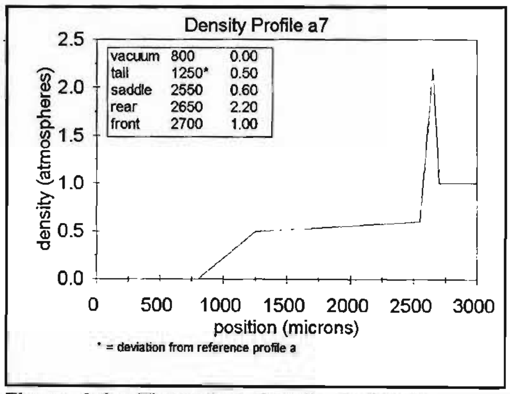


Figure A.8.a Theoretical Density Profile a7.

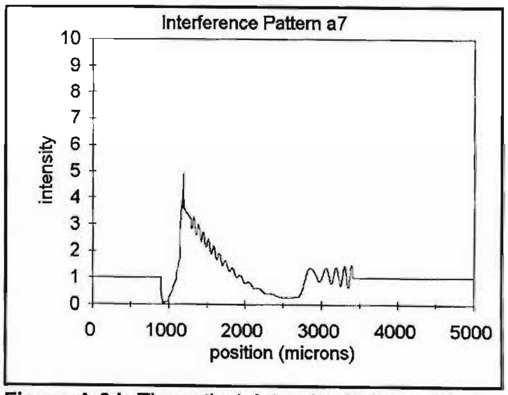


Figure A.8.b Theoretical Intensity Pattern a7.

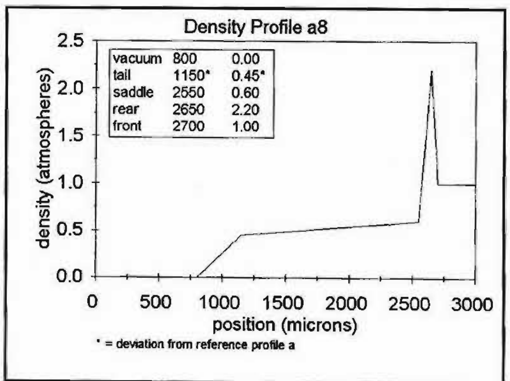


Figure A.9.a Theoretical Density Profile a8.

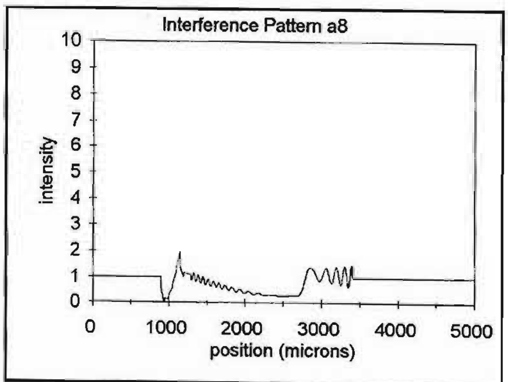


Figure A.9.b Theoretical Interference Pattern a8.

Table A.1 Sample Training Data

Density Profile	Interference Pattern

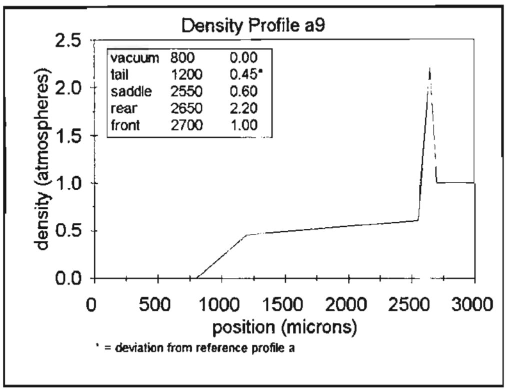


Figure A.10.a Theoretical Density Profile a9.

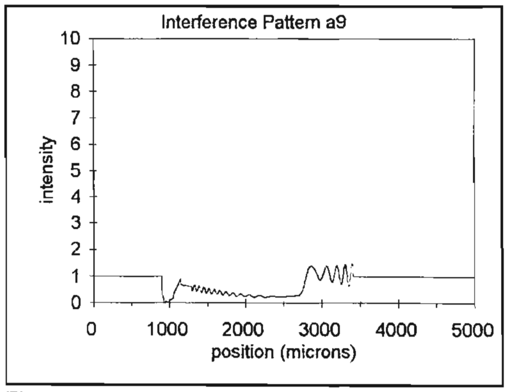


Figure A.10.b Theoretical Interference Pattern a9.

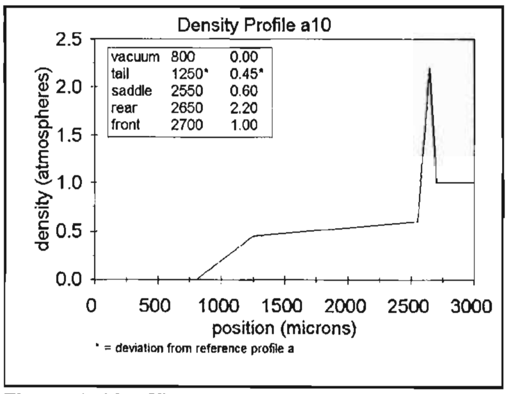


Figure A.11.a Theoretical Density Profile a10.

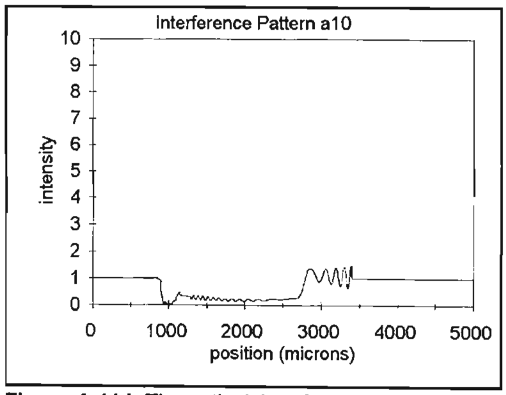


Figure A.11.b Theoretical Interference Pattern a10.

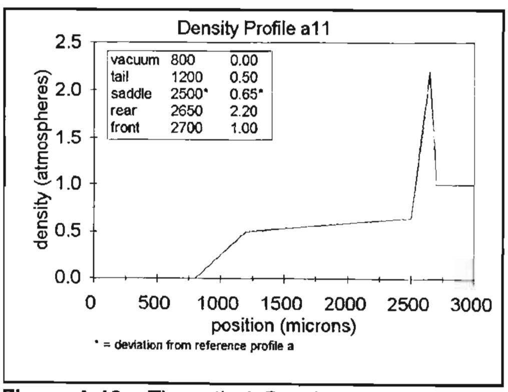


Figure A.12.a Theoretical Density Profile a11.

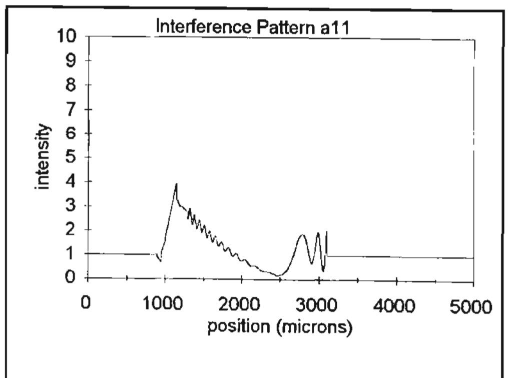


Figure A.12.b Theoretical Interference Pattern a11.

Table A.1 Sample Training Data

Density Profile	Interference Pattern

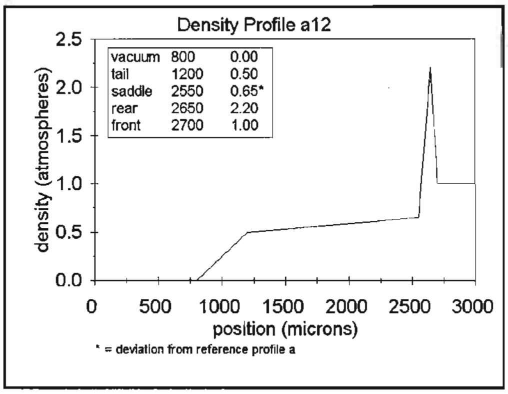


Figure A.13.a Theoretical Density Profile a12.

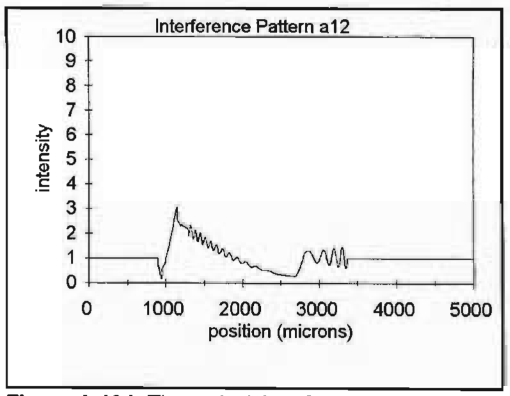


Figure A.13.b Theoretical Interference Pattern a12.

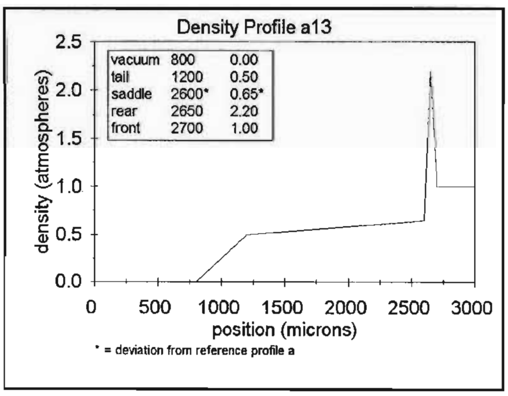


Figure A.14.a Theoretical Density Profile a13.

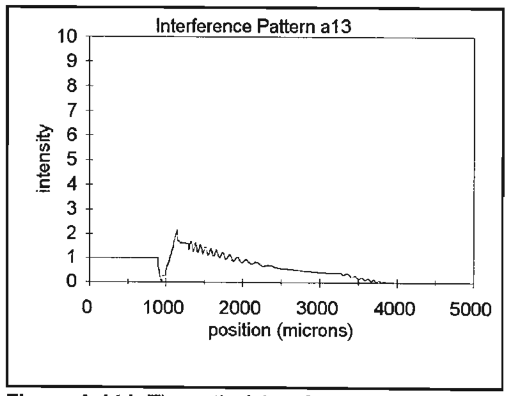


Figure A.14.b Theoretical Interference Pattern a13.

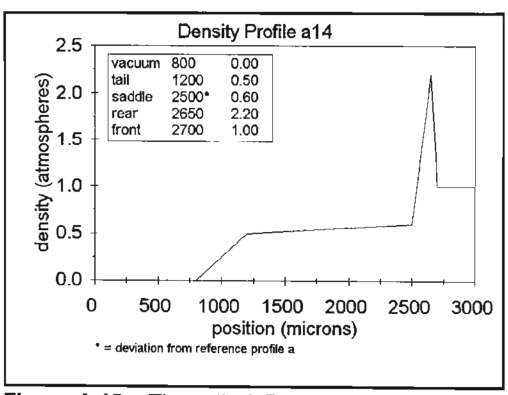


Figure A.15.a Theoretical Density Profile a14.

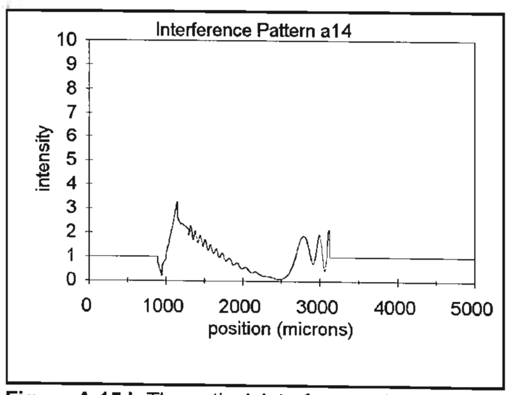


Figure A.15.b Theoretical Interference Pattern a14.

Table A.1 Sample Training Data

Density Profile	Interference Pattern

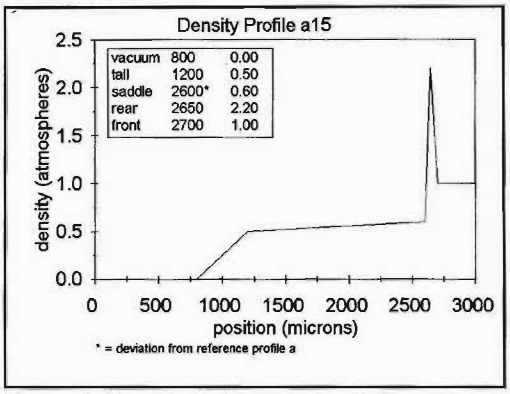


Figure A.16.a Theoretical Density Profile a15.

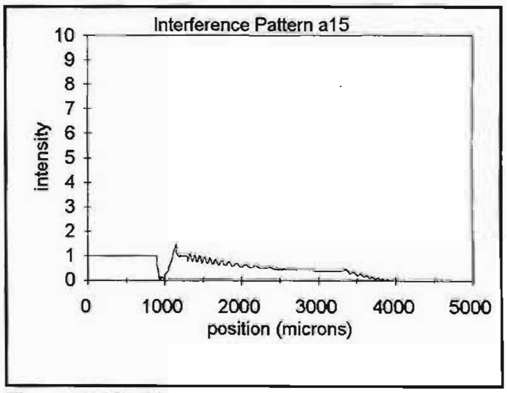


Figure A.16.b Theoretical Interference Pattern a15.

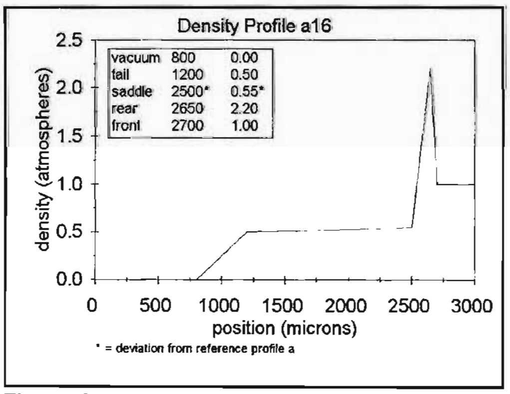


Figure A.17.a Theoretical Density Profile a16.

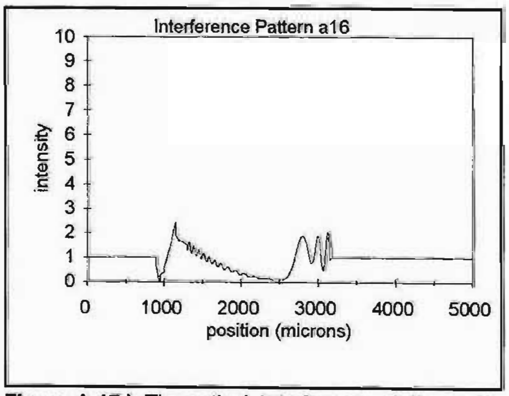


Figure A.17.b Theoretical Interference Pattern a16.

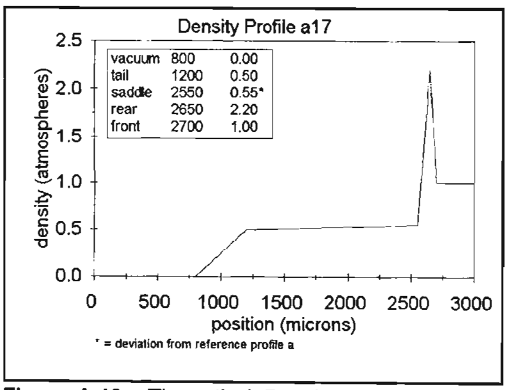


Figure A.18.a Theoretical Density Profile a17.

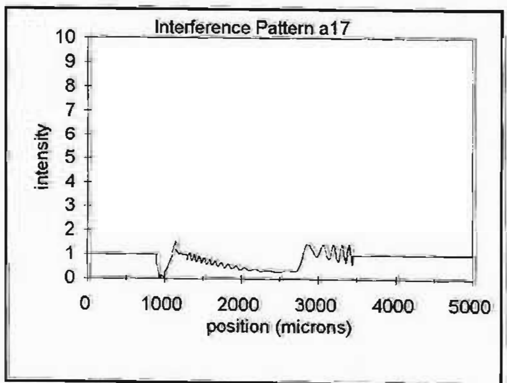


Figure A.18.b Theoretical Interference Pattern a17.

Table A.1 Sample Training Data

Density Profile	Interference Pattern

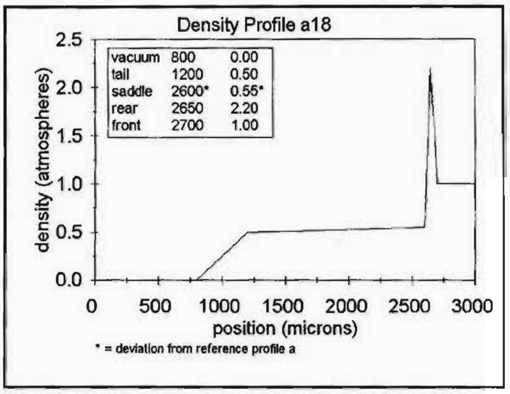


Figure A.19.a Theoretical Density Profile a18.

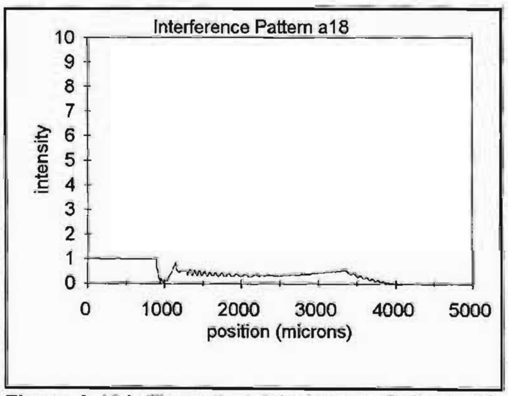


Figure A.19.b Theoretical Interference Pattern a18.

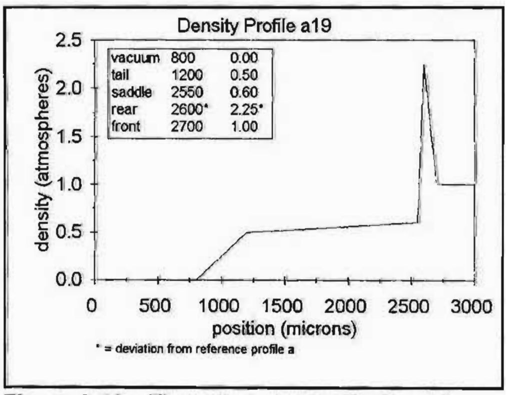


Figure A.20.a Theoretical Density Profile a19.

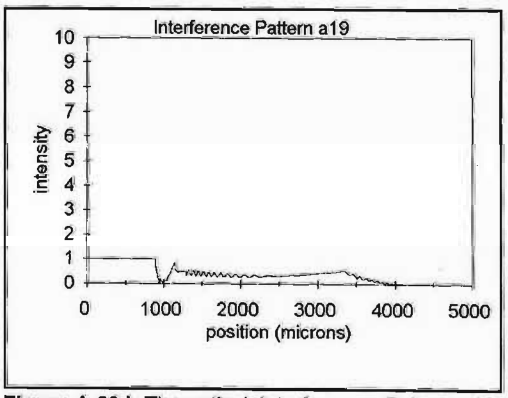


Figure A.20.b Theoretical Interference Pattern a19.

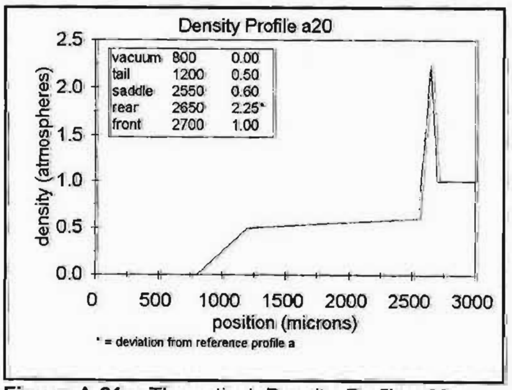


Figure A.21.a Theoretical Density Profile a20.

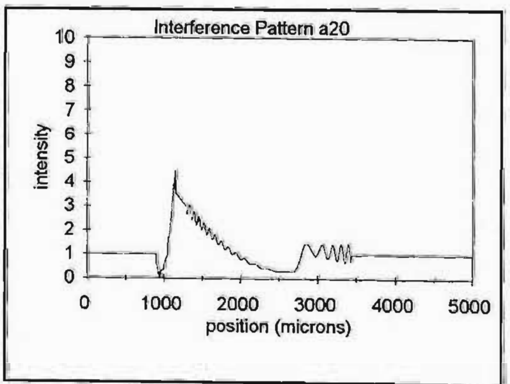
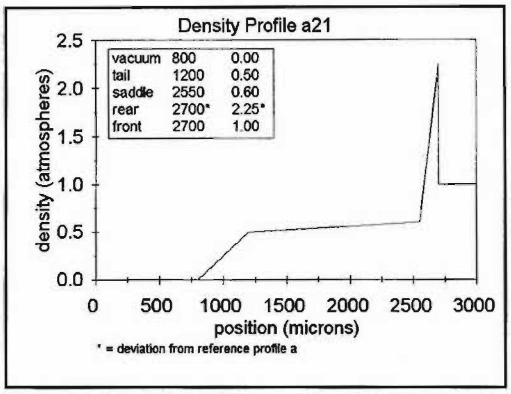
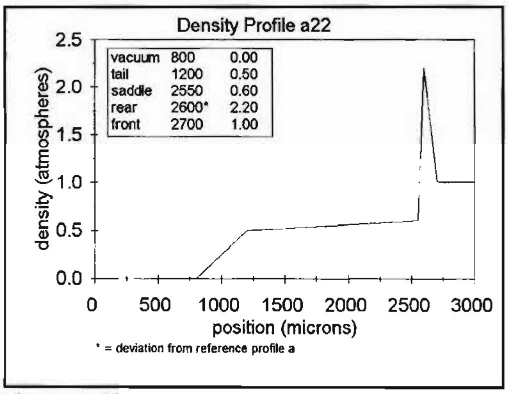
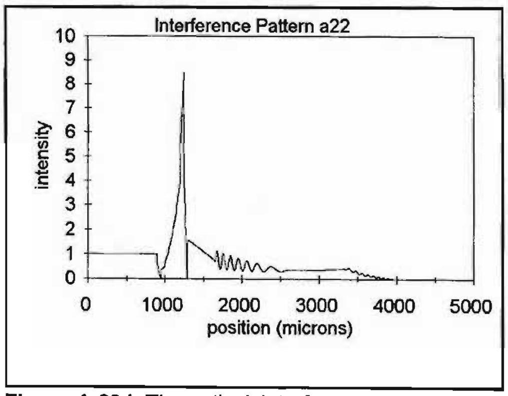
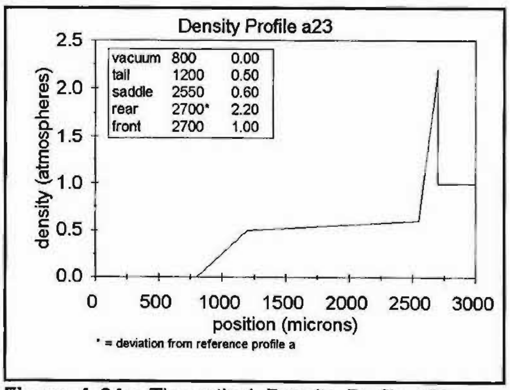


Figure A.21.b Theoretical Interference Pattern a20.

Table A.1 Sample Training Data

Density Profile	Interference Pattern
 **Figure A.22.a** Theoretical Density Profile a21.	The ray-tracer is unable to generate an interference pattern for the density profile parameters on the left.
 **Figure A.23.a** Theoretical Density Profile a22.	 **Figure A.23.b** Theoretical Interference Pattern a22.
 **Figure A.24.a** Theoretical Density Profile a23.	The ray-tracer is unable to generate an interference pattern for the density profile parameters on the left.

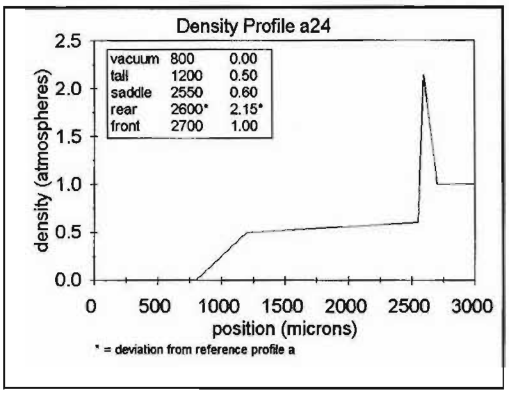
<table>
<tr><th colspan="2">Table A.1 Sample Training Data</th></tr>
<tr><th>Density Profile</th><th>Interference Pattern</th></tr>
<tr><td>


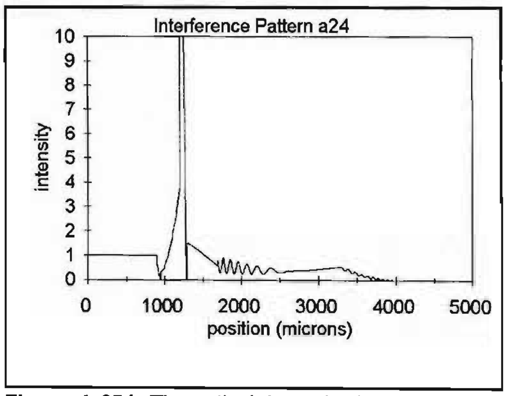
Figure A.25.a Theoretical Density Profile a24.
</td><td>


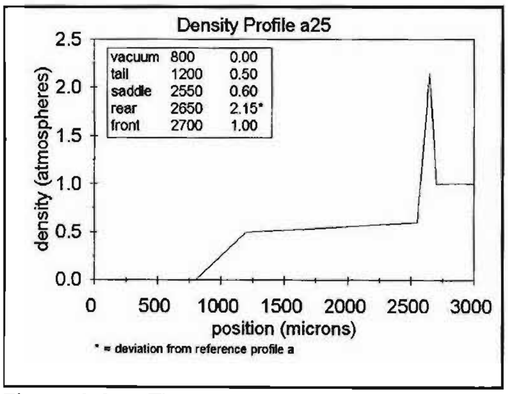
Figure A.25.b Theoretical Intensity Pattern a24.
</td></tr>
<tr><td>


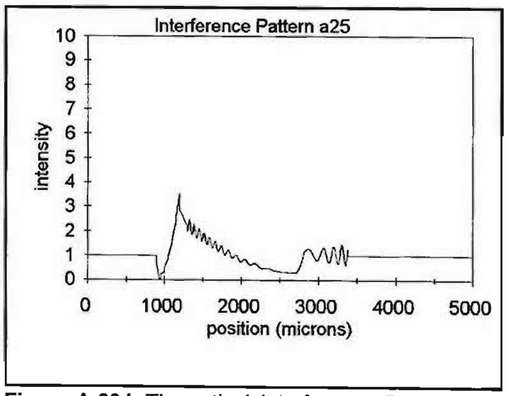
Figure A.26.a Theoretical Density Profile a25.
</td><td>


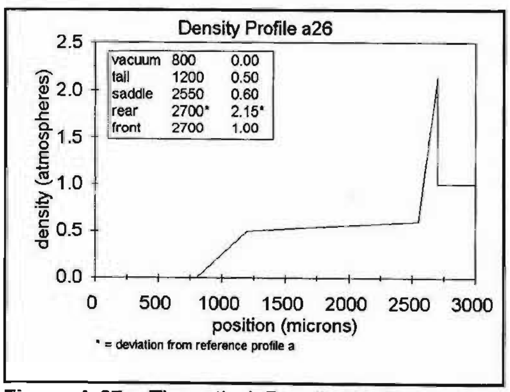
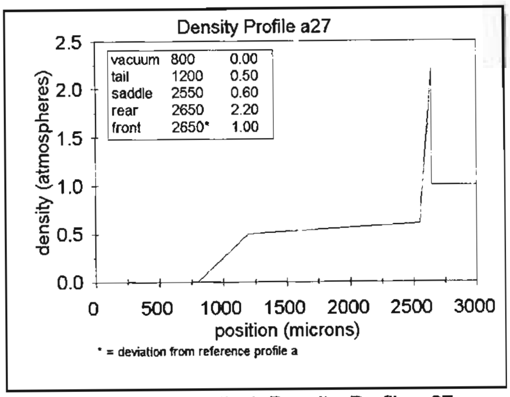
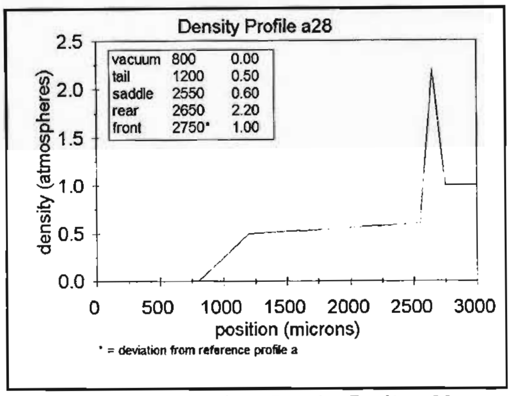
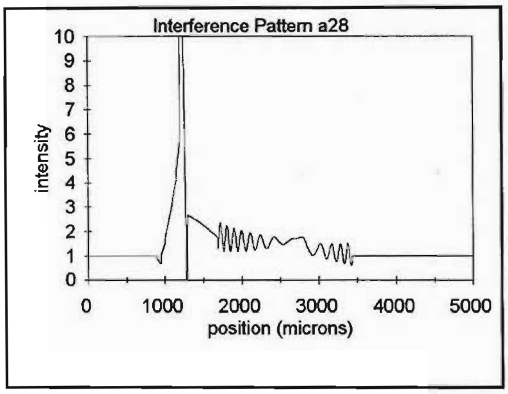
Figure A.26.b Theoretical Interference Pattern a25.
</td></tr>
<tr><td>


Figure A.27.a Theoretical Density Profile a26.
</td><td>
The ray-tracer is unable to generate an interference pattern for the density profile parameters on the left.
</td></tr>
</table>

Table A.1 Sample Training Data

Density Profile	Interference Pattern
 **Figure A.28.a** Theoretical Density Profile a27.	The ray-tracer is unable to generate an interference pattern for the density profile parameters on the left.
 **Figure A.29.a** Theoretical Density Profile a28.	 **Figure A.29.b** Theoretical Interference Pattern a28.

This subset of the training data was generated by altering the parameters of one shock region at a time. The full training set is generated by permuting the parameters of multiple shock regions simultaneously. The density profiles in figures (A.22.a), (A.24.a), (A.27.a) and (A.28.a) are included to demonstrate that the computational ray-tracer cannot produce an interference pattern when the shock front is completely vertical. Certain extreme values for the parameters, or certain combinations of individually valid values, also stress the ray-tracer. The density profiles fed to the ray-tracer must therefore be selected with some attention to their physical plausibility. A listing of the density parameters used to generate the complete training set is given in Appendix C.

Appendix B File Formats

B.1 Raytrace Macro File

```
!-------------------------------------------------------------------------
! Sample Raytrace Macro File
!-------------------------------------------------------------------------
new:
vacuum:          800   0.00000000
tail:           1200   1.12300000
saddle:         1500   1.24500000
rear:           2700   2.45670000
front:          3000   1.00000000
firerays:        800   3000
interpolate:    linear
display:        yes
parameterfile:  para1j.pks
densityfile:    dens1j.raw
intensityfile:  inten1j.raw
end:

new:
vacuum:          800   0.00000000
tail:           1200   1.10000000
saddle:         1500   1.42300000
rear:           2700   2.56000000
front:          3000   1.00000000
firerays:        800   3000
interpolate:    linear
display:        yes
parameterfile:  para2j.pks
densityfile:    dens2j.raw
intensityfile:  inten2j.raw
end:
```

Comments begin with the '!' character and must occupy the entire line. Each keyword and its parameters must occupy a separate line. With the exception of the **new:** and **end:** keywords bracketing each set, the keywords may be listed in any order. The **display:**, **parameterfile:**, **densityfile:** and **intensityfile:** keywords are optional. They specify if the interference pattern should be displayed on screen and the parameters saved to disk. The **display:** parameter may assume values of *yes* and *no*. The file parameters may have any legal DOS path and filename and do not reserve any particular file extension. The **interpolate:** parameter specifies the interpolation method to be used to interpolate the density parameters and may take on the values: *linear*, *lagrange*, *cubic* or *junction*. The macro file is case sensitive to text. The **vacuum:**, **tail:**, **saddle:**, **rear:** and **front:** parameters encode first the position in microns, and then the density in atmospheres, of the corresponding shock density profile features. The **firerays:** parameters specify the start and end positions, in microns, within which the ray-tracer must fire its rays. The ray-tracer will check the parameters for consistency before executing a set of instructions from a macro file. It will flag syntax and consistency errors to the user before attempting to parse any further sets in the macro file.

B.2 Raytrace Batch File

```
ray  811 0.00 1204 0.52 2485 0.56 2738 2.21 2695 1.00  811 2695 i z3.shk z3.den z3.int
ray  819 0.00 1185 0.51 2396 0.70 2545 2.20 2732 1.00  819 2732 i z4.shk z4.den z4.int
ray  812 0.00 1206 0.45 2381 0.63 2607 2.28 2794 1.00  812 2794 i z5.shk z5.den z5.int
ray  805 0.00 1183 0.49 2486 0.55 2675 2.26 2776 1.00  805 2776 i z6.shk z6.den z6.int
ray  807 0.00 1157 0.54 2524 0.58 2599 2.21 2759 1.00  807 2759 i z8.shk z8.den z8.int
```

Unlike the raytrace macro file, the raytrace batch file has a very specific ordering of parameters. These are the name of the program to invoke, the vacuum position and density, tail position and density, saddle position and density, rear position and density, front position and density, start and end raytrace limits, interpolation method and parameter, density and intensity filenames. Position has units of microns and density is expressed in atmospheres. The interpolation method uses the letters, **i**, **a**, **c** and **j** to encode, respectively, the **l**inear, l**a**grange, **c**ubic and **j**unction methods. Substitution of the word **none** for any of the filenames means that the corresponding file will not be saved. Some density parameters may cause the ray-tracer to crash. When this happens the ray-tracer returns to the DOS command interpreter and executes the next instruction in the batch file. The ray-tracer takes quite a few minutes to generate each intensity pattern. Since we need many thousands of patterns, it is important that processing of the batch files runs as smoothly as possible as it is likely that they will run unattended for lengthy periods.

B.3 Raytrace Parameter File

```
RayTrace Neural Network Parameter File
        850           0.000000000000000000
       1250           0.550000000000000000
       2550           0.650000000000000000
       2700           2.150000000000000000
       2750           1.000000000000000000
20
                             1405          9.729308294183276470
                             1470          8.849888292411240090
                             1493          9.432419437691015320
                             1527          7.912474113505405530
                             1563          8.361011290838646910
                             1599          6.966897688842785820
                             1624          7.393986628811651580
                             1675          6.078858943867466280
                             1702          6.398889874699841050
                             1771          5.390643754163167320
                             1831          4.271846376712410630
                             1862          4.385223033311142160
                             1944          3.334185253851512430
                             1961          3.387568930928524400
                             2521          0.163656067783555681
                             2750          1.273177884920649650
                             2751          1.270837733047983200
                             3037          1.998455765342119110
                             3104          0.320948202633931423
                             3141          1.765371394276879170
```

The file starts with a header and then lists the shock's vacuum position and density, tail position and density, saddle position and density, rear position and density and front position and density in units of microns and atmospheres. It then records the number of peaks and troughs in the generated intensity pattern before listing the position, in microns, and absolute intensity of each extrema.

After normalisation, this file is used as input for the neural network training file.

B.4 Neural Network Training File

```
!------------------------------------------------------------------ G3876.SHK 36
  0.255333 0.792762 0.255556 0.702208 0.266444 0.840219 0.277556 0.709412 0.277778 0.709477
0.288889 0.654574 0.292444 0.701358 0.300444 0.594593 0.305778 0.653942 0.314000 0.556817
0.319556 0.605368 0.325111 0.516831 0.334000 0.555282 0.340000 0.472308 0.346000 0.511000
0.355556 0.429488 0.362000 0.466175 0.372000 0.387831 0.378889 0.419633 0.389556 0.345406
0.397111 0.369363 0.408667 0.300991 0.416889 0.318275 0.429556 0.256725 0.438667 0.267146
& 0.463556 0.215196 0.572222 0.057259 0.572444 0.058274 0.610222 0.033051 0.611111 0.035265
0.611333 0.035001 0.689778 0.145497 0.720889 0.144264 0.744222 0.144689 0.765556 0.146295
0.775111 0.061649 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
& 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
& 0.283333 0.000000 0.400000 0.220000 0.866667 0.240000 0.900000 0.900000 0.916667 0.400000
!------------------------------------------------------------------ G3877.SHK 40
  0.222000 0.077879 0.301556 0.997303 0.315778 0.945780 0.330667 0.896280 0.335778 0.970029
0.346444 0.847745 0.352000 0.912546 0.360444 0.794152 0.369111 0.850755 0.378222 0.736559
0.387556 0.782718 0.397556 0.678519 0.404222 0.714334 0.418667 0.616597 0.426222 0.641447
0.442444 0.551478 0.451111 0.566150 0.480667 0.485242 0.564444 0.286294 0.564667 0.288775
0.599778 0.228481 0.600000 0.229748 0.734889 0.079422 0.748000 0.099487 0.768444 0.101799
& 0.780000 0.081035 0.787333 0.103436 0.798000 0.081229 0.804889 0.105309 0.814889 0.081452
0.821333 0.106609 0.830667 0.081897 0.836889 0.109292 0.845778 0.082564 0.851556 0.112118
0.860000 0.083651 0.865556 0.114791 0.873556 0.085336 0.878889 0.117038 0.884000 0.085567
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
& 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
& 0.283333 0.000000 0.400000 0.220000 0.866667 0.260000 0.883333 0.860000 0.900000 0.400000
...
```

This file is in standard NeuralWorks format. The comment character is '!' and the '**&**' character indicates the continuation of a logical row onto a new line. The normalised positions and intensities of 60 intensity peaks and troughs appear first. If the intensity pattern has fewer extrema the unused values at the end are set to 0. Extra extrema are ignored. The normalised position and density of the shock density profile parameters appear last as these form the 'teacher' pattern for the neural network.

B.5 Neural Network Testing File

```
!----------------------------------------------------------------- G4014.SHK 35
  0.222000 0.014887 0.244222 0.156236 0.244444 0.141453 0.255333 0.168065 0.277556 0.143929
0.277778 0.144004 0.288889 0.129511 0.292444 0.155215 0.300444 0.114413 0.305778 0.147447
0.319556 0.139008 0.325111 0.103747 0.334000 0.129849 0.340000 0.096438 0.349111 0.120104
0.355556 0.089768 0.362000 0.112135 0.372000 0.083604 0.378889 0.104570 0.389556 0.077268
0.397111 0.094890 0.408667 0.070327 0.416889 0.084709 0.429556 0.063352 0.438667 0.074473
& 0.453111 0.055969 0.463556 0.063983 0.493111 0.052916 0.556000 0.029835 0.611111 0.159659
0.611333 0.159256 0.655556 0.078349 0.673778 0.195844 0.689333 0.045513 0.702667 0.210641
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0
& 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
& 0.283333 0.000000 0.416667 0.180000 0.850000 0.260000 0.900000 0.900000 0.916667 0.400000
!----------------------------------------------------------------- G4081.SHK 24
  0.222000 0.013289 0.266444 0.296778 0.288889 0.191496 0.294444 0.236022 0.301556 0.171466
0.308444 0.214863 0.315778 0.155340 0.323111 0.192565 0.330667 0.140489 0.335778 0.172509
0.352000 0.153664 0.360444 0.109355 0.369111 0.133700 0.378222 0.093244 0.387556 0.112354
0.397556 0.077761 0.418667 0.061867 0.442444 0.046506 0.451111 0.052979 0.480667 0.034843
0.622889 0.181247 0.648444 0.064310 0.666000 0.195997 0.691556 0.216134 0.0 0.0
& 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
& 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
& 0.283333 0.000000 0.416667 0.200000 0.833333 0.240000 0.883333 0.860000 0.900000 0.400000
...
```

This example is identical to the network training file. The presence of the teacher output pattern means that in addition to reporting the network's response to the input pattern, NeuralWorks will also calculate the error in the response of each output neuron. The teacher output pattern is used for cross-validation testing on a reserved test-set. It is not necessary in a network testing file and does not appear when it is not known.

Appendix C Density Parameter Listing for the Training Set

Table C.1 Full Training Set Density Parameter Listing

file	vacuum pos µm	vacuum den atm	tail pos µm	tail den atm	saddle pos µm	saddle den atm	rear pos µm	rear den atm	front pos µm	front den atm	file	vacuum pos µm	vacuum den atm	tail pos µm	tail den atm	saddle pos µm	saddle den atm	rear pos µm	rear den atm	front pos µm	front den atm
G3876	850	0.00	1200	0.55	2600	0.60	2700	2.25	2750	1.00	G1867	800	0.00	1150	0.55	2550	0.60	2600	2.15	2650	1.00
G3877	850	0.00	1200	0.55	2600	0.65	2650	2.15	2700	1.00	G1868	800	0.00	1150	0.55	2550	0.60	2600	2.15	2700	1.00
G3878	850	0.00	1200	0.55	2600	0.65	2650	2.15	2750	1.00	G1870	800	0.00	1150	0.55	2550	0.60	2600	2.20	2650	1.00
G3879	850	0.00	1200	0.55	2600	0.65	2650	2.20	2700	1.00	G1871	800	0.00	1150	0.55	2550	0.60	2600	2.20	2700	1.00
G3880	850	0.00	1200	0.55	2600	0.65	2650	2.20	2750	1.00	G1873	800	0.00	1150	0.55	2550	0.60	2600	2.25	2650	1.00
G3881	850	0.00	1200	0.55	2600	0.65	2650	2.25	2700	1.00	G1874	800	0.00	1150	0.55	2550	0.60	2600	2.25	2700	1.00
G3882	850	0.00	1200	0.55	2600	0.65	2650	2.25	2750	1.00	G1876	800	0.00	1150	0.55	2550	0.60	2650	2.15	2700	1.00
G3884	850	0.00	1200	0.55	2600	0.65	2700	2.15	2750	1.00	G1877	800	0.00	1150	0.55	2550	0.60	2650	2.15	2750	1.00
G3886	850	0.00	1200	0.55	2600	0.65	2700	2.20	2750	1.00	G1878	800	0.00	1150	0.55	2550	0.60	2650	2.20	2700	1.00
G3888	850	0.00	1200	0.55	2600	0.65	2700	2.25	2750	1.00	G1879	800	0.00	1150	0.55	2550	0.60	2650	2.20	2750	1.00
G3889	850	0.00	1250	0.45	2500	0.55	2600	2.15	2650	1.00	G1880	800	0.00	1150	0.55	2550	0.60	2650	2.25	2700	1.00
G3890	850	0.00	1250	0.45	2500	0.55	2600	2.15	2700	1.00	G1881	800	0.00	1150	0.55	2550	0.60	2650	2.25	2750	1.00
G3892	850	0.00	1250	0.45	2500	0.55	2600	2.20	2650	1.00	G1883	800	0.00	1150	0.55	2550	0.60	2700	2.15	2750	1.00
G3893	850	0.00	1250	0.45	2500	0.55	2600	2.20	2700	1.00	G1887	800	0.00	1150	0.55	2550	0.60	2700	2.25	2750	1.00
G3895	850	0.00	1250	0.45	2500	0.55	2600	2.25	2650	1.00	G1888	800	0.00	1150	0.55	2550	0.65	2600	2.15	2650	1.00
G3896	850	0.00	1250	0.45	2500	0.55	2600	2.25	2700	1.00	G1889	800	0.00	1150	0.55	2550	0.65	2600	2.15	2700	1.00
G3898	850	0.00	1250	0.45	2500	0.55	2650	2.15	2700	1.00	G1891	800	0.00	1150	0.55	2550	0.65	2600	2.20	2650	1.00
G3899	850	0.00	1250	0.45	2500	0.55	2650	2.15	2750	1.00	G1892	800	0.00	1150	0.55	2550	0.65	2600	2.20	2700	1.00
G3900	850	0.00	1250	0.45	2500	0.55	2650	2.20	2700	1.00	G1894	800	0.00	1150	0.55	2550	0.65	2600	2.25	2650	1.00
G3901	850	0.00	1250	0.45	2500	0.55	2650	2.20	2750	1.00	G1895	800	0.00	1150	0.55	2550	0.65	2600	2.25	2700	1.00
G3903	850	0.00	1250	0.45	2500	0.55	2650	2.25	2750	1.00	G1898	800	0.00	1150	0.55	2550	0.65	2650	2.15	2750	1.00
G3910	850	0.00	1250	0.45	2500	0.60	2600	2.15	2650	1.00	G1899	800	0.00	1150	0.55	2550	0.65	2650	2.20	2700	1.00
G3911	850	0.00	1250	0.45	2500	0.60	2600	2.15	2700	1.00	G1900	800	0.00	1150	0.55	2550	0.65	2650	2.20	2750	1.00
G3913	850	0.00	1250	0.45	2500	0.60	2600	2.20	2650	1.00	G1901	800	0.00	1150	0.55	2550	0.65	2650	2.25	2700	1.00
G3914	850	0.00	1250	0.45	2500	0.60	2600	2.20	2700	1.00	G1902	800	0.00	1150	0.55	2550	0.65	2650	2.25	2750	1.00
G3916	850	0.00	1250	0.45	2500	0.60	2600	2.25	2650	1.00	G1904	800	0.00	1150	0.55	2550	0.65	2700	2.15	2750	1.00
G3917	850	0.00	1250	0.45	2500	0.60	2600	2.25	2700	1.00	G1908	800	0.00	1150	0.55	2550	0.65	2700	2.25	2750	1.00
G3919	850	0.00	1250	0.45	2500	0.60	2650	2.15	2700	1.00	G1909	800	0.00	1150	0.55	2600	0.55	2650	2.15	2700	1.00
G3920	850	0.00	1250	0.45	2500	0.60	2650	2.15	2750	1.00	G1910	800	0.00	1150	0.55	2600	0.55	2650	2.15	2750	1.00
G3921	850	0.00	1250	0.45	2500	0.60	2650	2.20	2700	1.00	G1911	800	0.00	1150	0.55	2600	0.55	2650	2.20	2700	1.00
G3922	850	0.00	1250	0.45	2500	0.60	2650	2.20	2750	1.00	G1912	800	0.00	1150	0.55	2600	0.55	2650	2.20	2750	1.00
G3923	850	0.00	1250	0.45	2500	0.60	2650	2.25	2700	1.00	G1913	800	0.00	1150	0.55	2600	0.55	2650	2.25	2700	1.00
G3924	850	0.00	1250	0.45	2500	0.60	2650	2.25	2750	1.00	G1914	800	0.00	1150	0.55	2600	0.55	2650	2.25	2750	1.00
G3931	850	0.00	1250	0.45	2500	0.65	2600	2.15	2650	1.00	G1916	800	0.00	1150	0.55	2600	0.55	2700	2.15	2750	1.00
G3932	850	0.00	1250	0.45	2500	0.65	2600	2.15	2700	1.00	G1920	800	0.00	1150	0.55	2600	0.55	2700	2.25	2750	1.00
G3934	850	0.00	1250	0.45	2500	0.65	2600	2.20	2650	1.00	G1921	800	0.00	1150	0.55	2600	0.60	2650	2.15	2700	1.00
G3935	850	0.00	1250	0.45	2500	0.65	2600	2.20	2700	1.00	G1922	800	0.00	1150	0.55	2600	0.60	2650	2.15	2750	1.00
G3937	850	0.00	1250	0.45	2500	0.65	2600	2.25	2650	1.00	G1923	800	0.00	1150	0.55	2600	0.60	2650	2.20	2700	1.00
G3938	850	0.00	1250	0.45	2500	0.65	2600	2.25	2700	1.00	G1924	800	0.00	1150	0.55	2600	0.60	2650	2.20	2750	1.00
G3940	850	0.00	1250	0.45	2500	0.65	2650	2.15	2700	1.00	G1925	800	0.00	1150	0.55	2600	0.60	2650	2.25	2700	1.00
G3941	850	0.00	1250	0.45	2500	0.65	2650	2.15	2750	1.00	G1926	800	0.00	1150	0.55	2600	0.60	2650	2.25	2750	1.00
G3942	850	0.00	1250	0.45	2500	0.65	2650	2.20	2700	1.00	G1928	800	0.00	1150	0.55	2600	0.60	2700	2.15	2750	1.00
G3944	850	0.00	1250	0.45	2500	0.65	2650	2.25	2700	1.00	G1932	800	0.00	1150	0.55	2600	0.60	2700	2.25	2750	1.00
G3952	850	0.00	1250	0.45	2550	0.55	2600	2.15	2650	1.00	G1933	800	0.00	1150	0.55	2600	0.65	2650	2.15	2700	1.00
G3953	850	0.00	1250	0.45	2550	0.55	2600	2.15	2700	1.00	G1934	800	0.00	1150	0.55	2600	0.65	2650	2.15	2750	1.00
G3955	850	0.00	1250	0.45	2550	0.55	2600	2.20	2650	1.00	G1935	800	0.00	1150	0.55	2600	0.65	2650	2.20	2700	1.00
G3958	850	0.00	1250	0.45	2550	0.55	2600	2.25	2650	1.00	G1936	800	0.00	1150	0.55	2600	0.65	2650	2.20	2750	1.00
G3959	850	0.00	1250	0.45	2550	0.55	2600	2.25	2700	1.00	G1937	800	0.00	1150	0.55	2600	0.65	2650	2.25	2700	1.00
G3961	850	0.00	1250	0.45	2550	0.55	2650	2.15	2700	1.00	G1938	800	0.00	1150	0.55	2600	0.65	2650	2.25	2750	1.00
G3962	850	0.00	1250	0.45	2550	0.55	2650	2.15	2750	1.00	G1940	800	0.00	1150	0.55	2600	0.65	2700	2.15	2750	1.00
G3963	850	0.00	1250	0.45	2550	0.55	2650	2.20	2700	1.00	G1944	800	0.00	1150	0.55	2600	0.65	2700	2.25	2750	1.00
G3964	850	0.00	1250	0.45	2550	0.55	2650	2.20	2750	1.00	G1945	800	0.00	1200	0.45	2500	0.55	2600	2.15	2650	1.00
G3965	850	0.00	1250	0.45	2550	0.55	2650	2.25	2700	1.00	G1946	800	0.00	1200	0.45	2500	0.55	2600	2.15	2700	1.00
G3966	850	0.00	1250	0.45	2550	0.55	2650	2.25	2750	1.00	G1948	800	0.00	1200	0.45	2500	0.55	2600	2.20	2650	1.00
G3968	850	0.00	1250	0.45	2550	0.55	2700	2.15	2750	1.00	G1949	800	0.00	1200	0.45	2500	0.55	2600	2.20	2700	1.00
G3970	850	0.00	1250	0.45	2550	0.55	2700	2.20	2750	1.00	G1951	800	0.00	1200	0.45	2500	0.55	2600	2.25	2650	1.00
G3972	850	0.00	1250	0.45	2550	0.55	2700	2.25	2750	1.00	G1952	800	0.00	1200	0.45	2500	0.55	2600	2.25	2700	1.00
G3973	850	0.00	1250	0.45	2550	0.60	2600	2.15	2650	1.00	G1954	800	0.00	1200	0.45	2500	0.55	2650	2.15	2700	1.00
G3976	850	0.00	1250	0.45	2550	0.60	2600	2.20	2650	1.00	G1955	800	0.00	1200	0.45	2500	0.55	2650	2.15	2750	1.00
G3977	850	0.00	1250	0.45	2550	0.60	2600	2.20	2700	1.00	G1956	800	0.00	1200	0.45	2500	0.55	2650	2.20	2700	1.00
G3979	850	0.00	1250	0.45	2550	0.60	2600	2.25	2650	1.00	G1957	800	0.00	1200	0.45	2500	0.55	2650	2.20	2750	1.00
G3982	850	0.00	1250	0.45	2550	0.60	2650	2.15	2700	1.00	G1958	800	0.00	1200	0.45	2500	0.55	2650	2.25	2700	1.00
G3983	850	0.00	1250	0.45	2550	0.60	2650	2.15	2750	1.00	G1966	800	0.00	1200	0.45	2500	0.60	2600	2.15	2650	1.00
G3984	850	0.00	1250	0.45	2550	0.60	2650	2.20	2700	1.00	G1969	800	0.00	1200	0.45	2500	0.60	2600	2.20	2650	1.00
G3985	850	0.00	1250	0.45	2550	0.60	2650	2.20	2750	1.00	G1970	800	0.00	1200	0.45	2500	0.60	2600	2.20	2700	1.00
G3986	850	0.00	1250	0.45	2550	0.60	2650	2.25	2700	1.00	G1972	800	0.00	1200	0.45	2500	0.60	2600	2.25	2650	1.00
G3987	850	0.00	1250	0.45	2550	0.60	2650	2.25	2750	1.00	G1973	800	0.00	1200	0.45	2500	0.60	2600	2.25	2700	1.00
G3989	850	0.00	1250	0.45	2550	0.60	2700	2.15	2750	1.00	G1975	800	0.00	1200	0.45	2500	0.60	2650	2.15	2700	1.00
G3991	850	0.00	1250	0.45	2550	0.60	2700	2.20	2750	1.00	G1976	800	0.00	1200	0.45	2500	0.60	2650	2.15	2750	1.00
G3993	850	0.00	1250	0.45	2550	0.60	2700	2.25	2750	1.00	G1977	800	0.00	1200	0.45	2500	0.60	2650	2.20	2700	1.00
G3994	850	0.00	1250	0.45	2550	0.65	2600	2.15	2650	1.00	G1978	800	0.00	1200	0.45	2500	0.60	2650	2.20	2750	1.00
G3995	850	0.00	1250	0.45	2550	0.65	2600	2.15	2700	1.00	G1979	800	0.00	1200	0.45	2500	0.60	2650	2.25	2700	1.00
G3997	850	0.00	1250	0.45	2550	0.65	2600	2.20	2650	1.00	G1980	800	0.00	1200	0.45	2500	0.60	2650	2.25	2750	1.00
G3998	850	0.00	1250	0.45	2550	0.65	2600	2.20	2700	1.00	G1987	800	0.00	1200	0.45	2500	0.65	2600	2.15	2650	1.00
G4000	850	0.00	1250	0.45	2550	0.65	2600	2.25	2650	1.00	G1988	800	0.00	1200	0.45	2500	0.65	2600	2.15	2700	1.00
G4001	850	0.00	1250	0.45	2550	0.65	2600	2.25	2700	1.00	G1990	800	0.00	1200	0.45	2500	0.65	2600	2.20	2650	1.00
G4003	850	0.00	1250	0.45	2550	0.65	2650	2.15	2700	1.00	G1991	800	0.00	1200	0.45	2500	0.65	2600	2.20	2700	1.00
G4004	850	0.00	1250	0.45	2550	0.65	2650	2.15	2750	1.00	G1993	800	0.00	1200	0.45	2500	0.65	2600	2.25	2650	1.00
G4005	850	0.00	1250	0.45	2550	0.65	2650	2.20	2700	1.00	G1994	800	0.00	1200	0.45	2500	0.65	2600	2.25	2700	1.00
G4006	850	0.00	1250	0.45	2550	0.65	2650	2.20	2750	1.00	G1996	800	0.00	1200	0.45	2500	0.65	2650	2.15	2700	1.00
G4007	850	0.00	1250	0.45	2550	0.65	2650	2.25	2700	1.00	G1997	800	0.00	1200	0.45	2500	0.65	2650	2.15	2750	1.00
G4008	850	0.00	1250	0.45	2550	0.65	2650	2.25	2750	1.00	G1998	800	0.00	1200	0.45	2500	0.65	2650	2.20	2700	1.00
G4010	850	0.00	1250	0.45	2550	0.65	2700	2.15	2750	1.00	G1999	800	0.00	1200	0.45	2500	0.65	2650	2.20	2750	1.00
G4012	850	0.00	1250	0.45	2550	0.65	2700	2.20	2750	1.00	G2000	800	0.00	1200	0.45	2500	0.65	2650	2.25	2700	1.00
G4015	850	0.00	1250	0.45	2600	0.55	2650	2.15	2700	1.00	G2001	800	0.00	1200	0.45	2500	0.65	2650	2.25	2750	1.00
G4016	850	0.00	1250	0.45	2600	0.55	2650	2.15	2750	1.00	G2008	800	0.00	1200	0.45	2550	0.55	2600	2.15	2650	1.00
G4017	850	0.00	1250	0.45	2600	0.55	2650	2.20	2700	1.00	G2011	800	0.00	1200	0.45	2550	0.55	2600	2.20	2650	1.00
G4018	850	0.00	1250	0.45	2600	0.55	2650	2.20	2750	1.00	G2012	800	0.00	1200	0.45	2550	0.55	2600	2.20	2700	1.00
G4019	850	0.00	1250	0.45	2600	0.55	2650	2.25	2700	1.00	G2014	800	0.00	1200	0.45	2550	0.55	2600	2.25	2650	1.00
G4020	850	0.00	1250	0.45	2600	0.55	2650	2.25	2750	1.00	G2015	800	0.00	1200	0.45	2550	0.55	2600	2.25	2700	1.00
G4022	850	0.00	1250	0.45	2600	0.55	2700	2.15	2750	1.00	G2017	800	0.00	1200	0.45	2550	0.55	2650	2.15	2700	1.00
G4024	850	0.00	1250	0.45	2600	0.55	2700	2.20	2750	1.00	G2018	800	0.00	1200	0.45	2550	0.55	2650	2.15	2750	1.00
G4026	850	0.00	1250	0.45	2600	0.55	2700	2.25	2750	1.00	G2019	800	0.00	1200	0.45	2550	0.55	2650	2.20	2700	1.00
G4027	850	0.00	1250	0.45	2600	0.60	2650	2.15	2700	1.00	G2020	800	0.00	1200	0.45	2550	0.55	2650	2.20	2750	1.00
G4028	850	0.00	1250	0.45	2600	0.60	2650	2.15	2750	1.00	G2021	800	0.00	1200	0.45	2550	0.55	2650	2.25	2700	1.00
G4029	850	0.00	1250	0.45	2600	0.60	2650	2.20	2700	1.00	G2022	800	0.00	1200	0.45	2550	0.55	2650	2.25	2750	1.00
G4030	850	0.00	1250	0.45	2600	0.60	2650	2.20	2750	1.00	G2024	800	0.00	1200	0.45	2550	0.55	2700	2.15	2750	1.00
G4031	850	0.00	1250	0.45	2600	0.60	2650	2.25	2700	1.00	G2026	800	0.00	1200	0.45	2550	0.55	2700	2.20	2750	1.00

G4032	850	0.00	1250	0.45	2600	0.60	2650	2.25	2750	1.00
G4034	850	0.00	1250	0.45	2600	0.60	2700	2.15	2750	1.00
G4036	850	0.00	1250	0.45	2600	0.60	2700	2.20	2750	1.00
G4038	850	0.00	1250	0.45	2600	0.60	2700	2.25	2750	1.00
G4039	850	0.00	1250	0.45	2600	0.65	2650	2.15	2700	1.00
G4040	850	0.00	1250	0.45	2600	0.65	2650	2.15	2750	1.00
G4041	850	0.00	1250	0.45	2600	0.65	2650	2.20	2700	1.00
G4042	850	0.00	1250	0.45	2600	0.65	2650	2.20	2750	1.00
G4043	850	0.00	1250	0.45	2600	0.65	2650	2.25	2700	1.00
G4046	850	0.00	1250	0.45	2600	0.65	2700	2.15	2750	1.00
G4048	850	0.00	1250	0.45	2600	0.65	2700	2.20	2750	1.00
G4050	850	0.00	1250	0.45	2600	0.65	2700	2.25	2750	1.00
G4051	850	0.00	1250	0.50	2500	0.55	2600	2.15	2650	1.00
G4052	850	0.00	1250	0.50	2500	0.55	2600	2.15	2700	1.00
G4054	850	0.00	1250	0.50	2500	0.55	2600	2.20	2650	1.00
G4055	850	0.00	1250	0.50	2500	0.55	2600	2.20	2700	1.00
G4058	850	0.00	1250	0.50	2500	0.55	2600	2.25	2700	1.00
G4060	850	0.00	1250	0.50	2500	0.55	2650	2.15	2700	1.00
G4062	850	0.00	1250	0.50	2500	0.55	2650	2.20	2700	1.00
G4063	850	0.00	1250	0.50	2500	0.55	2650	2.20	2750	1.00
G4064	850	0.00	1250	0.50	2500	0.55	2650	2.25	2700	1.00
G4065	850	0.00	1250	0.50	2500	0.55	2650	2.25	2750	1.00
G4072	850	0.00	1250	0.50	2500	0.60	2600	2.15	2650	1.00
G4073	850	0.00	1250	0.50	2500	0.60	2600	2.15	2700	1.00
G4075	850	0.00	1250	0.50	2500	0.60	2600	2.20	2650	1.00
G4076	850	0.00	1250	0.50	2500	0.60	2600	2.20	2700	1.00
G4078	850	0.00	1250	0.50	2500	0.60	2600	2.25	2650	1.00
G4079	850	0.00	1250	0.50	2500	0.60	2600	2.25	2700	1.00
G3943	850	0.00	1250	0.45	2500	0.65	2650	2.20	2750	1.00
G4082	850	0.00	1250	0.50	2500	0.60	2650	2.15	2750	1.00
G4083	850	0.00	1250	0.50	2500	0.60	2650	2.20	2700	1.00
G4084	850	0.00	1250	0.50	2500	0.60	2650	2.20	2750	1.00
G4085	850	0.00	1250	0.50	2500	0.60	2650	2.25	2700	1.00
G4086	850	0.00	1250	0.50	2500	0.60	2650	2.25	2750	1.00
G4093	850	0.00	1250	0.50	2500	0.65	2600	2.15	2650	1.00
G4094	850	0.00	1250	0.50	2500	0.65	2600	2.15	2700	1.00
G4096	850	0.00	1250	0.50	2500	0.65	2600	2.20	2650	1.00
G4097	850	0.00	1250	0.50	2500	0.65	2600	2.20	2700	1.00
G4099	850	0.00	1250	0.50	2500	0.65	2600	2.25	2650	1.00
G4100	850	0.00	1250	0.50	2500	0.65	2600	2.25	2700	1.00
G4102	850	0.00	1250	0.50	2500	0.65	2650	2.15	2700	1.00
G4103	850	0.00	1250	0.50	2500	0.65	2650	2.15	2750	1.00
G4104	850	0.00	1250	0.50	2500	0.65	2650	2.20	2700	1.00
G4105	850	0.00	1250	0.50	2500	0.65	2650	2.20	2750	1.00
G4106	850	0.00	1250	0.50	2500	0.65	2650	2.25	2700	1.00
G4107	850	0.00	1250	0.50	2500	0.65	2650	2.25	2750	1.00
G4114	850	0.00	1250	0.50	2550	0.55	2600	2.15	2650	1.00
G4115	850	0.00	1250	0.50	2550	0.55	2600	2.15	2700	1.00
G4117	850	0.00	1250	0.50	2550	0.55	2600	2.20	2650	1.00
G4118	850	0.00	1250	0.50	2550	0.55	2600	2.20	2700	1.00
G4120	850	0.00	1250	0.50	2550	0.55	2600	2.25	2650	1.00
G4121	850	0.00	1250	0.50	2550	0.55	2600	2.25	2700	1.00
G4123	850	0.00	1250	0.50	2550	0.55	2650	2.15	2700	1.00
G4124	850	0.00	1250	0.50	2550	0.55	2650	2.15	2750	1.00
G4125	850	0.00	1250	0.50	2550	0.55	2650	2.20	2700	1.00
G4126	850	0.00	1250	0.50	2550	0.55	2650	2.20	2750	1.00
G4127	850	0.00	1250	0.50	2550	0.55	2650	2.25	2700	1.00
G4128	850	0.00	1250	0.50	2550	0.55	2650	2.25	2750	1.00
G4130	850	0.00	1250	0.50	2550	0.55	2700	2.15	2750	1.00
G4132	850	0.00	1250	0.50	2550	0.55	2700	2.20	2750	1.00
G4134	850	0.00	1250	0.50	2550	0.55	2700	2.25	2750	1.00
G4135	850	0.00	1250	0.50	2550	0.60	2600	2.15	2650	1.00
G4136	850	0.00	1250	0.50	2550	0.60	2600	2.15	2700	1.00
G4138	850	0.00	1250	0.50	2550	0.60	2600	2.20	2650	1.00
G4139	850	0.00	1250	0.50	2550	0.60	2600	2.20	2700	1.00
G4141	850	0.00	1250	0.50	2550	0.60	2600	2.25	2650	1.00
G4142	850	0.00	1250	0.50	2550	0.60	2600	2.25	2700	1.00
G4144	850	0.00	1250	0.50	2550	0.60	2650	2.15	2700	1.00
G4145	850	0.00	1250	0.50	2550	0.60	2650	2.15	2750	1.00
G4146	850	0.00	1250	0.50	2550	0.60	2650	2.20	2700	1.00
G4147	850	0.00	1250	0.50	2550	0.60	2650	2.20	2750	1.00
G4149	850	0.00	1250	0.50	2550	0.60	2650	2.25	2750	1.00
G4151	850	0.00	1250	0.50	2550	0.60	2700	2.15	2750	1.00
G4153	850	0.00	1250	0.50	2550	0.60	2700	2.20	2750	1.00
G4155	850	0.00	1250	0.50	2550	0.60	2700	2.25	2750	1.00
G4156	850	0.00	1250	0.50	2550	0.65	2600	2.15	2650	1.00
G4157	850	0.00	1250	0.50	2550	0.65	2600	2.15	2700	1.00
G4159	850	0.00	1250	0.50	2550	0.65	2600	2.20	2650	1.00
G4160	850	0.00	1250	0.50	2550	0.65	2600	2.20	2700	1.00
G4162	850	0.00	1250	0.50	2550	0.65	2600	2.25	2650	1.00
G4163	850	0.00	1250	0.50	2550	0.65	2600	2.25	2700	1.00
G4165	850	0.00	1250	0.50	2550	0.65	2650	2.15	2700	1.00
G4166	850	0.00	1250	0.50	2550	0.65	2650	2.15	2750	1.00
G4167	850	0.00	1250	0.50	2550	0.65	2650	2.20	2700	1.00
G4168	850	0.00	1250	0.50	2550	0.65	2650	2.20	2750	1.00
G4169	850	0.00	1250	0.50	2550	0.65	2650	2.25	2700	1.00
G4170	850	0.00	1250	0.50	2550	0.65	2650	2.25	2750	1.00
G4172	850	0.00	1250	0.50	2550	0.65	2700	2.15	2750	1.00
G4174	850	0.00	1250	0.50	2550	0.65	2700	2.20	2750	1.00
G4176	850	0.00	1250	0.50	2550	0.65	2700	2.25	2750	1.00
G4177	850	0.00	1250	0.50	2600	0.55	2650	2.15	2700	1.00
G4178	850	0.00	1250	0.50	2600	0.55	2650	2.15	2750	1.00
G4179	850	0.00	1250	0.50	2600	0.55	2650	2.20	2700	1.00
G4180	850	0.00	1250	0.50	2600	0.55	2650	2.20	2750	1.00
G4181	850	0.00	1250	0.50	2600	0.55	2650	2.25	2700	1.00
G4182	850	0.00	1250	0.50	2600	0.55	2650	2.25	2750	1.00
G4184	850	0.00	1250	0.50	2600	0.55	2700	2.15	2750	1.00
G4186	850	0.00	1250	0.50	2600	0.55	2700	2.20	2750	1.00
G4188	850	0.00	1250	0.50	2600	0.55	2700	2.25	2750	1.00
G4189	850	0.00	1250	0.50	2600	0.60	2650	2.15	2700	1.00
G4190	850	0.00	1250	0.50	2600	0.60	2650	2.15	2750	1.00
G4191	850	0.00	1250	0.50	2600	0.60	2650	2.20	2700	1.00
G4192	850	0.00	1250	0.50	2600	0.60	2650	2.20	2750	1.00
G4193	850	0.00	1250	0.50	2600	0.60	2650	2.25	2700	1.00
G4194	850	0.00	1250	0.50	2600	0.60	2650	2.25	2750	1.00
G4196	850	0.00	1250	0.50	2600	0.60	2700	2.15	2750	1.00
G4198	850	0.00	1250	0.50	2600	0.60	2700	2.20	2750	1.00
G4200	850	0.00	1250	0.50	2600	0.60	2700	2.25	2750	1.00
G4201	850	0.00	1250	0.50	2600	0.65	2650	2.15	2700	1.00
G2028	800	0.00	1200	0.45	2550	0.55	2700	2.25	2750	1.00
G2029	800	0.00	1200	0.45	2550	0.60	2600	2.15	2650	1.00
G2032	800	0.00	1200	0.45	2550	0.60	2600	2.20	2650	1.00
G2035	800	0.00	1200	0.45	2550	0.60	2600	2.25	2650	1.00
G2036	800	0.00	1200	0.45	2550	0.60	2600	2.25	2700	1.00
G2038	800	0.00	1200	0.45	2550	0.60	2650	2.15	2700	1.00
G2040	800	0.00	1200	0.45	2550	0.60	2650	2.20	2700	1.00
G2041	800	0.00	1200	0.45	2550	0.60	2650	2.20	2750	1.00
G2042	800	0.00	1200	0.45	2550	0.60	2650	2.25	2700	1.00
G2043	800	0.00	1200	0.45	2550	0.60	2650	2.25	2750	1.00
G2045	800	0.00	1200	0.45	2550	0.60	2700	2.15	2750	1.00
G2047	800	0.00	1200	0.45	2550	0.60	2700	2.20	2750	1.00
G2049	800	0.00	1200	0.45	2550	0.60	2700	2.25	2750	1.00
G2050	800	0.00	1200	0.45	2550	0.65	2600	2.15	2650	1.00
G2051	800	0.00	1200	0.45	2550	0.65	2600	2.15	2700	1.00
G2053	800	0.00	1200	0.45	2550	0.65	2600	2.20	2650	1.00
G2054	800	0.00	1200	0.45	2550	0.65	2600	2.20	2700	1.00
G2056	800	0.00	1200	0.45	2550	0.65	2600	2.25	2650	1.00
G2057	800	0.00	1200	0.45	2550	0.65	2600	2.25	2700	1.00
G2059	800	0.00	1200	0.45	2550	0.65	2650	2.15	2700	1.00
G2060	800	0.00	1200	0.45	2550	0.65	2650	2.15	2750	1.00
G2061	800	0.00	1200	0.45	2550	0.65	2650	2.20	2700	1.00
G2062	800	0.00	1200	0.45	2550	0.65	2650	2.20	2750	1.00
G2063	800	0.00	1200	0.45	2550	0.65	2650	2.25	2700	1.00
G2064	800	0.00	1200	0.45	2550	0.65	2650	2.25	2750	1.00
G2068	800	0.00	1200	0.45	2550	0.65	2700	2.20	2750	1.00
G2070	800	0.00	1200	0.45	2550	0.65	2700	2.25	2750	1.00
G2071	800	0.00	1200	0.45	2600	0.55	2650	2.15	2700	1.00
G2072	800	0.00	1200	0.45	2600	0.55	2650	2.15	2750	1.00
G2073	800	0.00	1200	0.45	2600	0.55	2650	2.20	2700	1.00
G2074	800	0.00	1200	0.45	2600	0.55	2650	2.20	2750	1.00
G2075	800	0.00	1200	0.45	2600	0.55	2650	2.25	2700	1.00
G2076	800	0.00	1200	0.45	2600	0.55	2650	2.25	2750	1.00
G2078	800	0.00	1200	0.45	2600	0.55	2700	2.15	2750	1.00
G2080	800	0.00	1200	0.45	2600	0.55	2700	2.20	2750	1.00
G2082	800	0.00	1200	0.45	2600	0.55	2700	2.25	2750	1.00
G2083	800	0.00	1200	0.45	2600	0.60	2650	2.15	2700	1.00
G2084	800	0.00	1200	0.45	2600	0.60	2650	2.15	2750	1.00
G2085	800	0.00	1200	0.45	2600	0.60	2650	2.20	2700	1.00
G2086	800	0.00	1200	0.45	2600	0.60	2650	2.20	2750	1.00
G2087	800	0.00	1200	0.45	2600	0.60	2650	2.25	2700	1.00
G2088	800	0.00	1200	0.45	2600	0.60	2650	2.25	2750	1.00
G2090	800	0.00	1200	0.45	2600	0.60	2700	2.15	2750	1.00
G2092	800	0.00	1200	0.45	2600	0.60	2700	2.20	2750	1.00
G2094	800	0.00	1200	0.45	2600	0.60	2700	2.25	2750	1.00
G2095	800	0.00	1200	0.45	2600	0.65	2650	2.15	2700	1.00
G2096	800	0.00	1200	0.45	2600	0.65	2650	2.15	2750	1.00
G2097	800	0.00	1200	0.45	2600	0.65	2650	2.20	2700	1.00
G2099	800	0.00	1200	0.45	2600	0.65	2650	2.25	2700	1.00
G2100	800	0.00	1200	0.45	2600	0.65	2650	2.25	2750	1.00
G2102	800	0.00	1200	0.45	2600	0.65	2700	2.15	2750	1.00
G2104	800	0.00	1200	0.45	2600	0.65	2700	2.20	2750	1.00
G2106	800	0.00	1200	0.45	2600	0.65	2700	2.25	2750	1.00
G2107	800	0.00	1200	0.50	2500	0.55	2600	2.15	2650	1.00
G2108	800	0.00	1200	0.50	2500	0.55	2600	2.15	2700	1.00
G2110	800	0.00	1200	0.50	2500	0.55	2600	2.20	2650	1.00
G2111	800	0.00	1200	0.50	2500	0.55	2600	2.20	2700	1.00
G2113	800	0.00	1200	0.50	2500	0.55	2600	2.25	2650	1.00
G2114	800	0.00	1200	0.50	2500	0.55	2600	2.25	2700	1.00
G2116	800	0.00	1200	0.50	2500	0.55	2650	2.15	2700	1.00
G2117	800	0.00	1200	0.50	2500	0.55	2650	2.15	2750	1.00
G2118	800	0.00	1200	0.50	2500	0.55	2650	2.20	2700	1.00
G2119	800	0.00	1200	0.50	2500	0.55	2650	2.20	2750	1.00
G2120	800	0.00	1200	0.50	2500	0.55	2650	2.25	2700	1.00
G2121	800	0.00	1200	0.50	2500	0.55	2650	2.25	2750	1.00
G2128	800	0.00	1200	0.50	2500	0.60	2600	2.15	2650	1.00
G2129	800	0.00	1200	0.50	2500	0.60	2600	2.15	2700	1.00
G2131	800	0.00	1200	0.50	2500	0.60	2600	2.20	2650	1.00
G2132	800	0.00	1200	0.50	2500	0.60	2600	2.20	2700	1.00
G2134	800	0.00	1200	0.50	2500	0.60	2600	2.25	2650	1.00
G2135	800	0.00	1200	0.50	2500	0.60	2600	2.25	2700	1.00
G2137	800	0.00	1200	0.50	2500	0.60	2650	2.15	2700	1.00
G2138	800	0.00	1200	0.50	2500	0.60	2650	2.15	2750	1.00
G2139	800	0.00	1200	0.50	2500	0.60	2650	2.20	2700	1.00
G2140	800	0.00	1200	0.50	2500	0.60	2650	2.20	2750	1.00
G2141	800	0.00	1200	0.50	2500	0.60	2650	2.25	2700	1.00
G2142	800	0.00	1200	0.50	2500	0.60	2650	2.25	2750	1.00
G2149	800	0.00	1200	0.50	2500	0.65	2600	2.15	2650	1.00
G2150	800	0.00	1200	0.50	2500	0.65	2600	2.15	2700	1.00
G2152	800	0.00	1200	0.50	2500	0.65	2600	2.20	2650	1.00
G2153	800	0.00	1200	0.50	2500	0.65	2600	2.20	2700	1.00
G2155	800	0.00	1200	0.50	2500	0.65	2600	2.25	2650	1.00
G2156	800	0.00	1200	0.50	2500	0.65	2600	2.25	2700	1.00
G2158	800	0.00	1200	0.50	2500	0.65	2650	2.15	2700	1.00
G2159	800	0.00	1200	0.50	2500	0.65	2650	2.15	2750	1.00
G2160	800	0.00	1200	0.50	2500	0.65	2650	2.20	2700	1.00
G2161	800	0.00	1200	0.50	2500	0.65	2650	2.20	2750	1.00
G2162	800	0.00	1200	0.50	2500	0.65	2650	2.25	2700	1.00
G2163	800	0.00	1200	0.50	2500	0.65	2650	2.25	2750	1.00
G2170	800	0.00	1200	0.50	2550	0.55	2600	2.15	2650	1.00
G2173	800	0.00	1200	0.50	2550	0.55	2600	2.20	2650	1.00
G2174	800	0.00	1200	0.50	2550	0.55	2600	2.20	2700	1.00
G2176	800	0.00	1200	0.50	2550	0.55	2600	2.25	2650	1.00
G2177	800	0.00	1200	0.50	2550	0.55	2600	2.25	2700	1.00
G2179	800	0.00	1200	0.50	2550	0.55	2650	2.15	2700	1.00
G2180	800	0.00	1200	0.50	2550	0.55	2650	2.15	2750	1.00
G2181	800	0.00	1200	0.50	2550	0.55	2650	2.20	2700	1.00
G2182	800	0.00	1200	0.50	2550	0.55	2650	2.20	2750	1.00
G2183	800	0.00	1200	0.50	2550	0.55	2650	2.25	2700	1.00
G2184	800	0.00	1200	0.50	2550	0.55	2650	2.25	2750	1.00
G2186	800	0.00	1200	0.50	2550	0.55	2700	2.15	2750	1.00
G2188	800	0.00	1200	0.50	2550	0.55	2700	2.20	2750	1.00
G2190	800	0.00	1200	0.50	2550	0.55	2700	2.25	2750	1.00
G2191	800	0.00	1200	0.50	2550	0.60	2600	2.15	2650	1.00
G2192	800	0.00	1200	0.50	2550	0.60	2600	2.15	2700	1.00
G2194	800	0.00	1200	0.50	2550	0.60	2600	2.20	2650	1.00
G2197	800	0.00	1200	0.50	2550	0.60	2600	2.25	2650	1.00
G2198	800	0.00	1200	0.50	2550	0.60	2600	2.25	2700	1.00
G2200	800	0.00	1200	0.50	2550	0.60	2650	2.15	2700	1.00

G4202	850	0.00	1250	0.50	2600	0.65	2650	2.15	2750	1.00	G2201	800	0.00	1200	0.50	2550	0.60	2650	2.15	2750	1.00
G4203	850	0.00	1250	0.50	2600	0.65	2650	2.20	2700	1.00	G2203	800	0.00	1200	0.50	2550	0.60	2650	2.20	2750	1.00
G4204	850	0.00	1250	0.50	2600	0.65	2650	2.20	2750	1.00	G2204	800	0.00	1200	0.50	2550	0.60	2650	2.25	2700	1.00
G4205	850	0.00	1250	0.50	2600	0.65	2650	2.25	2700	1.00	G2205	800	0.00	1200	0.50	2550	0.60	2650	2.25	2750	1.00
G4208	850	0.00	1250	0.50	2600	0.65	2700	2.15	2750	1.00	G2207	800	0.00	1200	0.50	2550	0.60	2700	2.15	2750	1.00
G4210	850	0.00	1250	0.50	2600	0.65	2700	2.20	2750	1.00	G2209	800	0.00	1200	0.50	2550	0.60	2700	2.20	2750	1.00
G4212	850	0.00	1250	0.50	2600	0.65	2700	2.25	2750	1.00	G2211	800	0.00	1200	0.50	2550	0.60	2700	2.25	2750	1.00
G4213	850	0.00	1250	0.55	2500	0.55	2600	2.15	2650	1.00	G2212	800	0.00	1200	0.50	2550	0.65	2600	2.15	2650	1.00
G4214	850	0.00	1250	0.55	2500	0.55	2600	2.15	2700	1.00	G2213	800	0.00	1200	0.50	2550	0.65	2600	2.15	2700	1.00
G4216	850	0.00	1250	0.55	2500	0.55	2600	2.20	2650	1.00	G2215	800	0.00	1200	0.50	2550	0.65	2600	2.20	2650	1.00
G4217	850	0.00	1250	0.55	2500	0.55	2600	2.20	2700	1.00	G2216	800	0.00	1200	0.50	2550	0.65	2600	2.20	2700	1.00
G4219	850	0.00	1250	0.55	2500	0.55	2600	2.25	2650	1.00	G2218	800	0.00	1200	0.50	2550	0.65	2600	2.25	2650	1.00
G4220	850	0.00	1250	0.55	2500	0.55	2600	2.25	2700	1.00	G2219	800	0.00	1200	0.50	2550	0.65	2600	2.25	2700	1.00
G4222	850	0.00	1250	0.55	2500	0.55	2650	2.15	2700	1.00	G2221	800	0.00	1200	0.50	2550	0.65	2650	2.15	2700	1.00
G4223	850	0.00	1250	0.55	2500	0.55	2650	2.15	2750	1.00	G2222	800	0.00	1200	0.50	2550	0.65	2650	2.15	2750	1.00
G4224	850	0.00	1250	0.55	2500	0.55	2650	2.20	2700	1.00	G2223	800	0.00	1200	0.50	2550	0.65	2650	2.20	2700	1.00
G4225	850	0.00	1250	0.55	2500	0.55	2650	2.20	2750	1.00	G2224	800	0.00	1200	0.50	2550	0.65	2650	2.20	2750	1.00
G4226	850	0.00	1250	0.55	2500	0.55	2650	2.25	2700	1.00	G2225	800	0.00	1200	0.50	2550	0.65	2650	2.25	2700	1.00
G4227	850	0.00	1250	0.55	2500	0.55	2650	2.25	2750	1.00	G2226	800	0.00	1200	0.50	2550	0.65	2650	2.25	2750	1.00
G4234	850	0.00	1250	0.55	2500	0.60	2600	2.15	2650	1.00	G2228	800	0.00	1200	0.50	2550	0.65	2700	2.15	2750	1.00
G3345	850	0.00	1150	0.55	2550	0.60	2700	2.25	2750	1.00	G2230	800	0.00	1200	0.50	2550	0.65	2700	2.20	2750	1.00
G4235	850	0.00	1250	0.55	2500	0.60	2600	2.15	2700	1.00	G2232	800	0.00	1200	0.50	2550	0.65	2700	2.25	2750	1.00
G3346	850	0.00	1150	0.55	2550	0.65	2600	2.15	2650	1.00	G2234	800	0.00	1200	0.50	2600	0.55	2650	2.15	2750	1.00
G3347	850	0.00	1150	0.55	2550	0.65	2600	2.15	2700	1.00	G2235	800	0.00	1200	0.50	2600	0.55	2650	2.20	2700	1.00
G3349	850	0.00	1150	0.55	2550	0.65	2600	2.20	2650	1.00	G2236	800	0.00	1200	0.50	2600	0.55	2650	2.20	2750	1.00
G3350	850	0.00	1150	0.55	2550	0.65	2600	2.20	2700	1.00	G2237	800	0.00	1200	0.50	2600	0.55	2650	2.25	2700	1.00
G3353	850	0.00	1150	0.55	2550	0.65	2600	2.25	2700	1.00	G2238	800	0.00	1200	0.50	2600	0.55	2650	2.25	2750	1.00
G3355	850	0.00	1150	0.55	2550	0.65	2650	2.15	2700	1.00	G2240	800	0.00	1200	0.50	2600	0.55	2700	2.15	2750	1.00
G3357	850	0.00	1150	0.55	2550	0.65	2650	2.20	2700	1.00	G2242	800	0.00	1200	0.50	2600	0.55	2700	2.20	2750	1.00
G3358	850	0.00	1150	0.55	2550	0.65	2650	2.20	2750	1.00	G2244	800	0.00	1200	0.50	2600	0.55	2700	2.25	2750	1.00
G3359	850	0.00	1150	0.55	2550	0.65	2650	2.25	2700	1.00	G2245	800	0.00	1200	0.50	2600	0.60	2650	2.15	2700	1.00
G3362	850	0.00	1150	0.55	2550	0.65	2700	2.15	2750	1.00	G2246	800	0.00	1200	0.50	2600	0.60	2650	2.15	2750	1.00
G3366	850	0.00	1150	0.55	2550	0.65	2700	2.25	2750	1.00	G2247	800	0.00	1200	0.50	2600	0.60	2650	2.20	2700	1.00
G3367	850	0.00	1150	0.55	2600	0.55	2650	2.15	2700	1.00	G2248	800	0.00	1200	0.50	2600	0.60	2650	2.20	2750	1.00
G3369	850	0.00	1150	0.55	2600	0.55	2650	2.20	2700	1.00	G2249	800	0.00	1200	0.50	2600	0.60	2650	2.25	2700	1.00
G3370	850	0.00	1150	0.55	2600	0.55	2650	2.20	2750	1.00	G2250	800	0.00	1200	0.50	2600	0.60	2650	2.25	2750	1.00
G3371	850	0.00	1150	0.55	2600	0.55	2650	2.25	2700	1.00	G2252	800	0.00	1200	0.50	2600	0.60	2700	2.15	2750	1.00
G3374	850	0.00	1150	0.55	2600	0.55	2700	2.15	2750	1.00	G2254	800	0.00	1200	0.50	2600	0.60	2700	2.20	2750	1.00
G3378	850	0.00	1150	0.55	2600	0.55	2700	2.25	2750	1.00	G2256	800	0.00	1200	0.50	2600	0.60	2700	2.25	2750	1.00
G3379	850	0.00	1150	0.55	2600	0.60	2650	2.15	2700	1.00	G2257	800	0.00	1200	0.50	2600	0.65	2650	2.15	2700	1.00
G3381	850	0.00	1150	0.55	2600	0.60	2650	2.20	2700	1.00	G2258	800	0.00	1200	0.50	2600	0.65	2650	2.15	2750	1.00
G3382	850	0.00	1150	0.55	2600	0.60	2650	2.20	2750	1.00	G2259	800	0.00	1200	0.50	2600	0.65	2650	2.20	2700	1.00
G3383	850	0.00	1150	0.55	2600	0.60	2650	2.25	2700	1.00	G2260	800	0.00	1200	0.50	2600	0.65	2650	2.20	2750	1.00
G4237	850	0.00	1250	0.55	2500	0.60	2600	2.20	2650	1.00	G2261	800	0.00	1200	0.50	2600	0.65	2650	2.25	2700	1.00
G4238	850	0.00	1250	0.55	2500	0.60	2600	2.20	2700	1.00	G2262	800	0.00	1200	0.50	2600	0.65	2650	2.25	2750	1.00
G4240	850	0.00	1250	0.55	2500	0.60	2600	2.25	2650	1.00	G2264	800	0.00	1200	0.50	2600	0.65	2700	2.15	2750	1.00
G4243	850	0.00	1250	0.55	2500	0.60	2650	2.15	2700	1.00	G2266	800	0.00	1200	0.50	2600	0.65	2700	2.20	2750	1.00
G4244	850	0.00	1250	0.55	2500	0.60	2650	2.15	2750	1.00	G2268	800	0.00	1200	0.50	2600	0.65	2700	2.25	2750	1.00
G4245	850	0.00	1250	0.55	2500	0.60	2650	2.20	2700	1.00	G2269	800	0.00	1200	0.55	2500	0.55	2600	2.15	2650	1.00
G4246	850	0.00	1250	0.55	2500	0.60	2650	2.20	2750	1.00	G2270	800	0.00	1200	0.55	2500	0.55	2600	2.15	2700	1.00
G4247	850	0.00	1250	0.55	2500	0.60	2650	2.25	2700	1.00	G2272	800	0.00	1200	0.55	2500	0.55	2600	2.20	2650	1.00
G4248	850	0.00	1250	0.55	2500	0.60	2650	2.25	2750	1.00	G2273	800	0.00	1200	0.55	2500	0.55	2600	2.20	2700	1.00
G4255	850	0.00	1250	0.55	2500	0.65	2600	2.15	2650	1.00	G2275	800	0.00	1200	0.55	2500	0.55	2600	2.25	2650	1.00
G4256	850	0.00	1250	0.55	2500	0.65	2600	2.15	2700	1.00	G2276	800	0.00	1200	0.55	2500	0.55	2600	2.25	2700	1.00
G4258	850	0.00	1250	0.55	2500	0.65	2600	2.20	2650	1.00	G2278	800	0.00	1200	0.55	2500	0.55	2650	2.15	2700	1.00
G4261	850	0.00	1250	0.55	2500	0.65	2600	2.25	2650	1.00	G2279	800	0.00	1200	0.55	2500	0.55	2650	2.15	2750	1.00
G4262	850	0.00	1250	0.55	2500	0.65	2600	2.25	2700	1.00	G2280	800	0.00	1200	0.55	2500	0.55	2650	2.20	2700	1.00
G4264	850	0.00	1250	0.55	2500	0.65	2650	2.15	2700	1.00	G2281	800	0.00	1200	0.55	2500	0.55	2650	2.20	2750	1.00
G4265	850	0.00	1250	0.55	2500	0.65	2650	2.15	2750	1.00	G2282	800	0.00	1200	0.55	2500	0.55	2650	2.25	2700	1.00
G4266	850	0.00	1250	0.55	2500	0.65	2650	2.20	2700	1.00	G2283	800	0.00	1200	0.55	2500	0.55	2650	2.25	2750	1.00
G4267	850	0.00	1250	0.55	2500	0.65	2650	2.20	2750	1.00	G2290	800	0.00	1200	0.55	2500	0.60	2600	2.15	2650	1.00
G4268	850	0.00	1250	0.55	2500	0.65	2650	2.25	2700	1.00	G2291	800	0.00	1200	0.55	2500	0.60	2600	2.15	2700	1.00
G4269	850	0.00	1250	0.55	2500	0.65	2650	2.25	2750	1.00	G2293	800	0.00	1200	0.55	2500	0.60	2600	2.20	2650	1.00
G4276	850	0.00	1250	0.55	2550	0.55	2600	2.15	2650	1.00	G2294	800	0.00	1200	0.55	2500	0.60	2600	2.20	2700	1.00
G4277	850	0.00	1250	0.55	2550	0.55	2600	2.15	2700	1.00	G2297	800	0.00	1200	0.55	2500	0.60	2600	2.25	2700	1.00
G4279	850	0.00	1250	0.55	2550	0.55	2600	2.20	2650	1.00	G2300	800	0.00	1200	0.55	2500	0.60	2650	2.15	2750	1.00
G4280	850	0.00	1250	0.55	2550	0.55	2600	2.20	2700	1.00	G2301	800	0.00	1200	0.55	2500	0.60	2650	2.20	2700	1.00
G4282	850	0.00	1250	0.55	2550	0.55	2600	2.25	2650	1.00	G2302	800	0.00	1200	0.55	2500	0.60	2650	2.20	2750	1.00
G4283	850	0.00	1250	0.55	2550	0.55	2600	2.25	2700	1.00	G2303	800	0.00	1200	0.55	2500	0.60	2650	2.25	2700	1.00
G4285	850	0.00	1250	0.55	2550	0.55	2650	2.15	2700	1.00	G2304	800	0.00	1200	0.55	2500	0.60	2650	2.25	2750	1.00
G4286	850	0.00	1250	0.55	2550	0.55	2650	2.15	2750	1.00	G2311	800	0.00	1200	0.55	2500	0.65	2600	2.15	2650	1.00
G4287	850	0.00	1250	0.55	2550	0.55	2650	2.20	2700	1.00	G2312	800	0.00	1200	0.55	2500	0.65	2600	2.15	2700	1.00
G4288	850	0.00	1250	0.55	2550	0.55	2650	2.20	2750	1.00	G2314	800	0.00	1200	0.55	2500	0.65	2600	2.20	2650	1.00
G4289	850	0.00	1250	0.55	2550	0.55	2650	2.25	2700	1.00	G2315	800	0.00	1200	0.55	2500	0.65	2600	2.20	2700	1.00
G4290	850	0.00	1250	0.55	2550	0.55	2650	2.25	2750	1.00	G2317	800	0.00	1200	0.55	2500	0.65	2600	2.25	2650	1.00
G4292	850	0.00	1250	0.55	2550	0.55	2700	2.15	2750	1.00	G2318	800	0.00	1200	0.55	2500	0.65	2600	2.25	2700	1.00
G4294	850	0.00	1250	0.55	2550	0.55	2700	2.20	2750	1.00	G2321	800	0.00	1200	0.55	2500	0.65	2650	2.15	2750	1.00
G4296	850	0.00	1250	0.55	2550	0.55	2700	2.25	2750	1.00	G2322	800	0.00	1200	0.55	2500	0.65	2650	2.20	2700	1.00
G4297	850	0.00	1250	0.55	2550	0.60	2600	2.15	2650	1.00	G2323	800	0.00	1200	0.55	2500	0.65	2650	2.20	2750	1.00
G4300	850	0.00	1250	0.55	2550	0.60	2600	2.20	2650	1.00	G2324	800	0.00	1200	0.55	2500	0.65	2650	2.25	2700	1.00
G4301	850	0.00	1250	0.55	2550	0.60	2600	2.20	2700	1.00	G2325	800	0.00	1200	0.55	2500	0.65	2650	2.25	2750	1.00
G4303	850	0.00	1250	0.55	2550	0.60	2600	2.25	2650	1.00	G2332	800	0.00	1200	0.55	2550	0.55	2600	2.15	2650	1.00
G4304	850	0.00	1250	0.55	2550	0.60	2600	2.25	2700	1.00	G2333	800	0.00	1200	0.55	2550	0.55	2600	2.15	2700	1.00
G4306	850	0.00	1250	0.55	2550	0.60	2650	2.15	2700	1.00	G2335	800	0.00	1200	0.55	2550	0.55	2600	2.20	2650	1.00
G4307	850	0.00	1250	0.55	2550	0.60	2650	2.15	2750	1.00	G2336	800	0.00	1200	0.55	2550	0.55	2600	2.20	2700	1.00
G4308	850	0.00	1250	0.55	2550	0.60	2650	2.20	2700	1.00	G2338	800	0.00	1200	0.55	2550	0.55	2600	2.25	2650	1.00
G4309	850	0.00	1250	0.55	2550	0.60	2650	2.20	2750	1.00	G2339	800	0.00	1200	0.55	2550	0.55	2600	2.25	2700	1.00
G4310	850	0.00	1250	0.55	2550	0.60	2650	2.25	2700	1.00	G2341	800	0.00	1200	0.55	2550	0.55	2650	2.15	2700	1.00
G4313	850	0.00	1250	0.55	2550	0.60	2700	2.15	2750	1.00	G2342	800	0.00	1200	0.55	2550	0.55	2650	2.15	2750	1.00
G4315	850	0.00	1250	0.55	2550	0.60	2700	2.20	2750	1.00	G2343	800	0.00	1200	0.55	2550	0.55	2650	2.20	2700	1.00
G4317	850	0.00	1250	0.55	2550	0.60	2700	2.25	2750	1.00	G2344	800	0.00	1200	0.55	2550	0.55	2650	2.20	2750	1.00
G4318	850	0.00	1250	0.55	2550	0.65	2600	2.15	2650	1.00	G2345	800	0.00	1200	0.55	2550	0.55	2650	2.25	2700	1.00
G4319	850	0.00	1250	0.55	2550	0.65	2600	2.15	2700	1.00	G2346	800	0.00	1200	0.55	2550	0.55	2650	2.25	2750	1.00
G4321	850	0.00	1250	0.55	2550	0.65	2600	2.20	2650	1.00	G2348	800	0.00	1200	0.55	2550	0.55	2700	2.15	2750	1.00
G4322	850	0.00	1250	0.55	2550	0.65	2600	2.20	2700	1.00	G2350	800	0.00	1200	0.55	2550	0.55	2700	2.20	2750	1.00
G4324	850	0.00	1250	0.55	2550	0.65	2600	2.25	2650	1.00	G2352	800	0.00	1200	0.55	2550	0.55	2700	2.25	2750	1.00
G4325	850	0.00	1250	0.55	2550	0.65	2600	2.25	2700	1.00	G2353	800	0.00	1200	0.55	2550	0.60	2600	2.15	2650	1.00
G4327	850	0.00	1250	0.55	2550	0.65	2650	2.15	2700	1.00	G2354	800	0.00	1200	0.55	2550	0.60	2600	2.15	2700	1.00
G4328	850	0.00	1250	0.55	2550	0.65	2650	2.15	2750	1.00	G2356	800	0.00	1200	0.55	2550	0.60	2600	2.20	2650	1.00
G4329	850	0.00	1250	0.55	2550	0.65	2650	2.20	2700	1.00	G2357	800	0.00	1200	0.55	2550	0.60	2600	2.20	2700	1.00
G4330	850	0.00	1250	0.55	2550	0.65	2650	2.20	2750	1.00	G2359	800	0.00	1200	0.55	2550	0.60	2600	2.25	2650	1.00
G4331	850	0.00	1250	0.55	2550	0.65	2650	2.25	2700	1.00	G2360	800	0.00	1200	0.55	2550	0.60	2600	2.25	2700	1.00
G4332	850	0.00	1250	0.55	2550	0.65	2650	2.25	2750	1.00	G2362	800	0.00	1200	0.55	2550	0.60	2650	2.15	2700	1.00
G4334	850	0.00	1250	0.55	2550	0.65	2700	2.15	2750	1.00	G2363	800	0.00	1200	0.55	2550	0.60	2650	2.15	2750	1.00
G4336	850	0.00	1250	0.55	2550	0.65	2700	2.20	2750	1.00	G2364	800	0.00	1200	0.55	2550	0.60	2650	2.20	2700	1.00
G4338	850	0.00	1250	0.55	2550	0.65	2700	2.25	2750	1.00	G2365	800	0.00	1200	0.55	2550	0.60	2650	2.20	2750	1.00
G4339	850	0.00	1250	0.55	2600	0.55	2650	2.15	2700	1.00	G2367	800	0.00	1200	0.55	2550	0.60	2650	2.25	2750	1.00
G4340	850	0.00	1250	0.55	2600	0.55	2650	2.15	2750	1.00	G2369	800	0.00	1200	0.55	2550	0.60	2700	2.15	2750	1.00
G4341	850	0.00	1250	0.55	2600	0.55	2650	2.20	2700	1.00	G2371	800	0.00	1200	0.55	2550	0.60	2700	2.20	2750	1.00

G4342	850	0.00	1250	0.55	2600	0.55	2650	2.20	2750	1.00
G4343	850	0.00	1250	0.55	2600	0.55	2650	2.25	2700	1.00
G4344	850	0.00	1250	0.55	2600	0.55	2650	2.25	2750	1.00
G4346	850	0.00	1250	0.55	2600	0.55	2700	2.15	2750	1.00
G4348	850	0.00	1250	0.55	2600	0.55	2700	2.20	2750	1.00
G4350	850	0.00	1250	0.55	2600	0.55	2700	2.25	2750	1.00
G4351	850	0.00	1250	0.55	2600	0.60	2650	2.15	2700	1.00
G4352	850	0.00	1250	0.55	2600	0.60	2650	2.15	2750	1.00
G4353	850	0.00	1250	0.55	2600	0.60	2650	2.20	2700	1.00
G4354	850	0.00	1250	0.55	2600	0.60	2650	2.20	2750	1.00
G4355	850	0.00	1250	0.55	2600	0.60	2650	2.25	2700	1.00
G4356	850	0.00	1250	0.55	2600	0.60	2650	2.25	2750	1.00
G4358	850	0.00	1250	0.55	2600	0.60	2700	2.15	2750	1.00
G4360	850	0.00	1250	0.55	2600	0.60	2700	2.20	2750	1.00
G4362	850	0.00	1250	0.55	2600	0.60	2700	2.25	2750	1.00
G4363	850	0.00	1250	0.55	2600	0.65	2650	2.15	2700	1.00
G4364	850	0.00	1250	0.55	2600	0.65	2650	2.15	2750	1.00
G4365	850	0.00	1250	0.55	2600	0.65	2650	2.20	2700	1.00
G4366	850	0.00	1250	0.55	2600	0.65	2650	2.20	2750	1.00
G4367	850	0.00	1250	0.55	2600	0.65	2650	2.25	2700	1.00
G4368	850	0.00	1250	0.55	2600	0.65	2650	2.25	2750	1.00
G4372	850	0.00	1250	0.55	2600	0.65	2700	2.20	2750	1.00
G4374	850	0.00	1250	0.55	2600	0.65	2700	2.25	2750	1.00
G3318	850	0.00	1150	0.55	2550	0.55	2650	2.25	2750	1.00
G3386	850	0.00	1150	0.55	2600	0.60	2700	2.15	2750	1.00
G3390	850	0.00	1150	0.55	2600	0.60	2700	2.25	2750	1.00
G3391	850	0.00	1150	0.55	2600	0.65	2650	2.15	2700	1.00
G3393	850	0.00	1150	0.55	2600	0.65	2650	2.20	2700	1.00
G3394	850	0.00	1150	0.55	2600	0.65	2650	2.20	2750	1.00
G3395	850	0.00	1150	0.55	2600	0.65	2650	2.25	2700	1.00
G3398	850	0.00	1150	0.55	2600	0.65	2700	2.15	2750	1.00
G3402	850	0.00	1150	0.55	2600	0.65	2700	2.25	2750	1.00
G3403	850	0.00	1200	0.45	2500	0.55	2600	2.15	2650	1.00
G3406	850	0.00	1200	0.45	2500	0.55	2600	2.20	2650	1.00
G3407	850	0.00	1200	0.45	2500	0.55	2600	2.20	2700	1.00
G3409	850	0.00	1200	0.45	2500	0.55	2600	2.25	2650	1.00
G3410	850	0.00	1200	0.45	2500	0.55	2600	2.25	2700	1.00
G3413	850	0.00	1200	0.45	2500	0.55	2650	2.15	2750	1.00
G3414	850	0.00	1200	0.45	2500	0.55	2650	2.20	2700	1.00
G3415	850	0.00	1200	0.45	2500	0.55	2650	2.20	2750	1.00
G3417	850	0.00	1200	0.45	2500	0.55	2650	2.25	2750	1.00
G3427	850	0.00	1200	0.45	2500	0.60	2600	2.20	2650	1.00
G3430	850	0.00	1200	0.45	2500	0.60	2600	2.25	2650	1.00
G3431	850	0.00	1200	0.45	2500	0.60	2600	2.25	2700	1.00
G3433	850	0.00	1200	0.45	2500	0.60	2650	2.15	2700	1.00
G3434	850	0.00	1200	0.45	2500	0.60	2650	2.15	2750	1.00
G3435	850	0.00	1200	0.45	2500	0.60	2650	2.20	2700	1.00
G3437	850	0.00	1200	0.45	2500	0.60	2650	2.25	2700	1.00
G3438	850	0.00	1200	0.45	2500	0.60	2650	2.25	2750	1.00
G3445	850	0.00	1200	0.45	2500	0.65	2600	2.15	2650	1.00
G3446	850	0.00	1200	0.45	2500	0.65	2600	2.15	2700	1.00
G3449	850	0.00	1200	0.45	2500	0.65	2600	2.20	2700	1.00
G3451	850	0.00	1200	0.45	2500	0.65	2600	2.25	2650	1.00
G3454	850	0.00	1200	0.45	2500	0.65	2650	2.15	2700	1.00
G3455	850	0.00	1200	0.45	2500	0.65	2650	2.15	2750	1.00
G3457	850	0.00	1200	0.45	2500	0.65	2650	2.20	2750	1.00
G3458	850	0.00	1200	0.45	2500	0.65	2650	2.25	2700	1.00
G3459	850	0.00	1200	0.45	2500	0.65	2650	2.25	2750	1.00
G3466	850	0.00	1200	0.45	2550	0.55	2600	2.15	2650	1.00
G3469	850	0.00	1200	0.45	2550	0.55	2600	2.20	2650	1.00
G3470	850	0.00	1200	0.45	2550	0.55	2600	2.20	2700	1.00
G3473	850	0.00	1200	0.45	2550	0.55	2600	2.25	2700	1.00
G3475	850	0.00	1200	0.45	2550	0.55	2650	2.15	2700	1.00
G2202	800	0.00	1200	0.50	2550	0.60	2650	2.20	2700	1.00
G3478	850	0.00	1200	0.45	2550	0.55	2650	2.20	2750	1.00
G3479	850	0.00	1200	0.45	2550	0.55	2650	2.25	2700	1.00
G3482	850	0.00	1200	0.45	2550	0.55	2700	2.15	2750	1.00
G3486	850	0.00	1200	0.45	2550	0.55	2700	2.25	2750	1.00
G3487	850	0.00	1200	0.45	2550	0.60	2600	2.15	2650	1.00
G3490	850	0.00	1200	0.45	2550	0.60	2600	2.20	2650	1.00
G3493	850	0.00	1200	0.45	2550	0.60	2600	2.25	2650	1.00
G3494	850	0.00	1200	0.45	2550	0.60	2600	2.25	2700	1.00
G3497	850	0.00	1200	0.45	2550	0.60	2650	2.15	2750	1.00
G3498	850	0.00	1200	0.45	2550	0.60	2650	2.20	2700	1.00
G3499	850	0.00	1200	0.45	2550	0.60	2650	2.20	2750	1.00
G3501	850	0.00	1200	0.45	2550	0.60	2650	2.25	2750	1.00
G3505	850	0.00	1200	0.45	2550	0.60	2700	2.20	2750	1.00
G3507	850	0.00	1200	0.45	2550	0.60	2700	2.25	2750	1.00
G3509	850	0.00	1200	0.45	2550	0.65	2600	2.15	2700	1.00
G3511	850	0.00	1200	0.45	2550	0.65	2600	2.20	2650	1.00
G3514	850	0.00	1200	0.45	2550	0.65	2600	2.25	2650	1.00
G3515	850	0.00	1200	0.45	2550	0.65	2600	2.25	2700	1.00
G3517	850	0.00	1200	0.45	2550	0.65	2650	2.15	2700	1.00
G3518	850	0.00	1200	0.45	2550	0.65	2650	2.15	2750	1.00
G3519	850	0.00	1200	0.45	2550	0.65	2650	2.20	2700	1.00
G3521	850	0.00	1200	0.45	2550	0.65	2650	2.25	2700	1.00
G3522	850	0.00	1200	0.45	2550	0.65	2650	2.25	2750	1.00
G3526	850	0.00	1200	0.45	2550	0.65	2700	2.20	2750	1.00
G3529	850	0.00	1200	0.45	2600	0.55	2650	2.15	2700	1.00
G3530	850	0.00	1200	0.45	2600	0.55	2650	2.15	2750	1.00
G3531	850	0.00	1200	0.45	2600	0.55	2650	2.20	2700	1.00
G3533	850	0.00	1200	0.45	2600	0.55	2650	2.25	2700	1.00
G3534	850	0.00	1200	0.45	2600	0.55	2650	2.25	2750	1.00
G3538	850	0.00	1200	0.45	2600	0.55	2700	2.20	2750	1.00
G3541	850	0.00	1200	0.45	2600	0.60	2650	2.15	2700	1.00
G3542	850	0.00	1200	0.45	2600	0.60	2650	2.15	2750	1.00
G3543	850	0.00	1200	0.45	2600	0.60	2650	2.20	2700	1.00
G3545	850	0.00	1200	0.45	2600	0.60	2650	2.25	2700	1.00
G3546	850	0.00	1200	0.45	2600	0.60	2650	2.25	2750	1.00
G3550	850	0.00	1200	0.45	2600	0.60	2700	2.20	2750	1.00
G3553	850	0.00	1200	0.45	2600	0.65	2650	2.15	2700	1.00
G3554	850	0.00	1200	0.45	2600	0.65	2650	2.15	2750	1.00
G3555	850	0.00	1200	0.45	2600	0.65	2650	2.20	2700	1.00
G3557	850	0.00	1200	0.45	2600	0.65	2650	2.25	2700	1.00
G3558	850	0.00	1200	0.45	2600	0.65	2650	2.25	2750	1.00
G3562	850	0.00	1200	0.45	2600	0.65	2700	2.20	2750	1.00
G3566	850	0.00	1200	0.50	2500	0.55	2600	2.15	2700	1.00
G3569	850	0.00	1200	0.50	2500	0.55	2600	2.20	2700	1.00
G3571	850	0.00	1200	0.50	2500	0.55	2600	2.25	2650	1.00
G2373	800	0.00	1200	0.55	2550	0.60	2700	2.25	2750	1.00
G2374	800	0.00	1200	0.55	2550	0.65	2600	2.15	2650	1.00
G2375	800	0.00	1200	0.55	2550	0.65	2600	2.15	2700	1.00
G2377	800	0.00	1200	0.55	2550	0.65	2600	2.20	2650	1.00
G2378	800	0.00	1200	0.55	2550	0.65	2600	2.20	2700	1.00
G2380	800	0.00	1200	0.55	2550	0.65	2600	2.25	2650	1.00
G2381	800	0.00	1200	0.55	2550	0.65	2600	2.25	2700	1.00
G2384	800	0.00	1200	0.55	2550	0.65	2650	2.15	2750	1.00
G2385	800	0.00	1200	0.55	2550	0.65	2650	2.20	2700	1.00
G2386	800	0.00	1200	0.55	2550	0.65	2650	2.20	2750	1.00
G2387	800	0.00	1200	0.55	2550	0.65	2650	2.25	2700	1.00
G2388	800	0.00	1200	0.55	2550	0.65	2650	2.25	2750	1.00
G2390	800	0.00	1200	0.55	2550	0.65	2700	2.15	2750	1.00
G2392	800	0.00	1200	0.55	2550	0.65	2700	2.20	2750	1.00
G2394	800	0.00	1200	0.55	2550	0.65	2700	2.25	2750	1.00
G2395	800	0.00	1200	0.55	2600	0.55	2650	2.15	2700	1.00
G2396	800	0.00	1200	0.55	2600	0.55	2650	2.15	2750	1.00
G2397	800	0.00	1200	0.55	2600	0.55	2650	2.20	2700	1.00
G2398	800	0.00	1200	0.55	2600	0.55	2650	2.20	2750	1.00
G2399	800	0.00	1200	0.55	2600	0.55	2650	2.25	2700	1.00
G2400	800	0.00	1200	0.55	2600	0.55	2650	2.25	2750	1.00
G2402	800	0.00	1200	0.55	2600	0.55	2700	2.15	2750	1.00
G2404	800	0.00	1200	0.55	2600	0.55	2700	2.20	2750	1.00
G2406	800	0.00	1200	0.55	2600	0.55	2700	2.25	2750	1.00
G2407	800	0.00	1200	0.55	2600	0.60	2650	2.15	2700	1.00
G2408	800	0.00	1200	0.55	2600	0.60	2650	2.15	2750	1.00
G2409	800	0.00	1200	0.55	2600	0.60	2650	2.20	2700	1.00
G2410	800	0.00	1200	0.55	2600	0.60	2650	2.20	2750	1.00
G2411	800	0.00	1200	0.55	2600	0.60	2650	2.25	2700	1.00
G2412	800	0.00	1200	0.55	2600	0.60	2650	2.25	2750	1.00
G2414	800	0.00	1200	0.55	2600	0.60	2700	2.15	2750	1.00
G2416	800	0.00	1200	0.55	2600	0.60	2700	2.20	2750	1.00
G2418	800	0.00	1200	0.55	2600	0.60	2700	2.25	2750	1.00
G2419	800	0.00	1200	0.55	2600	0.65	2650	2.15	2700	1.00
G2420	800	0.00	1200	0.55	2600	0.65	2650	2.15	2750	1.00
G2421	800	0.00	1200	0.55	2600	0.65	2650	2.20	2700	1.00
G2422	800	0.00	1200	0.55	2600	0.65	2650	2.20	2750	1.00
G2423	800	0.00	1200	0.55	2600	0.65	2650	2.25	2700	1.00
G2424	800	0.00	1200	0.55	2600	0.65	2650	2.25	2750	1.00
G2428	800	0.00	1200	0.55	2600	0.65	2700	2.20	2750	1.00
G2430	800	0.00	1200	0.55	2600	0.65	2700	2.25	2750	1.00
G2431	800	0.00	1250	0.45	2500	0.55	2600	2.15	2650	1.00
G2432	800	0.00	1250	0.45	2500	0.55	2600	2.15	2700	1.00
G2434	800	0.00	1250	0.45	2500	0.55	2600	2.20	2650	1.00
G2435	800	0.00	1250	0.45	2500	0.55	2600	2.20	2700	1.00
G2437	800	0.00	1250	0.45	2500	0.55	2600	2.25	2650	1.00
G2438	800	0.00	1250	0.45	2500	0.55	2600	2.25	2700	1.00
G2440	800	0.00	1250	0.45	2500	0.55	2650	2.15	2700	1.00
G2441	800	0.00	1250	0.45	2500	0.55	2650	2.15	2750	1.00
G2442	800	0.00	1250	0.45	2500	0.55	2650	2.20	2700	1.00
G2443	800	0.00	1250	0.45	2500	0.55	2650	2.20	2750	1.00
G2445	800	0.00	1250	0.45	2500	0.55	2650	2.25	2750	1.00
G2452	800	0.00	1250	0.45	2500	0.60	2600	2.15	2650	1.00
G2453	800	0.00	1250	0.45	2500	0.60	2600	2.15	2700	1.00
G2455	800	0.00	1250	0.45	2500	0.60	2600	2.20	2650	1.00
G2456	800	0.00	1250	0.45	2500	0.60	2600	2.20	2700	1.00
G2458	800	0.00	1250	0.45	2500	0.60	2600	2.25	2650	1.00
G2459	800	0.00	1250	0.45	2500	0.60	2600	2.25	2700	1.00
G2461	800	0.00	1250	0.45	2500	0.60	2650	2.15	2700	1.00
G2462	800	0.00	1250	0.45	2500	0.60	2650	2.15	2750	1.00
G2463	800	0.00	1250	0.45	2500	0.60	2650	2.20	2700	1.00
G2464	800	0.00	1250	0.45	2500	0.60	2650	2.20	2750	1.00
G2465	800	0.00	1250	0.45	2500	0.60	2650	2.25	2700	1.00
G2466	800	0.00	1250	0.45	2500	0.60	2650	2.25	2750	1.00
G2473	800	0.00	1250	0.45	2500	0.65	2600	2.15	2650	1.00
G2474	800	0.00	1250	0.45	2500	0.65	2600	2.15	2700	1.00
G2476	800	0.00	1250	0.45	2500	0.65	2600	2.20	2650	1.00
G2477	800	0.00	1250	0.45	2500	0.65	2600	2.20	2700	1.00
G2479	800	0.00	1250	0.45	2500	0.65	2600	2.25	2650	1.00
G2480	800	0.00	1250	0.45	2500	0.65	2600	2.25	2700	1.00
G2482	800	0.00	1250	0.45	2500	0.65	2650	2.15	2700	1.00
G2483	800	0.00	1250	0.45	2500	0.65	2650	2.15	2750	1.00
G2484	800	0.00	1250	0.45	2500	0.65	2650	2.20	2700	1.00
G2485	800	0.00	1250	0.45	2500	0.65	2650	2.20	2750	1.00
G2486	800	0.00	1250	0.45	2500	0.65	2650	2.25	2700	1.00
G2494	800	0.00	1250	0.45	2550	0.55	2600	2.15	2650	1.00
G2495	800	0.00	1250	0.45	2550	0.55	2600	2.15	2700	1.00
G2497	800	0.00	1250	0.45	2550	0.55	2600	2.20	2650	1.00
G2500	800	0.00	1250	0.45	2550	0.55	2600	2.25	2650	1.00
G2501	800	0.00	1250	0.45	2550	0.55	2600	2.25	2700	1.00
G2503	800	0.00	1250	0.45	2550	0.55	2650	2.15	2700	1.00
G2505	800	0.00	1250	0.45	2550	0.55	2650	2.20	2700	1.00
G2506	800	0.00	1250	0.45	2550	0.55	2650	2.20	2750	1.00
G2507	800	0.00	1250	0.45	2550	0.55	2650	2.25	2700	1.00
G2508	800	0.00	1250	0.45	2550	0.55	2650	2.25	2750	1.00
G2510	800	0.00	1250	0.45	2550	0.55	2700	2.15	2750	1.00
G2512	800	0.00	1250	0.45	2550	0.55	2700	2.20	2750	1.00
G2514	800	0.00	1250	0.45	2550	0.55	2700	2.25	2750	1.00
G2515	800	0.00	1250	0.45	2550	0.60	2600	2.15	2650	1.00
G2518	800	0.00	1250	0.45	2550	0.60	2600	2.20	2650	1.00
G2519	800	0.00	1250	0.45	2550	0.60	2600	2.20	2700	1.00
G2521	800	0.00	1250	0.45	2550	0.60	2600	2.25	2650	1.00
G2525	800	0.00	1250	0.45	2550	0.60	2650	2.15	2750	1.00
G2526	800	0.00	1250	0.45	2550	0.60	2650	2.20	2700	1.00
G2527	800	0.00	1250	0.45	2550	0.60	2650	2.20	2750	1.00
G2529	800	0.00	1250	0.45	2550	0.60	2650	2.25	2750	1.00
G2531	800	0.00	1250	0.45	2550	0.60	2700	2.15	2750	1.00
G2533	800	0.00	1250	0.45	2550	0.60	2700	2.20	2750	1.00
G2535	800	0.00	1250	0.45	2550	0.60	2700	2.25	2750	1.00
G2537	800	0.00	1250	0.45	2550	0.65	2600	2.15	2700	1.00
G2539	800	0.00	1250	0.45	2550	0.65	2600	2.20	2650	1.00
G2542	800	0.00	1250	0.45	2550	0.65	2600	2.25	2650	1.00
G2543	800	0.00	1250	0.45	2550	0.65	2600	2.25	2700	1.00
G2545	800	0.00	1250	0.45	2550	0.65	2650	2.15	2700	1.00
G2546	800	0.00	1250	0.45	2550	0.65	2650	2.15	2750	1.00
G2547	800	0.00	1250	0.45	2550	0.65	2650	2.20	2700	1.00
G2549	800	0.00	1250	0.45	2550	0.65	2650	2.25	2700	1.00
G2550	800	0.00	1250	0.45	2550	0.65	2650	2.25	2750	1.00
G2554	800	0.00	1250	0.45	2550	0.65	2700	2.20	2750	1.00

G3574	850	0.00	1200	0.50	2500	0.55	2650	2.15	2700	1.00
G3577	850	0.00	1200	0.50	2500	0.55	2650	2.20	2750	1.00
G3578	850	0.00	1200	0.50	2500	0.55	2650	2.25	2700	1.00
G3579	850	0.00	1200	0.50	2500	0.55	2650	2.25	2750	1.00
G3586	850	0.00	1200	0.50	2500	0.60	2600	2.15	2650	1.00
G3587	850	0.00	1200	0.50	2500	0.60	2600	2.15	2700	1.00
G3589	850	0.00	1200	0.50	2500	0.60	2600	2.20	2650	1.00
G3590	850	0.00	1200	0.50	2500	0.60	2600	2.20	2700	1.00
G3593	850	0.00	1200	0.50	2500	0.60	2600	2.25	2700	1.00
G3595	850	0.00	1200	0.50	2500	0.60	2650	2.15	2700	1.00
G3597	850	0.00	1200	0.50	2500	0.60	2650	2.20	2700	1.00
G3598	850	0.00	1200	0.50	2500	0.60	2650	2.20	2750	1.00
G3599	850	0.00	1200	0.50	2500	0.60	2650	2.25	2700	1.00
G3607	850	0.00	1200	0.50	2500	0.65	2600	2.15	2650	1.00
G3610	850	0.00	1200	0.50	2500	0.65	2600	2.20	2650	1.00
G3611	850	0.00	1200	0.50	2500	0.65	2600	2.20	2700	1.00
G3613	850	0.00	1200	0.50	2500	0.65	2600	2.25	2650	1.00
G3614	850	0.00	1200	0.50	2500	0.65	2600	2.25	2700	1.00
G3617	850	0.00	1200	0.50	2500	0.65	2650	2.15	2750	1.00
G3618	850	0.00	1200	0.50	2500	0.65	2650	2.20	2700	1.00
G3619	850	0.00	1200	0.50	2500	0.65	2650	2.20	2750	1.00
G3621	850	0.00	1200	0.50	2500	0.65	2650	2.25	2750	1.00
G3629	850	0.00	1200	0.50	2550	0.55	2600	2.15	2700	1.00
G3631	850	0.00	1200	0.50	2550	0.55	2600	2.20	2650	1.00
G3634	850	0.00	1200	0.50	2550	0.55	2600	2.25	2650	1.00
G3635	850	0.00	1200	0.50	2550	0.55	2600	2.25	2700	1.00
G3637	850	0.00	1200	0.50	2550	0.55	2650	2.15	2700	1.00
G3638	850	0.00	1200	0.50	2550	0.55	2650	2.15	2750	1.00
G3639	850	0.00	1200	0.50	2550	0.55	2650	2.20	2700	1.00
G3641	850	0.00	1200	0.50	2550	0.55	2650	2.25	2700	1.00
G3642	850	0.00	1200	0.50	2550	0.55	2650	2.25	2750	1.00
G3646	850	0.00	1200	0.50	2550	0.55	2700	2.20	2750	1.00
G3649	850	0.00	1200	0.50	2550	0.60	2600	2.15	2650	1.00
G3650	850	0.00	1200	0.50	2550	0.60	2600	2.15	2700	1.00
G3655	850	0.00	1200	0.50	2550	0.60	2600	2.25	2650	1.00
G3658	850	0.00	1200	0.50	2550	0.60	2650	2.15	2700	1.00
G3659	850	0.00	1200	0.50	2550	0.60	2650	2.15	2750	1.00
G3661	850	0.00	1200	0.50	2550	0.60	2650	2.20	2750	1.00
G3662	850	0.00	1200	0.50	2550	0.60	2650	2.25	2700	1.00
G3665	850	0.00	1200	0.50	2550	0.60	2700	2.15	2750	1.00
G3667	850	0.00	1200	0.50	2550	0.60	2700	2.20	2750	1.00
G3669	850	0.00	1200	0.50	2550	0.60	2700	2.25	2750	1.00
G3670	850	0.00	1200	0.50	2550	0.65	2600	2.15	2650	1.00
G3671	850	0.00	1200	0.50	2550	0.65	2600	2.15	2700	1.00
G3673	850	0.00	1200	0.50	2550	0.65	2600	2.20	2650	1.00
G3674	850	0.00	1200	0.50	2550	0.65	2600	2.20	2700	1.00
G3677	850	0.00	1200	0.50	2550	0.65	2600	2.25	2700	1.00
G3679	850	0.00	1200	0.50	2550	0.65	2650	2.15	2700	1.00
G3681	850	0.00	1200	0.50	2550	0.65	2650	2.20	2700	1.00
G3682	850	0.00	1200	0.50	2550	0.65	2650	2.20	2750	1.00
G3683	850	0.00	1200	0.50	2550	0.65	2650	2.25	2700	1.00
G3690	850	0.00	1200	0.50	2550	0.65	2700	2.25	2750	1.00
G3691	850	0.00	1200	0.50	2600	0.55	2650	2.15	2700	1.00
G3693	850	0.00	1200	0.50	2600	0.55	2650	2.20	2700	1.00
G3694	850	0.00	1200	0.50	2600	0.55	2650	2.20	2750	1.00
G3695	850	0.00	1200	0.50	2600	0.55	2650	2.25	2700	1.00
G3698	850	0.00	1200	0.50	2600	0.55	2700	2.15	2750	1.00
G3702	850	0.00	1200	0.50	2600	0.55	2700	2.25	2750	1.00
G3703	850	0.00	1200	0.50	2600	0.60	2650	2.15	2700	1.00
G3705	850	0.00	1200	0.50	2600	0.60	2650	2.20	2700	1.00
G3706	850	0.00	1200	0.50	2600	0.60	2650	2.20	2750	1.00
G3707	850	0.00	1200	0.50	2600	0.60	2650	2.25	2700	1.00
G3710	850	0.00	1200	0.50	2600	0.60	2700	2.15	2750	1.00
G3714	850	0.00	1200	0.50	2600	0.60	2700	2.25	2750	1.00
G3715	850	0.00	1200	0.50	2600	0.65	2650	2.15	2700	1.00
G3717	850	0.00	1200	0.50	2600	0.65	2650	2.20	2700	1.00
G3718	850	0.00	1200	0.50	2600	0.65	2650	2.20	2750	1.00
G3719	850	0.00	1200	0.50	2600	0.65	2650	2.25	2700	1.00
G3722	850	0.00	1200	0.50	2600	0.65	2700	2.15	2750	1.00
G3726	850	0.00	1200	0.50	2600	0.65	2700	2.25	2750	1.00
G3727	850	0.00	1200	0.55	2500	0.55	2600	2.15	2650	1.00
G3730	850	0.00	1200	0.55	2500	0.55	2600	2.20	2650	1.00
G3731	850	0.00	1200	0.55	2500	0.55	2600	2.20	2700	1.00
G3733	850	0.00	1200	0.55	2500	0.55	2600	2.25	2650	1.00
G3734	850	0.00	1200	0.55	2500	0.55	2600	2.25	2700	1.00
G3737	850	0.00	1200	0.55	2500	0.55	2650	2.15	2750	1.00
G3738	850	0.00	1200	0.55	2500	0.55	2650	2.20	2700	1.00
G3739	850	0.00	1200	0.55	2500	0.55	2650	2.20	2750	1.00
G3741	850	0.00	1200	0.55	2500	0.55	2650	2.25	2750	1.00
G3749	850	0.00	1200	0.55	2500	0.60	2600	2.15	2700	1.00
G3751	850	0.00	1200	0.55	2500	0.60	2600	2.20	2650	1.00
G3755	850	0.00	1200	0.55	2500	0.60	2600	2.25	2700	1.00
G3757	850	0.00	1200	0.55	2500	0.60	2650	2.15	2700	1.00
G3758	850	0.00	1200	0.55	2500	0.60	2650	2.15	2750	1.00
G3759	850	0.00	1200	0.55	2500	0.60	2650	2.20	2700	1.00
G3761	850	0.00	1200	0.55	2500	0.60	2650	2.25	2700	1.00
G3762	850	0.00	1200	0.55	2500	0.60	2650	2.25	2750	1.00
G3769	850	0.00	1200	0.55	2500	0.65	2600	2.15	2650	1.00
G3770	850	0.00	1200	0.55	2500	0.65	2600	2.15	2700	1.00
G3773	850	0.00	1200	0.55	2500	0.65	2600	2.20	2700	1.00
G3775	850	0.00	1200	0.55	2500	0.65	2600	2.25	2650	1.00
G3778	850	0.00	1200	0.55	2500	0.65	2650	2.15	2700	1.00
G3779	850	0.00	1200	0.55	2500	0.65	2650	2.15	2750	1.00
G3781	850	0.00	1200	0.55	2500	0.65	2650	2.20	2750	1.00
G3782	850	0.00	1200	0.55	2500	0.65	2650	2.25	2700	1.00
G3783	850	0.00	1200	0.55	2500	0.65	2650	2.25	2750	1.00
G3790	850	0.00	1200	0.55	2550	0.55	2600	2.15	2650	1.00
G3791	850	0.00	1200	0.55	2550	0.55	2600	2.15	2700	1.00
G3793	850	0.00	1200	0.55	2550	0.55	2600	2.20	2650	1.00
G3794	850	0.00	1200	0.55	2550	0.55	2600	2.20	2700	1.00
G3797	850	0.00	1200	0.55	2550	0.55	2600	2.25	2700	1.00
G3799	850	0.00	1200	0.55	2550	0.55	2650	2.15	2700	1.00
G3801	850	0.00	1200	0.55	2550	0.55	2650	2.20	2700	1.00
G3802	850	0.00	1200	0.55	2550	0.55	2650	2.20	2750	1.00
G3803	850	0.00	1200	0.55	2550	0.55	2650	2.25	2700	1.00
G3806	850	0.00	1200	0.55	2550	0.55	2700	2.15	2750	1.00
G3810	850	0.00	1200	0.55	2550	0.55	2700	2.25	2750	1.00
G3811	850	0.00	1200	0.55	2550	0.60	2600	2.15	2650	1.00
G3814	850	0.00	1200	0.55	2550	0.60	2600	2.20	2650	1.00
G2557	800	0.00	1250	0.45	2600	0.55	2650	2.15	2700	1.00
G2558	800	0.00	1250	0.45	2600	0.55	2650	2.15	2750	1.00
G2559	800	0.00	1250	0.45	2600	0.55	2650	2.20	2700	1.00
G2561	800	0.00	1250	0.45	2600	0.55	2650	2.25	2700	1.00
G2562	800	0.00	1250	0.45	2600	0.55	2650	2.25	2750	1.00
G2566	800	0.00	1250	0.45	2600	0.55	2700	2.20	2750	1.00
G2569	800	0.00	1250	0.45	2600	0.60	2650	2.15	2700	1.00
G2570	800	0.00	1250	0.45	2600	0.60	2650	2.15	2750	1.00
G2571	800	0.00	1250	0.45	2600	0.60	2650	2.20	2700	1.00
G2573	800	0.00	1250	0.45	2600	0.60	2650	2.25	2700	1.00
G2574	800	0.00	1250	0.45	2600	0.60	2650	2.25	2750	1.00
G2578	800	0.00	1250	0.45	2600	0.60	2700	2.20	2750	1.00
G2581	800	0.00	1250	0.45	2600	0.65	2650	2.15	2700	1.00
G2582	800	0.00	1250	0.45	2600	0.65	2650	2.15	2750	1.00
G2585	800	0.00	1250	0.45	2600	0.65	2650	2.25	2700	1.00
G2590	800	0.00	1250	0.45	2600	0.65	2700	2.20	2750	1.00
G2593	800	0.00	1250	0.50	2500	0.55	2600	2.15	2650	1.00
G2594	800	0.00	1250	0.50	2500	0.55	2600	2.15	2700	1.00
G2597	800	0.00	1250	0.50	2500	0.55	2600	2.20	2700	1.00
G2599	800	0.00	1250	0.50	2500	0.55	2600	2.25	2650	1.00
G2602	800	0.00	1250	0.50	2500	0.55	2650	2.15	2700	1.00
G2605	800	0.00	1250	0.50	2500	0.55	2650	2.20	2750	1.00
G2606	800	0.00	1250	0.50	2500	0.55	2650	2.25	2700	1.00
G2607	800	0.00	1250	0.50	2500	0.55	2650	2.25	2750	1.00
G2614	800	0.00	1250	0.50	2500	0.60	2600	2.15	2650	1.00
G2615	800	0.00	1250	0.50	2500	0.60	2600	2.15	2700	1.00
G2617	800	0.00	1250	0.50	2500	0.60	2600	2.20	2650	1.00
G2618	800	0.00	1250	0.50	2500	0.60	2600	2.20	2700	1.00
G2621	800	0.00	1250	0.50	2500	0.60	2600	2.25	2700	1.00
G2623	800	0.00	1250	0.50	2500	0.60	2650	2.15	2700	1.00
G2625	800	0.00	1250	0.50	2500	0.60	2650	2.20	2700	1.00
G2626	800	0.00	1250	0.50	2500	0.60	2650	2.20	2750	1.00
G2627	800	0.00	1250	0.50	2500	0.60	2650	2.25	2700	1.00
G2635	800	0.00	1250	0.50	2500	0.65	2600	2.15	2650	1.00
G2638	800	0.00	1250	0.50	2500	0.65	2600	2.20	2650	1.00
G2639	800	0.00	1250	0.50	2500	0.65	2600	2.20	2700	1.00
G2641	800	0.00	1250	0.50	2500	0.65	2600	2.25	2650	1.00
G2642	800	0.00	1250	0.50	2500	0.65	2600	2.25	2700	1.00
G2645	800	0.00	1250	0.50	2500	0.65	2650	2.15	2750	1.00
G2646	800	0.00	1250	0.50	2500	0.65	2650	2.20	2700	1.00
G2647	800	0.00	1250	0.50	2500	0.65	2650	2.20	2750	1.00
G2649	800	0.00	1250	0.50	2500	0.65	2650	2.25	2750	1.00
G2657	800	0.00	1250	0.50	2550	0.55	2600	2.15	2700	1.00
G2659	800	0.00	1250	0.50	2550	0.55	2600	2.20	2650	1.00
G2662	800	0.00	1250	0.50	2550	0.55	2600	2.25	2650	1.00
G2663	800	0.00	1250	0.50	2550	0.55	2600	2.25	2700	1.00
G2665	800	0.00	1250	0.50	2550	0.55	2650	2.15	2700	1.00
G2666	800	0.00	1250	0.50	2550	0.55	2650	2.15	2750	1.00
G2667	800	0.00	1250	0.50	2550	0.55	2650	2.20	2700	1.00
G2669	800	0.00	1250	0.50	2550	0.55	2650	2.25	2700	1.00
G2670	800	0.00	1250	0.50	2550	0.55	2650	2.25	2750	1.00
G2674	800	0.00	1250	0.50	2550	0.55	2700	2.20	2750	1.00
G2677	800	0.00	1250	0.50	2550	0.60	2600	2.15	2650	1.00
G2678	800	0.00	1250	0.50	2550	0.60	2600	2.15	2700	1.00
G2681	800	0.00	1250	0.50	2550	0.60	2600	2.20	2700	1.00
G2683	800	0.00	1250	0.50	2550	0.60	2600	2.25	2650	1.00
G2687	800	0.00	1250	0.50	2550	0.60	2650	2.15	2750	1.00
G2689	800	0.00	1250	0.50	2550	0.60	2650	2.20	2750	1.00
G2690	800	0.00	1250	0.50	2550	0.60	2650	2.25	2700	1.00
G2691	800	0.00	1250	0.50	2550	0.60	2650	2.25	2750	1.00
G2693	800	0.00	1250	0.50	2550	0.60	2700	2.15	2750	1.00
G2695	800	0.00	1250	0.50	2550	0.60	2700	2.20	2750	1.00
G2697	800	0.00	1250	0.50	2550	0.60	2700	2.25	2750	1.00
G2698	800	0.00	1250	0.50	2550	0.65	2600	2.15	2650	1.00
G2699	800	0.00	1250	0.50	2550	0.65	2600	2.15	2700	1.00
G2701	800	0.00	1250	0.50	2550	0.65	2600	2.20	2650	1.00
G2702	800	0.00	1250	0.50	2550	0.65	2600	2.20	2700	1.00
G2705	800	0.00	1250	0.50	2550	0.65	2600	2.25	2700	1.00
G2707	800	0.00	1250	0.50	2550	0.65	2650	2.15	2700	1.00
G2709	800	0.00	1250	0.50	2550	0.65	2650	2.20	2700	1.00
G2710	800	0.00	1250	0.50	2550	0.65	2650	2.20	2750	1.00
G2711	800	0.00	1250	0.50	2550	0.65	2650	2.25	2700	1.00
G2714	800	0.00	1250	0.50	2550	0.65	2700	2.15	2750	1.00
G2718	800	0.00	1250	0.50	2550	0.65	2700	2.25	2750	1.00
G2719	800	0.00	1250	0.50	2600	0.55	2650	2.15	2700	1.00
G2721	800	0.00	1250	0.50	2600	0.55	2650	2.20	2700	1.00
G2722	800	0.00	1250	0.50	2600	0.55	2650	2.20	2750	1.00
G2723	800	0.00	1250	0.50	2600	0.55	2650	2.25	2700	1.00
G2726	800	0.00	1250	0.50	2600	0.55	2700	2.15	2750	1.00
G2730	800	0.00	1250	0.50	2600	0.55	2700	2.25	2750	1.00
G2731	800	0.00	1250	0.50	2600	0.60	2650	2.15	2700	1.00
G2733	800	0.00	1250	0.50	2600	0.60	2650	2.20	2700	1.00
G2734	800	0.00	1250	0.50	2600	0.60	2650	2.20	2750	1.00
G2735	800	0.00	1250	0.50	2600	0.60	2650	2.25	2700	1.00
G2738	800	0.00	1250	0.50	2600	0.60	2700	2.15	2750	1.00
G2742	800	0.00	1250	0.50	2600	0.60	2700	2.25	2750	1.00
G2743	800	0.00	1250	0.50	2600	0.65	2650	2.15	2700	1.00
G2745	800	0.00	1250	0.50	2600	0.65	2650	2.20	2700	1.00
G2746	800	0.00	1250	0.50	2600	0.65	2650	2.20	2750	1.00
G2747	800	0.00	1250	0.50	2600	0.65	2650	2.25	2700	1.00
G2750	800	0.00	1250	0.50	2600	0.65	2700	2.15	2750	1.00
G2754	800	0.00	1250	0.50	2600	0.65	2700	2.25	2750	1.00
G2755	800	0.00	1250	0.55	2500	0.55	2600	2.15	2650	1.00
G2758	800	0.00	1250	0.55	2500	0.55	2600	2.20	2650	1.00
G2759	800	0.00	1250	0.55	2500	0.55	2600	2.20	2700	1.00
G2761	800	0.00	1250	0.55	2500	0.55	2600	2.25	2650	1.00
G2762	800	0.00	1250	0.55	2500	0.55	2600	2.25	2700	1.00
G2765	800	0.00	1250	0.55	2500	0.55	2650	2.15	2750	1.00
G2767	800	0.00	1250	0.55	2500	0.55	2650	2.20	2750	1.00
G2769	800	0.00	1250	0.55	2500	0.55	2650	2.25	2750	1.00
G2777	800	0.00	1250	0.55	2500	0.60	2600	2.15	2700	1.00
G2779	800	0.00	1250	0.55	2500	0.60	2600	2.20	2650	1.00
G2782	800	0.00	1250	0.55	2500	0.60	2600	2.25	2650	1.00
G2783	800	0.00	1250	0.55	2500	0.60	2600	2.25	2700	1.00
G2785	800	0.00	1250	0.55	2500	0.60	2650	2.15	2700	1.00
G2786	800	0.00	1250	0.55	2500	0.60	2650	2.15	2750	1.00
G2787	800	0.00	1250	0.55	2500	0.60	2650	2.20	2700	1.00
G2789	800	0.00	1250	0.55	2500	0.60	2650	2.25	2700	1.00
G2790	800	0.00	1250	0.55	2500	0.60	2650	2.25	2750	1.00

G3815	850	0.00	1200	0.55	2550	0.60	2600	2.20	2700	1.00
G3817	850	0.00	1200	0.55	2550	0.60	2600	2.25	2650	1.00
G3818	850	0.00	1200	0.55	2550	0.60	2600	2.25	2700	1.00
G3821	850	0.00	1200	0.55	2550	0.60	2650	2.15	2750	1.00
G3822	850	0.00	1200	0.55	2550	0.60	2650	2.20	2700	1.00
G3823	850	0.00	1200	0.55	2550	0.60	2650	2.20	2750	1.00
G3825	850	0.00	1200	0.55	2550	0.60	2650	2.25	2750	1.00
G3827	850	0.00	1200	0.55	2550	0.60	2700	2.15	2750	1.00
G3829	850	0.00	1200	0.55	2550	0.60	2700	2.20	2750	1.00
G3831	850	0.00	1200	0.55	2550	0.60	2700	2.25	2750	1.00
G3833	850	0.00	1200	0.55	2550	0.65	2600	2.15	2700	1.00
G3835	850	0.00	1200	0.55	2550	0.65	2600	2.20	2650	1.00
G3838	850	0.00	1200	0.55	2550	0.65	2600	2.25	2650	1.00
G3839	850	0.00	1200	0.55	2550	0.65	2600	2.25	2700	1.00
G3842	850	0.00	1200	0.55	2550	0.65	2650	2.15	2750	1.00
G3843	850	0.00	1200	0.55	2550	0.65	2650	2.20	2700	1.00
G3845	850	0.00	1200	0.55	2550	0.65	2650	2.25	2700	1.00
G3846	850	0.00	1200	0.55	2550	0.65	2650	2.25	2750	1.00
G3850	850	0.00	1200	0.55	2550	0.65	2700	2.20	2750	1.00
G3853	850	0.00	1200	0.55	2600	0.55	2650	2.15	2700	1.00
G3854	850	0.00	1200	0.55	2600	0.55	2650	2.15	2750	1.00
G3855	850	0.00	1200	0.55	2600	0.55	2650	2.20	2700	1.00
G3857	850	0.00	1200	0.55	2600	0.55	2650	2.25	2700	1.00
G3858	850	0.00	1200	0.55	2600	0.55	2650	2.25	2750	1.00
G3860	850	0.00	1200	0.55	2600	0.55	2700	2.15	2750	1.00
G3862	850	0.00	1200	0.55	2600	0.55	2700	2.20	2750	1.00
G3864	850	0.00	1200	0.55	2600	0.55	2700	2.25	2750	1.00
G3865	850	0.00	1200	0.55	2600	0.60	2650	2.15	2700	1.00
G3866	850	0.00	1200	0.55	2600	0.60	2650	2.15	2750	1.00
G3867	850	0.00	1200	0.55	2600	0.60	2650	2.20	2700	1.00
G3868	850	0.00	1200	0.55	2600	0.60	2650	2.20	2750	1.00
G3869	850	0.00	1200	0.55	2600	0.60	2650	2.25	2700	1.00
G3870	850	0.00	1200	0.55	2600	0.60	2650	2.25	2750	1.00
G3872	850	0.00	1200	0.55	2600	0.60	2700	2.15	2750	1.00
G3874	850	0.00	1200	0.55	2600	0.60	2700	2.20	2750	1.00
G3325	850	0.00	1150	0.55	2550	0.60	2600	2.15	2650	1.00
G3326	850	0.00	1150	0.55	2550	0.60	2600	2.15	2700	1.00
G3329	850	0.00	1150	0.55	2550	0.60	2600	2.20	2700	1.00
G3331	850	0.00	1150	0.55	2550	0.60	2600	2.25	2650	1.00
G3334	850	0.00	1150	0.55	2550	0.60	2650	2.15	2700	1.00
G3335	850	0.00	1150	0.55	2550	0.60	2650	2.15	2750	1.00
G3337	850	0.00	1150	0.55	2550	0.60	2650	2.20	2750	1.00
G3338	850	0.00	1150	0.55	2550	0.60	2650	2.25	2700	1.00
G3339	850	0.00	1150	0.55	2550	0.60	2650	2.25	2750	1.00
G3341	850	0.00	1150	0.55	2550	0.60	2700	2.15	2750	1.00
G2	750	0.00	1150	0.45	2500	0.55	2600	2.15	2700	1.00
G5	750	0.00	1150	0.45	2500	0.55	2600	2.20	2700	1.00
G7	750	0.00	1150	0.45	2500	0.55	2600	2.25	2650	1.00
G11	750	0.00	1150	0.45	2500	0.55	2650	2.15	2750	1.00
G13	750	0.00	1150	0.45	2500	0.55	2650	2.20	2750	1.00
G14	750	0.00	1150	0.45	2500	0.55	2650	2.25	2700	1.00
G15	750	0.00	1150	0.45	2500	0.55	2650	2.25	2750	1.00
G22	750	0.00	1150	0.45	2500	0.60	2600	2.15	2650	1.00
G23	750	0.00	1150	0.45	2500	0.60	2600	2.15	2700	1.00
G25	750	0.00	1150	0.45	2500	0.60	2600	2.20	2650	1.00
G26	750	0.00	1150	0.45	2500	0.60	2600	2.20	2700	1.00
G31	750	0.00	1150	0.45	2500	0.60	2650	2.15	2700	1.00
G33	750	0.00	1150	0.45	2500	0.60	2650	2.20	2700	1.00
G34	750	0.00	1150	0.45	2500	0.60	2650	2.20	2750	1.00
G35	750	0.00	1150	0.45	2500	0.60	2650	2.25	2700	1.00
G46	750	0.00	1150	0.45	2500	0.65	2600	2.20	2650	1.00
G47	750	0.00	1150	0.45	2500	0.65	2600	2.20	2700	1.00
G49	750	0.00	1150	0.45	2500	0.65	2600	2.25	2650	1.00
G50	750	0.00	1150	0.45	2500	0.65	2600	2.25	2700	1.00
G53	750	0.00	1150	0.45	2500	0.65	2650	2.15	2750	1.00
G54	750	0.00	1150	0.45	2500	0.65	2650	2.20	2700	1.00
G55	750	0.00	1150	0.45	2500	0.65	2650	2.20	2750	1.00
G57	750	0.00	1150	0.45	2500	0.65	2650	2.25	2750	1.00
G65	750	0.00	1150	0.45	2550	0.55	2600	2.15	2700	1.00
G67	750	0.00	1150	0.45	2550	0.55	2600	2.20	2650	1.00
G70	750	0.00	1150	0.45	2550	0.55	2600	2.25	2650	1.00
G71	750	0.00	1150	0.45	2550	0.55	2600	2.25	2700	1.00
G73	750	0.00	1150	0.45	2550	0.55	2650	2.15	2700	1.00
G74	750	0.00	1150	0.45	2550	0.55	2650	2.15	2750	1.00
G75	750	0.00	1150	0.45	2550	0.55	2650	2.20	2700	1.00
G77	750	0.00	1150	0.45	2550	0.55	2650	2.25	2700	1.00
G78	750	0.00	1150	0.45	2550	0.55	2650	2.25	2750	1.00
G82	750	0.00	1150	0.45	2550	0.55	2700	2.20	2750	1.00
G85	750	0.00	1150	0.45	2550	0.60	2600	2.15	2650	1.00
G86	750	0.00	1150	0.45	2550	0.60	2600	2.15	2700	1.00
G89	750	0.00	1150	0.45	2550	0.60	2600	2.20	2700	1.00
G91	750	0.00	1150	0.45	2550	0.60	2600	2.25	2650	1.00
G94	750	0.00	1150	0.45	2550	0.60	2650	2.15	2700	1.00
G95	750	0.00	1150	0.45	2550	0.60	2650	2.15	2750	1.00
G97	750	0.00	1150	0.45	2550	0.60	2650	2.20	2750	1.00
G98	750	0.00	1150	0.45	2550	0.60	2650	2.25	2700	1.00
G99	750	0.00	1150	0.45	2550	0.60	2650	2.25	2750	1.00
G101	750	0.00	1150	0.45	2550	0.60	2700	2.15	2750	1.00
G105	750	0.00	1150	0.45	2550	0.60	2700	2.25	2750	1.00
G106	750	0.00	1150	0.45	2550	0.65	2600	2.15	2650	1.00
G107	750	0.00	1150	0.45	2550	0.65	2600	2.15	2700	1.00
G109	750	0.00	1150	0.45	2550	0.65	2600	2.20	2650	1.00
G110	750	0.00	1150	0.45	2550	0.65	2600	2.20	2700	1.00
G113	750	0.00	1150	0.45	2550	0.65	2600	2.25	2700	1.00
G115	750	0.00	1150	0.45	2550	0.65	2650	2.15	2700	1.00
G117	750	0.00	1150	0.45	2550	0.65	2650	2.20	2700	1.00
G118	750	0.00	1150	0.45	2550	0.65	2650	2.20	2750	1.00
G119	750	0.00	1150	0.45	2550	0.65	2650	2.25	2700	1.00
G126	750	0.00	1150	0.45	2550	0.65	2700	2.25	2750	1.00
G127	750	0.00	1150	0.45	2600	0.55	2650	2.15	2700	1.00
G129	750	0.00	1150	0.45	2600	0.55	2650	2.20	2700	1.00
G130	750	0.00	1150	0.45	2600	0.55	2650	2.20	2750	1.00
G131	750	0.00	1150	0.45	2600	0.55	2650	2.25	2700	1.00
G134	750	0.00	1150	0.45	2600	0.55	2700	2.15	2750	1.00
G138	750	0.00	1150	0.45	2600	0.55	2700	2.25	2750	1.00
G139	750	0.00	1150	0.45	2600	0.60	2650	2.15	2700	1.00
G141	750	0.00	1150	0.45	2600	0.60	2650	2.20	2700	1.00
G142	750	0.00	1150	0.45	2600	0.60	2650	2.20	2750	1.00
G143	750	0.00	1150	0.45	2600	0.60	2650	2.25	2700	1.00
G2797	800	0.00	1250	0.55	2500	0.65	2600	2.15	2650	1.00
G2798	800	0.00	1250	0.55	2500	0.65	2600	2.15	2700	1.00
G2803	800	0.00	1250	0.55	2500	0.65	2600	2.25	2650	1.00
G2806	800	0.00	1250	0.55	2500	0.65	2650	2.15	2700	1.00
G2807	800	0.00	1250	0.55	2500	0.65	2650	2.15	2750	1.00
G2809	800	0.00	1250	0.55	2500	0.65	2650	2.20	2750	1.00
G2810	800	0.00	1250	0.55	2500	0.65	2650	2.25	2700	1.00
G2811	800	0.00	1250	0.55	2500	0.65	2650	2.25	2750	1.00
G2818	800	0.00	1250	0.55	2550	0.55	2600	2.15	2650	1.00
G2819	800	0.00	1250	0.55	2550	0.55	2600	2.15	2700	1.00
G2821	800	0.00	1250	0.55	2550	0.55	2600	2.20	2650	1.00
G2822	800	0.00	1250	0.55	2550	0.55	2600	2.20	2700	1.00
G2825	800	0.00	1250	0.55	2550	0.55	2600	2.25	2700	1.00
G2827	800	0.00	1250	0.55	2550	0.55	2650	2.15	2700	1.00
G2829	800	0.00	1250	0.55	2550	0.55	2650	2.20	2700	1.00
G2830	800	0.00	1250	0.55	2550	0.55	2650	2.20	2750	1.00
G2831	800	0.00	1250	0.55	2550	0.55	2650	2.25	2700	1.00
G2834	800	0.00	1250	0.55	2550	0.55	2700	2.15	2750	1.00
G2838	800	0.00	1250	0.55	2550	0.55	2700	2.25	2750	1.00
G2839	800	0.00	1250	0.55	2550	0.60	2600	2.15	2650	1.00
G2842	800	0.00	1250	0.55	2550	0.60	2600	2.20	2650	1.00
G2843	800	0.00	1250	0.55	2550	0.60	2600	2.20	2700	1.00
G2845	800	0.00	1250	0.55	2550	0.60	2600	2.25	2650	1.00
G2846	800	0.00	1250	0.55	2550	0.60	2600	2.25	2700	1.00
G2849	800	0.00	1250	0.55	2550	0.60	2650	2.15	2750	1.00
G2850	800	0.00	1250	0.55	2550	0.60	2650	2.20	2700	1.00
G2851	800	0.00	1250	0.55	2550	0.60	2650	2.20	2750	1.00
G2853	800	0.00	1250	0.55	2550	0.60	2650	2.25	2750	1.00
G2855	800	0.00	1250	0.55	2550	0.60	2700	2.15	2750	1.00
G2857	800	0.00	1250	0.55	2550	0.60	2700	2.20	2750	1.00
G2859	800	0.00	1250	0.55	2550	0.60	2700	2.25	2750	1.00
G2863	800	0.00	1250	0.55	2550	0.65	2600	2.20	2650	1.00
G2866	800	0.00	1250	0.55	2550	0.65	2600	2.25	2650	1.00
G2867	800	0.00	1250	0.55	2550	0.65	2600	2.25	2700	1.00
G2869	800	0.00	1250	0.55	2550	0.65	2650	2.15	2700	1.00
G2870	800	0.00	1250	0.55	2550	0.65	2650	2.15	2750	1.00
G2871	800	0.00	1250	0.55	2550	0.65	2650	2.20	2700	1.00
G2873	800	0.00	1250	0.55	2550	0.65	2650	2.25	2700	1.00
G2874	800	0.00	1250	0.55	2550	0.65	2650	2.25	2750	1.00
G2878	800	0.00	1250	0.55	2550	0.65	2700	2.20	2750	1.00
G2881	800	0.00	1250	0.55	2600	0.55	2650	2.15	2700	1.00
G2882	800	0.00	1250	0.55	2600	0.55	2650	2.15	2750	1.00
G2883	800	0.00	1250	0.55	2600	0.55	2650	2.20	2700	1.00
G2885	800	0.00	1250	0.55	2600	0.55	2650	2.25	2700	1.00
G2886	800	0.00	1250	0.55	2600	0.55	2650	2.25	2750	1.00
G2890	800	0.00	1250	0.55	2600	0.55	2700	2.20	2750	1.00
G2893	800	0.00	1250	0.55	2600	0.60	2650	2.15	2700	1.00
G2894	800	0.00	1250	0.55	2600	0.60	2650	2.15	2750	1.00
G2895	800	0.00	1250	0.55	2600	0.60	2650	2.20	2700	1.00
G2897	800	0.00	1250	0.55	2600	0.60	2650	2.25	2700	1.00
G2898	800	0.00	1250	0.55	2600	0.60	2650	2.25	2750	1.00
G2902	800	0.00	1250	0.55	2600	0.60	2700	2.20	2750	1.00
G2905	800	0.00	1250	0.55	2600	0.65	2650	2.15	2700	1.00
G2906	800	0.00	1250	0.55	2600	0.65	2650	2.15	2750	1.00
G2907	800	0.00	1250	0.55	2600	0.65	2650	2.20	2700	1.00
G2909	800	0.00	1250	0.55	2600	0.65	2650	2.25	2700	1.00
G2910	800	0.00	1250	0.55	2600	0.65	2650	2.25	2750	1.00
G2914	800	0.00	1250	0.55	2600	0.65	2700	2.20	2750	1.00
G2918	850	0.00	1150	0.45	2500	0.55	2600	2.15	2700	1.00
G2921	850	0.00	1150	0.45	2500	0.55	2600	2.20	2700	1.00
G2923	850	0.00	1150	0.45	2500	0.55	2600	2.25	2650	1.00
G2926	850	0.00	1150	0.45	2500	0.55	2650	2.15	2700	1.00
G2927	850	0.00	1150	0.45	2500	0.55	2650	2.15	2750	1.00
G2929	850	0.00	1150	0.45	2500	0.55	2650	2.20	2750	1.00
G2930	850	0.00	1150	0.45	2500	0.55	2650	2.25	2700	1.00
G2931	850	0.00	1150	0.45	2500	0.55	2650	2.25	2750	1.00
G2939	850	0.00	1150	0.45	2500	0.60	2600	2.15	2700	1.00
G2941	850	0.00	1150	0.45	2500	0.60	2600	2.20	2650	1.00
G2942	850	0.00	1150	0.45	2500	0.60	2600	2.20	2700	1.00
G2945	850	0.00	1150	0.45	2500	0.60	2600	2.25	2700	1.00
G2947	850	0.00	1150	0.45	2500	0.60	2650	2.15	2700	1.00
G2949	850	0.00	1150	0.45	2500	0.60	2650	2.20	2700	1.00
G2950	850	0.00	1150	0.45	2500	0.60	2650	2.20	2750	1.00
G2959	850	0.00	1150	0.45	2500	0.65	2600	2.15	2650	1.00
G2962	850	0.00	1150	0.45	2500	0.65	2600	2.20	2650	1.00
G2963	850	0.00	1150	0.45	2500	0.65	2600	2.20	2700	1.00
G2965	850	0.00	1150	0.45	2500	0.65	2600	2.25	2650	1.00
G2966	850	0.00	1150	0.45	2500	0.65	2600	2.25	2700	1.00
G2969	850	0.00	1150	0.45	2500	0.65	2650	2.15	2750	1.00
G2970	850	0.00	1150	0.45	2500	0.65	2650	2.20	2700	1.00
G2971	850	0.00	1150	0.45	2500	0.65	2650	2.20	2750	1.00
G2973	850	0.00	1150	0.45	2500	0.65	2650	2.25	2750	1.00
G2981	850	0.00	1150	0.45	2550	0.55	2600	2.15	2700	1.00
G2983	850	0.00	1150	0.45	2550	0.55	2600	2.20	2650	1.00
G2986	850	0.00	1150	0.45	2550	0.55	2600	2.25	2650	1.00
G2987	850	0.00	1150	0.45	2550	0.55	2600	2.25	2700	1.00
G2989	850	0.00	1150	0.45	2550	0.55	2650	2.15	2700	1.00
G2990	850	0.00	1150	0.45	2550	0.55	2650	2.15	2750	1.00
G2991	850	0.00	1150	0.45	2550	0.55	2650	2.20	2700	1.00
G2993	850	0.00	1150	0.45	2550	0.55	2650	2.25	2700	1.00
G2994	850	0.00	1150	0.45	2550	0.55	2650	2.25	2750	1.00
G3001	850	0.00	1150	0.45	2550	0.60	2600	2.15	2650	1.00
G3002	850	0.00	1150	0.45	2550	0.60	2600	2.15	2700	1.00
G3005	850	0.00	1150	0.45	2550	0.60	2600	2.20	2700	1.00
G3007	850	0.00	1150	0.45	2550	0.60	2600	2.25	2650	1.00
G3010	850	0.00	1150	0.45	2550	0.60	2650	2.15	2700	1.00
G3011	850	0.00	1150	0.45	2550	0.60	2650	2.15	2750	1.00
G3013	850	0.00	1150	0.45	2550	0.60	2650	2.20	2750	1.00
G3014	850	0.00	1150	0.45	2550	0.60	2650	2.25	2700	1.00
G3015	850	0.00	1150	0.45	2550	0.60	2650	2.25	2750	1.00
G3017	850	0.00	1150	0.45	2550	0.60	2700	2.15	2750	1.00
G3021	850	0.00	1150	0.45	2550	0.60	2700	2.25	2750	1.00
G3022	850	0.00	1150	0.45	2550	0.65	2600	2.15	2650	1.00
G3023	850	0.00	1150	0.45	2550	0.65	2600	2.15	2700	1.00
G3025	850	0.00	1150	0.45	2550	0.65	2600	2.20	2650	1.00
G3026	850	0.00	1150	0.45	2550	0.65	2600	2.20	2700	1.00
G3029	850	0.00	1150	0.45	2550	0.65	2600	2.25	2700	1.00
G3031	850	0.00	1150	0.45	2550	0.65	2650	2.15	2700	1.00
G3033	850	0.00	1150	0.45	2550	0.65	2650	2.20	2700	1.00

G146	750	0.00	1150	0.45	2600	0.60	2700	2.15	2750	1.00
G150	750	0.00	1150	0.45	2600	0.60	2700	2.25	2750	1.00
G151	750	0.00	1150	0.45	2600	0.65	2650	2.15	2700	1.00
G153	750	0.00	1150	0.45	2600	0.65	2650	2.20	2700	1.00
G154	750	0.00	1150	0.45	2600	0.65	2650	2.20	2750	1.00
G155	750	0.00	1150	0.45	2600	0.65	2650	2.25	2700	1.00
G158	750	0.00	1150	0.45	2600	0.65	2700	2.15	2750	1.00
G162	750	0.00	1150	0.45	2600	0.65	2700	2.25	2750	1.00
G166	750	0.00	1150	0.50	2500	0.55	2600	2.20	2650	1.00
G167	750	0.00	1150	0.50	2500	0.55	2600	2.20	2700	1.00
G169	750	0.00	1150	0.50	2500	0.55	2600	2.25	2650	1.00
G170	750	0.00	1150	0.50	2500	0.55	2600	2.25	2700	1.00
G173	750	0.00	1150	0.50	2500	0.55	2650	2.15	2750	1.00
G174	750	0.00	1150	0.50	2500	0.55	2650	2.20	2700	1.00
G175	750	0.00	1150	0.50	2500	0.55	2650	2.20	2750	1.00
G177	750	0.00	1150	0.50	2500	0.55	2650	2.25	2750	1.00
G185	750	0.00	1150	0.50	2500	0.60	2600	2.15	2700	1.00
G187	750	0.00	1150	0.50	2500	0.60	2600	2.20	2650	1.00
G190	750	0.00	1150	0.50	2500	0.60	2600	2.25	2650	1.00
G193	750	0.00	1150	0.50	2500	0.60	2650	2.15	2700	1.00
G194	750	0.00	1150	0.50	2500	0.60	2650	2.15	2750	1.00
G195	750	0.00	1150	0.50	2500	0.60	2650	2.20	2700	1.00
G197	750	0.00	1150	0.50	2500	0.60	2650	2.25	2700	1.00
G198	750	0.00	1150	0.50	2500	0.60	2650	2.25	2750	1.00
G205	750	0.00	1150	0.50	2500	0.65	2600	2.15	2650	1.00
G206	750	0.00	1150	0.50	2500	0.65	2600	2.15	2700	1.00
G209	750	0.00	1150	0.50	2500	0.65	2600	2.20	2700	1.00
G211	750	0.00	1150	0.50	2500	0.65	2600	2.25	2650	1.00
G214	750	0.00	1150	0.50	2500	0.65	2650	2.15	2700	1.00
G215	750	0.00	1150	0.50	2500	0.65	2650	2.15	2750	1.00
G217	750	0.00	1150	0.50	2500	0.65	2650	2.20	2750	1.00
G219	750	0.00	1150	0.50	2500	0.65	2650	2.25	2750	1.00
G226	750	0.00	1150	0.50	2550	0.55	2600	2.15	2650	1.00
G227	750	0.00	1150	0.50	2550	0.55	2600	2.15	2700	1.00
G229	750	0.00	1150	0.50	2550	0.55	2600	2.20	2650	1.00
G230	750	0.00	1150	0.50	2550	0.55	2600	2.20	2700	1.00
G233	750	0.00	1150	0.50	2550	0.55	2600	2.25	2700	1.00
G235	750	0.00	1150	0.50	2550	0.55	2650	2.15	2700	1.00
G237	750	0.00	1150	0.50	2550	0.55	2650	2.20	2700	1.00
G238	750	0.00	1150	0.50	2550	0.55	2650	2.20	2750	1.00
G239	750	0.00	1150	0.50	2550	0.55	2650	2.25	2700	1.00
G242	750	0.00	1150	0.50	2550	0.55	2700	2.15	2750	1.00
G246	750	0.00	1150	0.50	2550	0.55	2700	2.25	2750	1.00
G247	750	0.00	1150	0.50	2550	0.60	2600	2.15	2650	1.00
G250	750	0.00	1150	0.50	2550	0.60	2600	2.20	2650	1.00
G251	750	0.00	1150	0.50	2550	0.60	2600	2.20	2700	1.00
G253	750	0.00	1150	0.50	2550	0.60	2600	2.25	2650	1.00
G254	750	0.00	1150	0.50	2550	0.60	2600	2.25	2700	1.00
G257	750	0.00	1150	0.50	2550	0.60	2650	2.15	2750	1.00
G258	750	0.00	1150	0.50	2550	0.60	2650	2.20	2700	1.00
G259	750	0.00	1150	0.50	2550	0.60	2650	2.20	2750	1.00
G261	750	0.00	1150	0.50	2550	0.60	2650	2.25	2750	1.00
G263	750	0.00	1150	0.50	2550	0.60	2700	2.15	2750	1.00
G267	750	0.00	1150	0.50	2550	0.60	2700	2.25	2750	1.00
G269	750	0.00	1150	0.50	2550	0.65	2600	2.15	2700	1.00
G271	750	0.00	1150	0.50	2550	0.65	2600	2.20	2650	1.00
G274	750	0.00	1150	0.50	2550	0.65	2600	2.25	2650	1.00
G275	750	0.00	1150	0.50	2550	0.65	2600	2.25	2700	1.00
G277	750	0.00	1150	0.50	2550	0.65	2650	2.15	2700	1.00
G278	750	0.00	1150	0.50	2550	0.65	2650	2.15	2750	1.00
G279	750	0.00	1150	0.50	2550	0.65	2650	2.20	2700	1.00
G281	750	0.00	1150	0.50	2550	0.65	2650	2.25	2700	1.00
G282	750	0.00	1150	0.50	2550	0.65	2650	2.25	2750	1.00
G289	750	0.00	1150	0.50	2600	0.55	2650	2.15	2700	1.00
G290	750	0.00	1150	0.50	2600	0.55	2650	2.15	2750	1.00
G291	750	0.00	1150	0.50	2600	0.55	2650	2.20	2700	1.00
G293	750	0.00	1150	0.50	2600	0.55	2650	2.25	2700	1.00
G294	750	0.00	1150	0.50	2600	0.55	2650	2.25	2750	1.00
G301	750	0.00	1150	0.50	2600	0.60	2650	2.15	2700	1.00
G302	750	0.00	1150	0.50	2600	0.60	2650	2.15	2750	1.00
G303	750	0.00	1150	0.50	2600	0.60	2650	2.20	2700	1.00
G305	750	0.00	1150	0.50	2600	0.60	2650	2.25	2700	1.00
G306	750	0.00	1150	0.50	2600	0.60	2650	2.25	2750	1.00
G313	750	0.00	1150	0.50	2600	0.65	2650	2.15	2700	1.00
G314	750	0.00	1150	0.50	2600	0.65	2650	2.15	2750	1.00
G315	750	0.00	1150	0.50	2600	0.65	2650	2.20	2700	1.00
G317	750	0.00	1150	0.50	2600	0.65	2650	2.25	2700	1.00
G318	750	0.00	1150	0.50	2600	0.65	2650	2.25	2750	1.00
G326	750	0.00	1150	0.55	2500	0.55	2600	2.15	2700	1.00
G329	750	0.00	1150	0.55	2500	0.55	2600	2.20	2700	1.00
G331	750	0.00	1150	0.55	2500	0.55	2600	2.25	2650	1.00
G334	750	0.00	1150	0.55	2500	0.55	2650	2.15	2700	1.00
G335	750	0.00	1150	0.55	2500	0.55	2650	2.15	2750	1.00
G337	750	0.00	1150	0.55	2500	0.55	2650	2.20	2750	1.00
G338	750	0.00	1150	0.55	2500	0.55	2650	2.25	2700	1.00
G339	750	0.00	1150	0.55	2500	0.55	2650	2.25	2750	1.00
G347	750	0.00	1150	0.55	2500	0.60	2600	2.15	2700	1.00
G349	750	0.00	1150	0.55	2500	0.60	2600	2.20	2650	1.00
G350	750	0.00	1150	0.55	2500	0.60	2600	2.20	2700	1.00
G355	750	0.00	1150	0.55	2500	0.60	2650	2.15	2700	1.00
G357	750	0.00	1150	0.55	2500	0.60	2650	2.20	2700	1.00
G358	750	0.00	1150	0.55	2500	0.60	2650	2.20	2750	1.00
G359	750	0.00	1150	0.55	2500	0.60	2650	2.25	2700	1.00
G370	750	0.00	1150	0.55	2500	0.65	2600	2.20	2650	1.00
G373	750	0.00	1150	0.55	2500	0.65	2600	2.25	2650	1.00
G374	750	0.00	1150	0.55	2500	0.65	2600	2.25	2700	1.00
G377	750	0.00	1150	0.55	2500	0.65	2650	2.15	2750	1.00
G378	750	0.00	1150	0.55	2500	0.65	2650	2.20	2700	1.00
G379	750	0.00	1150	0.55	2500	0.65	2650	2.20	2750	1.00
G381	750	0.00	1150	0.55	2500	0.65	2650	2.25	2750	1.00
G389	750	0.00	1150	0.55	2550	0.55	2600	2.15	2700	1.00
G391	750	0.00	1150	0.55	2550	0.55	2600	2.20	2650	1.00
G394	750	0.00	1150	0.55	2550	0.55	2600	2.25	2650	1.00
G395	750	0.00	1150	0.55	2550	0.55	2600	2.25	2700	1.00
G397	750	0.00	1150	0.55	2550	0.55	2650	2.15	2700	1.00
G398	750	0.00	1150	0.55	2550	0.55	2650	2.15	2750	1.00
G399	750	0.00	1150	0.55	2550	0.55	2650	2.20	2700	1.00
G401	750	0.00	1150	0.55	2550	0.55	2650	2.25	2700	1.00
G402	750	0.00	1150	0.55	2550	0.55	2650	2.25	2750	1.00
G3034	850	0.00	1150	0.45	2550	0.65	2650	2.20	2750	1.00
G3035	850	0.00	1150	0.45	2550	0.65	2650	2.25	2700	1.00
G3038	850	0.00	1150	0.45	2550	0.65	2700	2.15	2750	1.00
G3042	850	0.00	1150	0.45	2550	0.65	2700	2.25	2750	1.00
G3043	850	0.00	1150	0.45	2600	0.55	2650	2.15	2700	1.00
G3045	850	0.00	1150	0.45	2600	0.55	2650	2.20	2700	1.00
G3047	850	0.00	1150	0.45	2600	0.55	2650	2.25	2700	1.00
G3050	850	0.00	1150	0.45	2600	0.55	2700	2.15	2750	1.00
G3054	850	0.00	1150	0.45	2600	0.55	2700	2.25	2750	1.00
G3055	850	0.00	1150	0.45	2600	0.60	2650	2.15	2700	1.00
G3057	850	0.00	1150	0.45	2600	0.60	2650	2.20	2700	1.00
G3058	850	0.00	1150	0.45	2600	0.60	2650	2.20	2750	1.00
G3059	850	0.00	1150	0.45	2600	0.60	2650	2.25	2700	1.00
G3062	850	0.00	1150	0.45	2600	0.60	2700	2.15	2750	1.00
G3066	850	0.00	1150	0.45	2600	0.60	2700	2.25	2750	1.00
G3067	850	0.00	1150	0.45	2600	0.65	2650	2.15	2700	1.00
G3069	850	0.00	1150	0.45	2600	0.65	2650	2.20	2700	1.00
G3070	850	0.00	1150	0.45	2600	0.65	2650	2.20	2750	1.00
G3071	850	0.00	1150	0.45	2600	0.65	2650	2.25	2700	1.00
G3074	850	0.00	1150	0.45	2600	0.65	2700	2.15	2750	1.00
G3078	850	0.00	1150	0.45	2600	0.65	2700	2.25	2750	1.00
G3082	850	0.00	1150	0.50	2500	0.55	2600	2.20	2650	1.00
G3083	850	0.00	1150	0.50	2500	0.55	2600	2.20	2700	1.00
G3085	850	0.00	1150	0.50	2500	0.55	2600	2.25	2650	1.00
G3086	850	0.00	1150	0.50	2500	0.55	2600	2.25	2700	1.00
G3090	850	0.00	1150	0.50	2500	0.55	2650	2.20	2700	1.00
G3091	850	0.00	1150	0.50	2500	0.55	2650	2.20	2750	1.00
G3093	850	0.00	1150	0.50	2500	0.55	2650	2.25	2750	1.00
G3101	850	0.00	1150	0.50	2500	0.60	2600	2.15	2700	1.00
G3103	850	0.00	1150	0.50	2500	0.60	2600	2.20	2650	1.00
G3106	850	0.00	1150	0.50	2500	0.60	2600	2.25	2650	1.00
G3109	850	0.00	1150	0.50	2500	0.60	2650	2.15	2700	1.00
G3110	850	0.00	1150	0.50	2500	0.60	2650	2.15	2750	1.00
G3111	850	0.00	1150	0.50	2500	0.60	2650	2.20	2700	1.00
G3113	850	0.00	1150	0.50	2500	0.60	2650	2.25	2700	1.00
G3114	850	0.00	1150	0.50	2500	0.60	2650	2.25	2750	1.00
G3122	850	0.00	1150	0.50	2500	0.65	2600	2.15	2700	1.00
G3125	850	0.00	1150	0.50	2500	0.65	2600	2.20	2700	1.00
G3127	850	0.00	1150	0.50	2500	0.65	2600	2.25	2650	1.00
G3130	850	0.00	1150	0.50	2500	0.65	2650	2.15	2700	1.00
G3131	850	0.00	1150	0.50	2500	0.65	2650	2.15	2750	1.00
G3133	850	0.00	1150	0.50	2500	0.65	2650	2.20	2750	1.00
G3134	850	0.00	1150	0.50	2500	0.65	2650	2.25	2700	1.00
G3135	850	0.00	1150	0.50	2500	0.65	2650	2.25	2750	1.00
G3142	850	0.00	1150	0.50	2550	0.55	2600	2.15	2650	1.00
G3143	850	0.00	1150	0.50	2550	0.55	2600	2.15	2700	1.00
G3145	850	0.00	1150	0.50	2550	0.55	2600	2.20	2650	1.00
G3146	850	0.00	1150	0.50	2550	0.55	2600	2.20	2700	1.00
G3151	850	0.00	1150	0.50	2550	0.55	2650	2.15	2700	1.00
G3153	850	0.00	1150	0.50	2550	0.55	2650	2.20	2700	1.00
G3154	850	0.00	1150	0.50	2550	0.55	2650	2.20	2750	1.00
G3155	850	0.00	1150	0.50	2550	0.55	2650	2.25	2700	1.00
G3158	850	0.00	1150	0.50	2550	0.55	2700	2.15	2750	1.00
G3162	850	0.00	1150	0.50	2550	0.55	2700	2.25	2750	1.00
G3163	850	0.00	1150	0.50	2550	0.60	2600	2.15	2650	1.00
G3166	850	0.00	1150	0.50	2550	0.60	2600	2.20	2650	1.00
G3167	850	0.00	1150	0.50	2550	0.60	2600	2.20	2700	1.00
G3169	850	0.00	1150	0.50	2550	0.60	2600	2.25	2650	1.00
G3170	850	0.00	1150	0.50	2550	0.60	2600	2.25	2700	1.00
G3173	850	0.00	1150	0.50	2550	0.60	2650	2.15	2750	1.00
G3174	850	0.00	1150	0.50	2550	0.60	2650	2.20	2700	1.00
G3175	850	0.00	1150	0.50	2550	0.60	2650	2.20	2750	1.00
G3177	850	0.00	1150	0.50	2550	0.60	2650	2.25	2750	1.00
G3179	850	0.00	1150	0.50	2550	0.60	2700	2.15	2750	1.00
G3183	850	0.00	1150	0.50	2550	0.60	2700	2.25	2750	1.00
G3185	850	0.00	1150	0.50	2550	0.65	2600	2.15	2700	1.00
G3187	850	0.00	1150	0.50	2550	0.65	2600	2.20	2650	1.00
G3190	850	0.00	1150	0.50	2550	0.65	2600	2.25	2650	1.00
G3191	850	0.00	1150	0.50	2550	0.65	2600	2.25	2700	1.00
G3193	850	0.00	1150	0.50	2550	0.65	2650	2.15	2700	1.00
G3194	850	0.00	1150	0.50	2550	0.65	2650	2.15	2750	1.00
G3195	850	0.00	1150	0.50	2550	0.65	2650	2.20	2700	1.00
G3197	850	0.00	1150	0.50	2550	0.65	2650	2.25	2700	1.00
G3198	850	0.00	1150	0.50	2550	0.65	2650	2.25	2750	1.00
G3205	850	0.00	1150	0.50	2600	0.55	2650	2.15	2700	1.00
G3206	850	0.00	1150	0.50	2600	0.55	2650	2.15	2750	1.00
G3207	850	0.00	1150	0.50	2600	0.55	2650	2.20	2700	1.00
G3209	850	0.00	1150	0.50	2600	0.55	2650	2.25	2700	1.00
G3210	850	0.00	1150	0.50	2600	0.55	2650	2.25	2750	1.00
G3217	850	0.00	1150	0.50	2600	0.60	2650	2.15	2700	1.00
G3218	850	0.00	1150	0.50	2600	0.60	2650	2.15	2750	1.00
G3219	850	0.00	1150	0.50	2600	0.60	2650	2.20	2700	1.00
G3221	850	0.00	1150	0.50	2600	0.60	2650	2.25	2700	1.00
G3222	850	0.00	1150	0.50	2600	0.60	2650	2.25	2750	1.00
G3229	850	0.00	1150	0.50	2600	0.65	2650	2.15	2700	1.00
G3230	850	0.00	1150	0.50	2600	0.65	2650	2.15	2750	1.00
G3231	850	0.00	1150	0.50	2600	0.65	2650	2.20	2700	1.00
G3233	850	0.00	1150	0.50	2600	0.65	2650	2.25	2700	1.00
G3234	850	0.00	1150	0.50	2600	0.65	2650	2.25	2750	1.00
G3242	850	0.00	1150	0.55	2500	0.55	2600	2.15	2700	1.00
G3247	850	0.00	1150	0.55	2500	0.55	2600	2.25	2650	1.00
G3250	850	0.00	1150	0.55	2500	0.55	2650	2.15	2700	1.00
G3251	850	0.00	1150	0.55	2500	0.55	2650	2.15	2750	1.00
G3253	850	0.00	1150	0.55	2500	0.55	2650	2.20	2750	1.00
G3254	850	0.00	1150	0.55	2500	0.55	2650	2.25	2700	1.00
G3255	850	0.00	1150	0.55	2500	0.55	2650	2.25	2750	1.00
G3263	850	0.00	1150	0.55	2500	0.60	2600	2.15	2700	1.00
G3265	850	0.00	1150	0.55	2500	0.60	2600	2.20	2650	1.00
G3266	850	0.00	1150	0.55	2500	0.60	2600	2.20	2700	1.00
G3271	850	0.00	1150	0.55	2500	0.60	2650	2.15	2700	1.00
G3273	850	0.00	1150	0.55	2500	0.60	2650	2.20	2700	1.00
G3274	850	0.00	1150	0.55	2500	0.60	2650	2.20	2750	1.00
G3275	850	0.00	1150	0.55	2500	0.60	2650	2.25	2700	1.00
G3286	850	0.00	1150	0.55	2500	0.65	2600	2.20	2650	1.00
G3289	850	0.00	1150	0.55	2500	0.65	2600	2.25	2650	1.00
G3290	850	0.00	1150	0.55	2500	0.65	2600	2.25	2700	1.00
G3293	850	0.00	1150	0.55	2500	0.65	2650	2.15	2750	1.00
G3294	850	0.00	1150	0.55	2500	0.65	2650	2.20	2700	1.00
G3295	850	0.00	1150	0.55	2500	0.65	2650	2.20	2750	1.00

G406	750	0.00	1150	0.55	2550	0.55	2700	2.20	2750	1.00
G409	750	0.00	1150	0.55	2550	0.60	2600	2.15	2650	1.00
G410	750	0.00	1150	0.55	2550	0.60	2600	2.15	2700	1.00
G413	750	0.00	1150	0.55	2550	0.60	2600	2.20	2700	1.00
G415	750	0.00	1150	0.55	2550	0.60	2600	2.25	2650	1.00
G418	750	0.00	1150	0.55	2550	0.60	2650	2.15	2700	1.00
G421	750	0.00	1150	0.55	2550	0.60	2650	2.20	2750	1.00
G422	750	0.00	1150	0.55	2550	0.60	2650	2.25	2700	1.00
G423	750	0.00	1150	0.55	2550	0.60	2650	2.25	2750	1.00
G425	750	0.00	1150	0.55	2550	0.60	2700	2.15	2750	1.00
G429	750	0.00	1150	0.55	2550	0.60	2700	2.25	2750	1.00
G430	750	0.00	1150	0.55	2550	0.65	2600	2.15	2650	1.00
G431	750	0.00	1150	0.55	2550	0.65	2600	2.15	2700	1.00
G433	750	0.00	1150	0.55	2550	0.65	2600	2.20	2650	1.00
G434	750	0.00	1150	0.55	2550	0.65	2600	2.20	2700	1.00
G437	750	0.00	1150	0.55	2550	0.65	2600	2.25	2700	1.00
G439	750	0.00	1150	0.55	2550	0.65	2650	2.15	2700	1.00
G441	750	0.00	1150	0.55	2550	0.65	2650	2.20	2700	1.00
G442	750	0.00	1150	0.55	2550	0.65	2650	2.20	2750	1.00
G443	750	0.00	1150	0.55	2550	0.65	2650	2.25	2700	1.00
G446	750	0.00	1150	0.55	2550	0.65	2700	2.15	2750	1.00
G450	750	0.00	1150	0.55	2550	0.65	2700	2.25	2750	1.00
G451	750	0.00	1150	0.55	2600	0.55	2650	2.15	2700	1.00
G453	750	0.00	1150	0.55	2600	0.55	2650	2.20	2700	1.00
G454	750	0.00	1150	0.55	2600	0.55	2650	2.20	2750	1.00
G455	750	0.00	1150	0.55	2600	0.55	2650	2.25	2700	1.00
G458	750	0.00	1150	0.55	2600	0.55	2700	2.15	2750	1.00
G462	750	0.00	1150	0.55	2600	0.55	2700	2.25	2750	1.00
G463	750	0.00	1150	0.55	2600	0.60	2650	2.15	2700	1.00
G465	750	0.00	1150	0.55	2600	0.60	2650	2.20	2700	1.00
G466	750	0.00	1150	0.55	2600	0.60	2650	2.20	2750	1.00
G467	750	0.00	1150	0.55	2600	0.60	2650	2.25	2700	1.00
G470	750	0.00	1150	0.55	2600	0.60	2700	2.15	2750	1.00
G474	750	0.00	1150	0.55	2600	0.60	2700	2.25	2750	1.00
G475	750	0.00	1150	0.55	2600	0.65	2650	2.15	2700	1.00
G477	750	0.00	1150	0.55	2600	0.65	2650	2.20	2700	1.00
G478	750	0.00	1150	0.55	2600	0.65	2650	2.20	2750	1.00
G479	750	0.00	1150	0.55	2600	0.65	2650	2.25	2700	1.00
G482	750	0.00	1150	0.55	2600	0.65	2700	2.15	2750	1.00
G486	750	0.00	1150	0.55	2600	0.65	2700	2.25	2750	1.00
G487	750	0.00	1200	0.45	2500	0.55	2600	2.15	2650	1.00
G490	750	0.00	1200	0.45	2500	0.55	2600	2.20	2650	1.00
G491	750	0.00	1200	0.45	2500	0.55	2600	2.20	2700	1.00
G493	750	0.00	1200	0.45	2500	0.55	2600	2.25	2650	1.00
G494	750	0.00	1200	0.45	2500	0.55	2600	2.25	2700	1.00
G497	750	0.00	1200	0.45	2500	0.55	2650	2.15	2750	1.00
G498	750	0.00	1200	0.45	2500	0.55	2650	2.20	2700	1.00
G499	750	0.00	1200	0.45	2500	0.55	2650	2.20	2750	1.00
G511	750	0.00	1200	0.45	2500	0.60	2600	2.20	2650	1.00
G514	750	0.00	1200	0.45	2500	0.60	2600	2.25	2650	1.00
G515	750	0.00	1200	0.45	2500	0.60	2600	2.25	2700	1.00
G517	750	0.00	1200	0.45	2500	0.60	2650	2.15	2700	1.00
G518	750	0.00	1200	0.45	2500	0.60	2650	2.15	2750	1.00
G519	750	0.00	1200	0.45	2500	0.60	2650	2.20	2700	1.00
G521	750	0.00	1200	0.45	2500	0.60	2650	2.25	2700	1.00
G522	750	0.00	1200	0.45	2500	0.60	2650	2.25	2750	1.00
G529	750	0.00	1200	0.45	2500	0.65	2600	2.15	2650	1.00
G530	750	0.00	1200	0.45	2500	0.65	2600	2.15	2700	1.00
G533	750	0.00	1200	0.45	2500	0.65	2600	2.20	2700	1.00
G535	750	0.00	1200	0.45	2500	0.65	2600	2.25	2650	1.00
G538	750	0.00	1200	0.45	2500	0.65	2650	2.15	2700	1.00
G539	750	0.00	1200	0.45	2500	0.65	2650	2.15	2750	1.00
G541	750	0.00	1200	0.45	2500	0.65	2650	2.20	2750	1.00
G542	750	0.00	1200	0.45	2500	0.65	2650	2.25	2700	1.00
G543	750	0.00	1200	0.45	2500	0.65	2650	2.25	2750	1.00
G550	750	0.00	1200	0.45	2550	0.55	2600	2.15	2650	1.00
G553	750	0.00	1200	0.45	2550	0.55	2600	2.20	2650	1.00
G554	750	0.00	1200	0.45	2550	0.55	2600	2.20	2700	1.00
G557	750	0.00	1200	0.45	2550	0.55	2600	2.25	2700	1.00
G559	750	0.00	1200	0.45	2550	0.55	2650	2.15	2700	1.00
G561	750	0.00	1200	0.45	2550	0.55	2650	2.20	2700	1.00
G562	750	0.00	1200	0.45	2550	0.55	2650	2.20	2750	1.00
G563	750	0.00	1200	0.45	2550	0.55	2650	2.25	2700	1.00
G566	750	0.00	1200	0.45	2550	0.55	2700	2.15	2750	1.00
G570	750	0.00	1200	0.45	2550	0.55	2700	2.25	2750	1.00
G571	750	0.00	1200	0.45	2550	0.60	2600	2.15	2650	1.00
G574	750	0.00	1200	0.45	2550	0.60	2600	2.20	2650	1.00
G577	750	0.00	1200	0.45	2550	0.60	2600	2.25	2650	1.00
G578	750	0.00	1200	0.45	2550	0.60	2600	2.25	2700	1.00
G581	750	0.00	1200	0.45	2550	0.60	2650	2.15	2750	1.00
G582	750	0.00	1200	0.45	2550	0.60	2650	2.20	2700	1.00
G583	750	0.00	1200	0.45	2550	0.60	2650	2.20	2750	1.00
G585	750	0.00	1200	0.45	2550	0.60	2650	2.25	2750	1.00
G587	750	0.00	1200	0.45	2550	0.60	2700	2.15	2750	1.00
G589	750	0.00	1200	0.45	2550	0.60	2700	2.20	2750	1.00
G591	750	0.00	1200	0.45	2550	0.60	2700	2.25	2750	1.00
G593	750	0.00	1200	0.45	2550	0.65	2600	2.15	2700	1.00
G595	750	0.00	1200	0.45	2550	0.65	2600	2.20	2650	1.00
G598	750	0.00	1200	0.45	2550	0.65	2600	2.25	2650	1.00
G599	750	0.00	1200	0.45	2550	0.65	2600	2.25	2700	1.00
G602	750	0.00	1200	0.45	2550	0.65	2650	2.15	2750	1.00
G603	750	0.00	1200	0.45	2550	0.65	2650	2.20	2700	1.00
G605	750	0.00	1200	0.45	2550	0.65	2650	2.25	2700	1.00
G606	750	0.00	1200	0.45	2550	0.65	2650	2.25	2750	1.00
G610	750	0.00	1200	0.45	2550	0.65	2700	2.20	2750	1.00
G613	750	0.00	1200	0.45	2600	0.55	2650	2.15	2700	1.00
G614	750	0.00	1200	0.45	2600	0.55	2650	2.15	2750	1.00
G615	750	0.00	1200	0.45	2600	0.55	2650	2.20	2700	1.00
G617	750	0.00	1200	0.45	2600	0.55	2650	2.25	2700	1.00
G618	750	0.00	1200	0.45	2600	0.55	2650	2.25	2750	1.00
G622	750	0.00	1200	0.45	2600	0.55	2700	2.20	2750	1.00
G625	750	0.00	1200	0.45	2600	0.60	2650	2.15	2700	1.00
G626	750	0.00	1200	0.45	2600	0.60	2650	2.15	2750	1.00
G627	750	0.00	1200	0.45	2600	0.60	2650	2.20	2700	1.00
G629	750	0.00	1200	0.45	2600	0.60	2650	2.25	2700	1.00
G630	750	0.00	1200	0.45	2600	0.60	2650	2.25	2750	1.00
G634	750	0.00	1200	0.45	2600	0.60	2700	2.20	2750	1.00
G637	750	0.00	1200	0.45	2600	0.65	2650	2.15	2700	1.00
G638	750	0.00	1200	0.45	2600	0.65	2650	2.15	2750	1.00
G3297	850	0.00	1150	0.55	2500	0.65	2650	2.25	2750	1.00
G3305	850	0.00	1150	0.55	2550	0.55	2600	2.15	2700	1.00
G3307	850	0.00	1150	0.55	2550	0.55	2600	2.20	2650	1.00
G3310	850	0.00	1150	0.55	2550	0.55	2600	2.25	2650	1.00
G3311	850	0.00	1150	0.55	2550	0.55	2600	2.25	2700	1.00
G3313	850	0.00	1150	0.55	2550	0.55	2650	2.15	2700	1.00
G3314	850	0.00	1150	0.55	2550	0.55	2650	2.15	2750	1.00
G3315	850	0.00	1150	0.55	2550	0.55	2650	2.20	2700	1.00
G3317	850	0.00	1150	0.55	2550	0.55	2650	2.25	2700	1.00
G3945C	850	0.00	1250	0.45	2500	0.65	2650	2.25	2750	1.00
G3956C	850	0.00	1250	0.45	2550	0.55	2600	2.20	2700	1.00
G3974C	850	0.00	1250	0.45	2550	0.60	2600	2.15	2700	1.00
G3980C	850	0.00	1250	0.45	2550	0.60	2600	2.25	2700	1.00
G4044C	850	0.00	1250	0.45	2600	0.65	2650	2.25	2750	1.00
G4259C	850	0.00	1250	0.55	2500	0.65	2600	2.20	2700	1.00
G4298C	850	0.00	1250	0.55	2550	0.60	2600	2.15	2700	1.00
G3425C	850	0.00	1200	0.45	2500	0.60	2600	2.15	2700	1.00
G3467C	850	0.00	1200	0.45	2550	0.55	2600	2.15	2700	1.00
G3491C	850	0.00	1200	0.45	2550	0.60	2600	2.20	2700	1.00
G3653C	850	0.00	1200	0.50	2550	0.60	2600	2.20	2700	1.00
G191C	750	0.00	1150	0.50	2500	0.60	2600	2.25	2700	1.00
G353C	750	0.00	1150	0.55	2500	0.60	2600	2.25	2700	1.00
G371C	750	0.00	1150	0.55	2500	0.65	2600	2.20	2700	1.00
G551C	750	0.00	1200	0.45	2550	0.55	2600	2.15	2700	1.00
G575C	750	0.00	1200	0.45	2550	0.60	2600	2.20	2700	1.00
G737C	750	0.00	1200	0.50	2550	0.60	2600	2.20	2700	1.00
G862C	750	0.00	1200	0.55	2500	0.65	2650	2.15	2700	1.00
G1598C	800	0.00	1150	0.45	2600	0.60	2650	2.15	2750	1.00
G1610C	800	0.00	1150	0.45	2600	0.65	2650	2.15	2750	1.00
G1649C	800	0.00	1150	0.50	2500	0.60	2600	2.25	2700	1.00
G1811C	800	0.00	1150	0.55	2500	0.60	2600	2.25	2700	1.00
G1829C	800	0.00	1150	0.55	2500	0.65	2600	2.20	2700	1.00
G1967C	800	0.00	1200	0.45	2500	0.60	2600	2.15	2700	1.00
G2009C	800	0.00	1200	0.45	2550	0.55	2600	2.15	2700	1.00
G2030C	800	0.00	1200	0.45	2550	0.60	2600	2.15	2700	1.00
G2033C	800	0.00	1200	0.45	2550	0.60	2600	2.20	2700	1.00
G2195C	800	0.00	1200	0.50	2550	0.60	2600	2.20	2700	1.00
G2383C	800	0.00	1200	0.55	2550	0.65	2650	2.15	2700	1.00
G2487C	800	0.00	1250	0.45	2500	0.65	2650	2.25	2750	1.00
G2498C	800	0.00	1250	0.45	2550	0.55	2600	2.20	2700	1.00
G2516C	800	0.00	1250	0.45	2550	0.60	2600	2.15	2700	1.00
G2522C	800	0.00	1250	0.45	2550	0.60	2600	2.25	2700	1.00
G2586C	800	0.00	1250	0.45	2600	0.65	2650	2.25	2750	1.00
G2801C	800	0.00	1250	0.55	2500	0.65	2600	2.20	2700	1.00
G3107C	850	0.00	1150	0.50	2500	0.60	2600	2.25	2700	1.00
G3269C	850	0.00	1150	0.55	2500	0.60	2600	2.25	2700	1.00
G3287C	850	0.00	1150	0.55	2500	0.65	2600	2.20	2700	1.00
G116A	750	0.00	1150	0.45	2550	0.65	2650	2.15	2750	1.00
G120A	750	0.00	1150	0.45	2550	0.65	2650	2.25	2750	1.00
G124A	750	0.00	1150	0.45	2550	0.65	2700	2.20	2750	1.00
G12A	750	0.00	1150	0.45	2500	0.55	2650	2.20	2700	1.00
G136A	750	0.00	1150	0.45	2600	0.55	2700	2.20	2750	1.00
G140A	750	0.00	1150	0.45	2600	0.60	2650	2.15	2750	1.00
G144A	750	0.00	1150	0.45	2600	0.60	2650	2.25	2750	1.00
G1468A	800	0.00	1150	0.45	2500	0.55	2650	2.15	2700	1.00
G1472A	800	0.00	1150	0.45	2500	0.55	2650	2.25	2700	1.00
G1484A	800	0.00	1150	0.45	2500	0.60	2600	2.20	2700	1.00
G148A	750	0.00	1150	0.45	2600	0.60	2700	2.20	2750	1.00
G1492A	800	0.00	1150	0.45	2500	0.60	2650	2.20	2750	1.00
G1504A	800	0.00	1150	0.45	2500	0.65	2600	2.20	2650	1.00
G1508A	800	0.00	1150	0.45	2500	0.65	2600	2.25	2700	1.00
G1512A	800	0.00	1150	0.45	2500	0.65	2650	2.20	2700	1.00
G1532A	800	0.00	1150	0.45	2550	0.55	2650	2.15	2750	1.00
G1536A	800	0.00	1150	0.45	2550	0.55	2650	2.25	2750	1.00
G1544A	800	0.00	1150	0.45	2550	0.60	2600	2.15	2700	1.00
G1556A	800	0.00	1150	0.45	2550	0.60	2650	2.25	2700	1.00
G1564A	800	0.00	1150	0.45	2550	0.65	2600	2.15	2650	1.00
G1568A	800	0.00	1150	0.45	2550	0.65	2600	2.20	2700	1.00
G1576A	800	0.00	1150	0.45	2550	0.65	2650	2.20	2750	1.00
G1580A	800	0.00	1150	0.45	2550	0.65	2700	2.15	2750	1.00
G1584A	800	0.00	1150	0.45	2550	0.65	2700	2.25	2750	1.00
G1592A	800	0.00	1150	0.45	2600	0.55	2700	2.15	2750	1.00
G1596A	800	0.00	1150	0.45	2600	0.55	2700	2.25	2750	1.00
G1600A	800	0.00	1150	0.45	2600	0.60	2650	2.20	2750	1.00
G1604A	800	0.00	1150	0.45	2600	0.60	2700	2.15	2750	1.00
G1608A	800	0.00	1150	0.45	2600	0.60	2700	2.25	2750	1.00
G160A	750	0.00	1150	0.45	2600	0.65	2700	2.20	2750	1.00
G1616A	800	0.00	1150	0.45	2600	0.65	2700	2.15	2750	1.00
G1620A	800	0.00	1150	0.45	2600	0.65	2700	2.25	2750	1.00
G1624A	800	0.00	1150	0.50	2500	0.55	2600	2.20	2650	1.00
G1628A	800	0.00	1150	0.50	2500	0.55	2600	2.25	2700	1.00
G1632A	800	0.00	1150	0.50	2500	0.55	2650	2.20	2700	1.00
G1648A	800	0.00	1150	0.50	2500	0.60	2600	2.25	2650	1.00
G164A	750	0.00	1150	0.50	2500	0.55	2600	2.15	2700	1.00
G1652A	800	0.00	1150	0.50	2500	0.60	2650	2.15	2750	1.00
G1656A	800	0.00	1150	0.50	2500	0.60	2650	2.25	2750	1.00
G1664A	800	0.00	1150	0.50	2500	0.65	2600	2.15	2700	1.00
G1672A	800	0.00	1150	0.50	2500	0.65	2650	2.15	2700	1.00
G1676A	800	0.00	1150	0.50	2500	0.65	2650	2.25	2700	1.00
G1684A	800	0.00	1150	0.50	2550	0.55	2600	2.15	2650	1.00
G1688A	800	0.00	1150	0.50	2550	0.55	2600	2.20	2700	1.00
G1696A	800	0.00	1150	0.50	2550	0.55	2650	2.20	2750	1.00
G1700A	800	0.00	1150	0.50	2550	0.55	2700	2.15	2750	1.00
G1704A	800	0.00	1150	0.50	2550	0.55	2700	2.25	2750	1.00
G1712A	800	0.00	1150	0.50	2550	0.60	2600	2.25	2700	1.00
G1716A	800	0.00	1150	0.50	2550	0.60	2650	2.20	2700	1.00
G172A	750	0.00	1150	0.50	2500	0.55	2650	2.15	2700	1.00
G1736A	800	0.00	1150	0.50	2550	0.65	2650	2.15	2750	1.00
G1740A	800	0.00	1150	0.50	2550	0.65	2650	2.25	2750	1.00
G1748A	800	0.00	1150	0.50	2600	0.55	2650	2.15	2750	1.00
G1752A	800	0.00	1150	0.50	2600	0.55	2650	2.25	2750	1.00
G1760A	800	0.00	1150	0.50	2600	0.60	2650	2.15	2750	1.00
G1764A	800	0.00	1150	0.50	2600	0.60	2650	2.25	2750	1.00
G1768A	800	0.00	1150	0.50	2600	0.60	2700	2.20	2750	1.00
G176A	750	0.00	1150	0.50	2500	0.55	2650	2.25	2700	1.00
G1784A	800	0.00	1150	0.55	2500	0.55	2600	2.15	2700	1.00
G188A	750	0.00	1150	0.50	2500	0.60	2600	2.20	2700	1.00
G208A	750	0.00	1150	0.50	2500	0.65	2600	2.20	2650	1.00
G212A	750	0.00	1150	0.50	2500	0.65	2600	2.25	2700	1.00

G639	750	0.00	1200	0.45	2600	0.65	2650	2.20	2700	1.00
G641	750	0.00	1200	0.45	2600	0.65	2650	2.25	2700	1.00
G642	750	0.00	1200	0.45	2600	0.65	2650	2.25	2750	1.00
G646	750	0.00	1200	0.45	2600	0.65	2700	2.20	2750	1.00
G649	750	0.00	1200	0.50	2500	0.55	2600	2.15	2650	1.00
G650	750	0.00	1200	0.50	2500	0.55	2600	2.15	2700	1.00
G653	750	0.00	1200	0.50	2500	0.55	2600	2.20	2700	1.00
G655	750	0.00	1200	0.50	2500	0.55	2600	2.25	2650	1.00
G658	750	0.00	1200	0.50	2500	0.55	2650	2.15	2700	1.00
G659	750	0.00	1200	0.50	2500	0.55	2650	2.15	2750	1.00
G661	750	0.00	1200	0.50	2500	0.55	2650	2.20	2750	1.00
G662	750	0.00	1200	0.50	2500	0.55	2650	2.25	2700	1.00
G663	750	0.00	1200	0.50	2500	0.55	2650	2.25	2750	1.00
G670	750	0.00	1200	0.50	2500	0.60	2600	2.15	2650	1.00
G671	750	0.00	1200	0.50	2500	0.60	2600	2.15	2700	1.00
G673	750	0.00	1200	0.50	2500	0.60	2600	2.20	2650	1.00
G674	750	0.00	1200	0.50	2500	0.60	2600	2.20	2700	1.00
G677	750	0.00	1200	0.50	2500	0.60	2600	2.25	2700	1.00
G679	750	0.00	1200	0.50	2500	0.60	2650	2.15	2700	1.00
G681	750	0.00	1200	0.50	2500	0.60	2650	2.20	2700	1.00
G682	750	0.00	1200	0.50	2500	0.60	2650	2.20	2750	1.00
G683	750	0.00	1200	0.50	2500	0.60	2650	2.25	2700	1.00
G691	750	0.00	1200	0.50	2500	0.65	2600	2.15	2650	1.00
G695	750	0.00	1200	0.50	2500	0.65	2600	2.20	2700	1.00
G697	750	0.00	1200	0.50	2500	0.65	2600	2.25	2650	1.00
G698	750	0.00	1200	0.50	2500	0.65	2600	2.25	2700	1.00
G701	750	0.00	1200	0.50	2500	0.65	2650	2.15	2750	1.00
G702	750	0.00	1200	0.50	2500	0.65	2650	2.20	2700	1.00
G703	750	0.00	1200	0.50	2500	0.65	2650	2.20	2750	1.00
G705	750	0.00	1200	0.50	2500	0.65	2650	2.25	2750	1.00
G713	750	0.00	1200	0.50	2550	0.55	2600	2.15	2700	1.00
G715	750	0.00	1200	0.50	2550	0.55	2600	2.20	2650	1.00
G718	750	0.00	1200	0.50	2550	0.55	2600	2.25	2650	1.00
G719	750	0.00	1200	0.50	2550	0.55	2600	2.25	2700	1.00
G721	750	0.00	1200	0.50	2550	0.55	2650	2.15	2700	1.00
G722	750	0.00	1200	0.50	2550	0.55	2650	2.15	2750	1.00
G723	750	0.00	1200	0.50	2550	0.55	2650	2.20	2700	1.00
G725	750	0.00	1200	0.50	2550	0.55	2650	2.25	2700	1.00
G726	750	0.00	1200	0.50	2550	0.55	2650	2.25	2750	1.00
G730	750	0.00	1200	0.50	2550	0.55	2700	2.20	2750	1.00
G733	750	0.00	1200	0.50	2550	0.60	2600	2.15	2650	1.00
G734	750	0.00	1200	0.50	2550	0.60	2600	2.15	2700	1.00
G739	750	0.00	1200	0.50	2550	0.60	2600	2.25	2650	1.00
G742	750	0.00	1200	0.50	2550	0.60	2650	2.15	2700	1.00
G743	750	0.00	1200	0.50	2550	0.60	2650	2.15	2750	1.00
G745	750	0.00	1200	0.50	2550	0.60	2650	2.20	2750	1.00
G746	750	0.00	1200	0.50	2550	0.60	2650	2.25	2700	1.00
G747	750	0.00	1200	0.50	2550	0.60	2650	2.25	2750	1.00
G749	750	0.00	1200	0.50	2550	0.60	2700	2.15	2750	1.00
G751	750	0.00	1200	0.50	2550	0.60	2700	2.20	2750	1.00
G753	750	0.00	1200	0.50	2550	0.60	2700	2.25	2750	1.00
G754	750	0.00	1200	0.50	2550	0.65	2600	2.15	2650	1.00
G755	750	0.00	1200	0.50	2550	0.65	2600	2.15	2700	1.00
G757	750	0.00	1200	0.50	2550	0.65	2600	2.20	2650	1.00
G758	750	0.00	1200	0.50	2550	0.65	2600	2.20	2700	1.00
G761	750	0.00	1200	0.50	2550	0.65	2600	2.25	2700	1.00
G763	750	0.00	1200	0.50	2550	0.65	2650	2.15	2700	1.00
G765	750	0.00	1200	0.50	2550	0.65	2650	2.20	2700	1.00
G766	750	0.00	1200	0.50	2550	0.65	2650	2.20	2750	1.00
G767	750	0.00	1200	0.50	2550	0.65	2650	2.25	2700	1.00
G770	750	0.00	1200	0.50	2550	0.65	2700	2.15	2750	1.00
G774	750	0.00	1200	0.50	2550	0.65	2700	2.25	2750	1.00
G775	750	0.00	1200	0.50	2600	0.55	2650	2.15	2700	1.00
G777	750	0.00	1200	0.50	2600	0.55	2650	2.20	2700	1.00
G778	750	0.00	1200	0.50	2600	0.55	2650	2.20	2750	1.00
G782	750	0.00	1200	0.50	2600	0.55	2700	2.15	2750	1.00
G786	750	0.00	1200	0.50	2600	0.55	2700	2.25	2750	1.00
G787	750	0.00	1200	0.50	2600	0.60	2650	2.15	2700	1.00
G789	750	0.00	1200	0.50	2600	0.60	2650	2.20	2700	1.00
G790	750	0.00	1200	0.50	2600	0.60	2650	2.20	2750	1.00
G791	750	0.00	1200	0.50	2600	0.60	2650	2.25	2700	1.00
G794	750	0.00	1200	0.50	2600	0.60	2700	2.15	2750	1.00
G798	750	0.00	1200	0.50	2600	0.60	2700	2.25	2750	1.00
G799	750	0.00	1200	0.50	2600	0.65	2650	2.15	2700	1.00
G801	750	0.00	1200	0.50	2600	0.65	2650	2.20	2700	1.00
G802	750	0.00	1200	0.50	2600	0.65	2650	2.20	2750	1.00
G803	750	0.00	1200	0.50	2600	0.65	2650	2.25	2700	1.00
G806	750	0.00	1200	0.50	2600	0.65	2700	2.15	2750	1.00
G810	750	0.00	1200	0.50	2600	0.65	2700	2.25	2750	1.00
G811	750	0.00	1200	0.55	2500	0.55	2600	2.15	2650	1.00
G814	750	0.00	1200	0.55	2500	0.55	2600	2.20	2650	1.00
G815	750	0.00	1200	0.55	2500	0.55	2600	2.20	2700	1.00
G817	750	0.00	1200	0.55	2500	0.55	2600	2.25	2650	1.00
G818	750	0.00	1200	0.55	2500	0.55	2600	2.25	2700	1.00
G821	750	0.00	1200	0.55	2500	0.55	2650	2.15	2750	1.00
G822	750	0.00	1200	0.55	2500	0.55	2650	2.20	2700	1.00
G823	750	0.00	1200	0.55	2500	0.55	2650	2.20	2750	1.00
G825	750	0.00	1200	0.55	2500	0.55	2650	2.25	2750	1.00
G833	750	0.00	1200	0.55	2500	0.60	2600	2.15	2700	1.00
G835	750	0.00	1200	0.55	2500	0.60	2600	2.20	2650	1.00
G838	750	0.00	1200	0.55	2500	0.60	2600	2.25	2650	1.00
G839	750	0.00	1200	0.55	2500	0.60	2600	2.25	2700	1.00
G841	750	0.00	1200	0.55	2500	0.60	2650	2.15	2700	1.00
G842	750	0.00	1200	0.55	2500	0.60	2650	2.15	2750	1.00
G843	750	0.00	1200	0.55	2500	0.60	2650	2.20	2700	1.00
G845	750	0.00	1200	0.55	2500	0.60	2650	2.25	2700	1.00
G846	750	0.00	1200	0.55	2500	0.60	2650	2.25	2750	1.00
G853	750	0.00	1200	0.55	2500	0.65	2600	2.15	2650	1.00
G854	750	0.00	1200	0.55	2500	0.65	2600	2.15	2700	1.00
G857	750	0.00	1200	0.55	2500	0.65	2600	2.20	2700	1.00
G859	750	0.00	1200	0.55	2500	0.65	2600	2.25	2650	1.00
G863	750	0.00	1200	0.55	2500	0.65	2650	2.15	2750	1.00
G865	750	0.00	1200	0.55	2500	0.65	2650	2.20	2750	1.00
G866	750	0.00	1200	0.55	2500	0.65	2650	2.25	2700	1.00
G867	750	0.00	1200	0.55	2500	0.65	2650	2.25	2750	1.00
G874	750	0.00	1200	0.55	2550	0.55	2600	2.15	2650	1.00
G875	750	0.00	1200	0.55	2550	0.55	2600	2.15	2700	1.00
G878	750	0.00	1200	0.55	2550	0.55	2600	2.20	2700	1.00
G881	750	0.00	1200	0.55	2550	0.55	2600	2.25	2700	1.00

G216A	750	0.00	1150	0.50	2500	0.65	2650	2.20	2700	1.00
G236A	750	0.00	1150	0.50	2550	0.55	2650	2.15	2750	1.00
G240A	750	0.00	1150	0.50	2550	0.55	2650	2.25	2750	1.00
G248A	750	0.00	1150	0.50	2550	0.60	2600	2.15	2700	1.00
G2524A	800	0.00	1250	0.45	2550	0.60	2650	2.15	2700	1.00
G2528A	800	0.00	1250	0.45	2550	0.60	2650	2.25	2700	1.00
G2536A	800	0.00	1250	0.45	2550	0.65	2600	2.15	2650	1.00
G2540A	800	0.00	1250	0.45	2550	0.65	2600	2.20	2700	1.00
G2548A	800	0.00	1250	0.45	2550	0.65	2650	2.20	2750	1.00
G2552A	800	0.00	1250	0.45	2550	0.65	2700	2.15	2750	1.00
G2556A	800	0.00	1250	0.45	2550	0.65	2700	2.25	2750	1.00
G2564A	800	0.00	1250	0.45	2600	0.55	2700	2.15	2750	1.00
G2568A	800	0.00	1250	0.45	2600	0.55	2700	2.25	2750	1.00
G256A	750	0.00	1150	0.50	2550	0.60	2650	2.15	2700	1.00
G2572A	800	0.00	1250	0.45	2600	0.60	2650	2.20	2750	1.00
G2576A	800	0.00	1250	0.45	2600	0.60	2700	2.15	2750	1.00
G2580A	800	0.00	1250	0.45	2600	0.60	2700	2.25	2750	1.00
G2584A	800	0.00	1250	0.45	2600	0.65	2650	2.20	2750	1.00
G2588A	800	0.00	1250	0.45	2600	0.65	2700	2.15	2750	1.00
G2592A	800	0.00	1250	0.45	2600	0.65	2700	2.25	2750	1.00
G2596A	800	0.00	1250	0.50	2500	0.55	2600	2.20	2650	1.00
G2600A	800	0.00	1250	0.50	2500	0.55	2600	2.25	2700	1.00
G2604A	800	0.00	1250	0.50	2500	0.55	2650	2.20	2700	1.00
G260A	750	0.00	1150	0.50	2550	0.60	2650	2.25	2700	1.00
G2620A	800	0.00	1250	0.50	2500	0.60	2600	2.25	2650	1.00
G2624A	800	0.00	1250	0.50	2500	0.60	2650	2.15	2750	1.00
G2628A	800	0.00	1250	0.50	2500	0.60	2650	2.25	2750	1.00
G2636A	800	0.00	1250	0.50	2500	0.65	2600	2.15	2700	1.00
G2644A	800	0.00	1250	0.50	2500	0.65	2650	2.15	2700	1.00
G2648A	800	0.00	1250	0.50	2500	0.65	2650	2.25	2700	1.00
G2660A	800	0.00	1250	0.50	2550	0.55	2600	2.20	2700	1.00
G2668A	800	0.00	1250	0.50	2550	0.55	2650	2.20	2750	1.00
G2672A	800	0.00	1250	0.50	2550	0.55	2700	2.15	2750	1.00
G2676A	800	0.00	1250	0.50	2550	0.55	2700	2.25	2750	1.00
G2684A	800	0.00	1250	0.50	2550	0.60	2600	2.25	2700	1.00
G2688A	800	0.00	1250	0.50	2550	0.60	2650	2.20	2700	1.00
G268A	750	0.00	1150	0.50	2550	0.65	2600	2.15	2650	1.00
G2708A	800	0.00	1250	0.50	2550	0.65	2650	2.15	2750	1.00
G2712A	800	0.00	1250	0.50	2550	0.65	2650	2.25	2750	1.00
G2716A	800	0.00	1250	0.50	2550	0.65	2700	2.20	2750	1.00
G2724A	800	0.00	1250	0.50	2600	0.55	2650	2.25	2750	1.00
G2728A	800	0.00	1250	0.50	2600	0.55	2700	2.20	2750	1.00
G272A	750	0.00	1150	0.50	2550	0.65	2600	2.20	2700	1.00
G2732A	800	0.00	1250	0.50	2600	0.60	2650	2.15	2750	1.00
G2736A	800	0.00	1250	0.50	2600	0.60	2650	2.25	2750	1.00
G2740A	800	0.00	1250	0.50	2600	0.60	2700	2.20	2750	1.00
G2752A	800	0.00	1250	0.50	2600	0.65	2700	2.20	2750	1.00
G2756A	800	0.00	1250	0.55	2500	0.55	2600	2.15	2700	1.00
G2764A	800	0.00	1250	0.55	2500	0.55	2650	2.15	2700	1.00
G2768A	800	0.00	1250	0.55	2500	0.55	2650	2.25	2700	1.00
G2776A	800	0.00	1250	0.55	2500	0.60	2600	2.15	2650	1.00
G2780A	800	0.00	1250	0.55	2500	0.60	2600	2.20	2700	1.00
G2788A	800	0.00	1250	0.55	2500	0.60	2650	2.20	2750	1.00
G2800A	800	0.00	1250	0.55	2500	0.65	2600	2.20	2650	1.00
G2804A	800	0.00	1250	0.55	2500	0.65	2600	2.25	2700	1.00
G2808A	800	0.00	1250	0.55	2500	0.65	2650	2.20	2700	1.00
G280A	750	0.00	1150	0.50	2550	0.65	2650	2.20	2750	1.00
G2828A	800	0.00	1250	0.55	2550	0.55	2650	2.15	2750	1.00
G2832A	800	0.00	1250	0.55	2550	0.55	2650	2.25	2750	1.00
G2836A	800	0.00	1250	0.55	2550	0.55	2700	2.20	2750	1.00
G2840A	800	0.00	1250	0.55	2550	0.60	2600	2.15	2700	1.00
G2848A	800	0.00	1250	0.55	2550	0.60	2650	2.15	2700	1.00
G284A	750	0.00	1150	0.50	2550	0.65	2700	2.15	2750	1.00
G2852A	800	0.00	1250	0.55	2550	0.60	2650	2.25	2700	1.00
G2860A	800	0.00	1250	0.55	2550	0.65	2600	2.15	2650	1.00
G2864A	800	0.00	1250	0.55	2550	0.65	2600	2.20	2700	1.00
G2872A	800	0.00	1250	0.55	2550	0.65	2650	2.20	2750	1.00
G2876A	800	0.00	1250	0.55	2550	0.65	2700	2.15	2750	1.00
G2880A	800	0.00	1250	0.55	2550	0.65	2700	2.25	2750	1.00
G2884A	800	0.00	1250	0.55	2600	0.55	2650	2.20	2750	1.00
G2888A	800	0.00	1250	0.55	2600	0.55	2700	2.15	2750	1.00
G288A	750	0.00	1150	0.50	2550	0.65	2700	2.25	2750	1.00
G2892A	800	0.00	1250	0.55	2600	0.55	2700	2.25	2750	1.00
G2896A	800	0.00	1250	0.55	2600	0.60	2650	2.20	2750	1.00
G28A	750	0.00	1150	0.45	2500	0.60	2600	2.25	2650	1.00
G2900A	800	0.00	1250	0.55	2600	0.60	2700	2.15	2750	1.00
G2904A	800	0.00	1250	0.55	2600	0.60	2700	2.25	2750	1.00
G2908A	800	0.00	1250	0.55	2600	0.65	2650	2.20	2750	1.00
G2912A	800	0.00	1250	0.55	2600	0.65	2700	2.15	2750	1.00
G2916A	800	0.00	1250	0.55	2600	0.65	2700	2.25	2750	1.00
G2920A	850	0.00	1150	0.45	2500	0.55	2600	2.20	2650	1.00
G2924A	850	0.00	1150	0.45	2500	0.55	2600	2.25	2700	1.00
G292A	750	0.00	1150	0.50	2600	0.55	2650	2.20	2750	1.00
G2944A	850	0.00	1150	0.45	2500	0.60	2600	2.25	2650	1.00
G2948A	850	0.00	1150	0.45	2500	0.60	2650	2.15	2750	1.00
G2952A	850	0.00	1150	0.45	2500	0.60	2650	2.25	2750	1.00
G2960A	850	0.00	1150	0.45	2500	0.65	2600	2.15	2700	1.00
G2968A	850	0.00	1150	0.45	2500	0.65	2650	2.15	2700	1.00
G296A	750	0.00	1150	0.50	2600	0.55	2700	2.15	2750	1.00
G2972A	850	0.00	1150	0.45	2500	0.65	2650	2.25	2700	1.00
G2980A	850	0.00	1150	0.45	2550	0.55	2600	2.15	2650	1.00
G2992A	850	0.00	1150	0.45	2550	0.55	2650	2.20	2750	1.00
G2996A	850	0.00	1150	0.45	2550	0.55	2700	2.15	2750	1.00
G3000A	850	0.00	1150	0.45	2550	0.55	2700	2.25	2750	1.00
G300A	750	0.00	1150	0.50	2600	0.55	2700	2.25	2750	1.00
G3012A	850	0.00	1150	0.45	2550	0.60	2650	2.20	2700	1.00
G3032A	850	0.00	1150	0.45	2550	0.65	2650	2.15	2750	1.00
G3036A	850	0.00	1150	0.45	2550	0.65	2650	2.25	2750	1.00
G304A	750	0.00	1150	0.50	2600	0.60	2650	2.20	2750	1.00
G3056A	850	0.00	1150	0.45	2600	0.60	2650	2.15	2750	1.00
G3060A	850	0.00	1150	0.45	2600	0.60	2650	2.25	2750	1.00
G3064A	850	0.00	1150	0.45	2600	0.60	2700	2.20	2750	1.00
G3080A	850	0.00	1150	0.50	2500	0.55	2600	2.15	2700	1.00
G3088A	850	0.00	1150	0.50	2500	0.55	2650	2.15	2700	1.00
G308A	750	0.00	1150	0.50	2600	0.60	2700	2.15	2750	1.00
G3092A	850	0.00	1150	0.50	2500	0.55	2650	2.25	2700	1.00
G3104A	850	0.00	1150	0.50	2500	0.60	2600	2.20	2700	1.00
G3112A	850	0.00	1150	0.50	2500	0.60	2650	2.20	2750	1.00
G3124A	850	0.00	1150	0.50	2500	0.65	2600	2.20	2650	1.00

G883	750	0.00	1200	0.55	2550	0.55	2650	2.15	2700	1.00
G885	750	0.00	1200	0.55	2550	0.55	2650	2.20	2700	1.00
G886	750	0.00	1200	0.55	2550	0.55	2650	2.20	2750	1.00
G887	750	0.00	1200	0.55	2550	0.55	2650	2.25	2700	1.00
G890	750	0.00	1200	0.55	2550	0.55	2700	2.15	2750	1.00
G894	750	0.00	1200	0.55	2550	0.55	2700	2.25	2750	1.00
G895	750	0.00	1200	0.55	2550	0.60	2600	2.15	2650	1.00
G898	750	0.00	1200	0.55	2550	0.60	2600	2.20	2650	1.00
G899	750	0.00	1200	0.55	2550	0.60	2600	2.20	2700	1.00
G901	750	0.00	1200	0.55	2550	0.60	2600	2.25	2650	1.00
G902	750	0.00	1200	0.55	2550	0.60	2600	2.25	2700	1.00
G905	750	0.00	1200	0.55	2550	0.60	2650	2.15	2750	1.00
G906	750	0.00	1200	0.55	2550	0.60	2650	2.20	2700	1.00
G907	750	0.00	1200	0.55	2550	0.60	2650	2.20	2750	1.00
G909	750	0.00	1200	0.55	2550	0.60	2650	2.25	2750	1.00
G911	750	0.00	1200	0.55	2550	0.60	2700	2.15	2750	1.00
G913	750	0.00	1200	0.55	2550	0.60	2700	2.20	2750	1.00
G915	750	0.00	1200	0.55	2550	0.60	2700	2.25	2750	1.00
G917	750	0.00	1200	0.55	2550	0.65	2600	2.15	2700	1.00
G919	750	0.00	1200	0.55	2550	0.65	2600	2.20	2650	1.00
G922	750	0.00	1200	0.55	2550	0.65	2600	2.25	2650	1.00
G923	750	0.00	1200	0.55	2550	0.65	2600	2.25	2700	1.00
G925	750	0.00	1200	0.55	2550	0.65	2650	2.15	2700	1.00
G926	750	0.00	1200	0.55	2550	0.65	2650	2.15	2750	1.00
G927	750	0.00	1200	0.55	2550	0.65	2650	2.20	2700	1.00
G929	750	0.00	1200	0.55	2550	0.65	2650	2.25	2700	1.00
G930	750	0.00	1200	0.55	2550	0.65	2650	2.25	2750	1.00
G934	750	0.00	1200	0.55	2550	0.65	2700	2.20	2750	1.00
G937	750	0.00	1200	0.55	2600	0.55	2650	2.15	2700	1.00
G938	750	0.00	1200	0.55	2600	0.55	2650	2.15	2750	1.00
G939	750	0.00	1200	0.55	2600	0.55	2650	2.20	2700	1.00
G941	750	0.00	1200	0.55	2600	0.55	2650	2.25	2700	1.00
G942	750	0.00	1200	0.55	2600	0.55	2650	2.25	2750	1.00
G946	750	0.00	1200	0.55	2600	0.55	2700	2.20	2750	1.00
G949	750	0.00	1200	0.55	2600	0.60	2650	2.15	2700	1.00
G950	750	0.00	1200	0.55	2600	0.60	2650	2.15	2750	1.00
G951	750	0.00	1200	0.55	2600	0.60	2650	2.20	2700	1.00
G953	750	0.00	1200	0.55	2600	0.60	2650	2.25	2700	1.00
G954	750	0.00	1200	0.55	2600	0.60	2650	2.25	2750	1.00
G958	750	0.00	1200	0.55	2600	0.60	2700	2.20	2750	1.00
G962	750	0.00	1200	0.55	2600	0.65	2650	2.15	2750	1.00
G963	750	0.00	1200	0.55	2600	0.65	2650	2.20	2700	1.00
G965	750	0.00	1200	0.55	2600	0.65	2650	2.25	2700	1.00
G966	750	0.00	1200	0.55	2600	0.65	2650	2.25	2750	1.00
G970	750	0.00	1200	0.55	2600	0.65	2700	2.20	2750	1.00
G1459	800	0.00	1150	0.45	2500	0.55	2600	2.15	2650	1.00
G1460	800	0.00	1150	0.45	2500	0.55	2600	2.15	2700	1.00
G1462	800	0.00	1150	0.45	2500	0.55	2600	2.20	2650	1.00
G1463	800	0.00	1150	0.45	2500	0.55	2600	2.20	2700	1.00
G1465	800	0.00	1150	0.45	2500	0.55	2600	2.25	2650	1.00
G1466	800	0.00	1150	0.45	2500	0.55	2600	2.25	2700	1.00
G1469	800	0.00	1150	0.45	2500	0.55	2650	2.15	2750	1.00
G1470	800	0.00	1150	0.45	2500	0.55	2650	2.20	2700	1.00
G1471	800	0.00	1150	0.45	2500	0.55	2650	2.20	2750	1.00
G1473	800	0.00	1150	0.45	2500	0.55	2650	2.25	2750	1.00
G1481	800	0.00	1150	0.45	2500	0.60	2600	2.15	2700	1.00
G1483	800	0.00	1150	0.45	2500	0.60	2600	2.20	2650	1.00
G1486	800	0.00	1150	0.45	2500	0.60	2600	2.25	2650	1.00
G1487	800	0.00	1150	0.45	2500	0.60	2600	2.25	2700	1.00
G1489	800	0.00	1150	0.45	2500	0.60	2650	2.15	2700	1.00
G1490	800	0.00	1150	0.45	2500	0.60	2650	2.15	2750	1.00
G1491	800	0.00	1150	0.45	2500	0.60	2650	2.20	2700	1.00
G1493	800	0.00	1150	0.45	2500	0.60	2650	2.25	2700	1.00
G1494	800	0.00	1150	0.45	2500	0.60	2650	2.25	2750	1.00
G1501	800	0.00	1150	0.45	2500	0.65	2600	2.15	2650	1.00
G1502	800	0.00	1150	0.45	2500	0.65	2600	2.15	2700	1.00
G1505	800	0.00	1150	0.45	2500	0.65	2600	2.20	2700	1.00
G1507	800	0.00	1150	0.45	2500	0.65	2600	2.25	2650	1.00
G1510	800	0.00	1150	0.45	2500	0.65	2650	2.15	2700	1.00
G1511	800	0.00	1150	0.45	2500	0.65	2650	2.15	2750	1.00
G1513	800	0.00	1150	0.45	2500	0.65	2650	2.20	2750	1.00
G1514	800	0.00	1150	0.45	2500	0.65	2650	2.25	2700	1.00
G1515	800	0.00	1150	0.45	2500	0.65	2650	2.25	2750	1.00
G1522	800	0.00	1150	0.45	2550	0.55	2600	2.15	2650	1.00
G1523	800	0.00	1150	0.45	2550	0.55	2600	2.15	2700	1.00
G1525	800	0.00	1150	0.45	2550	0.55	2600	2.20	2650	1.00
G1526	800	0.00	1150	0.45	2550	0.55	2600	2.20	2700	1.00
G1529	800	0.00	1150	0.45	2550	0.55	2600	2.25	2700	1.00
G1531	800	0.00	1150	0.45	2550	0.55	2650	2.15	2700	1.00
G1533	800	0.00	1150	0.45	2550	0.55	2650	2.20	2700	1.00
G1534	800	0.00	1150	0.45	2550	0.55	2650	2.20	2750	1.00
G1535	800	0.00	1150	0.45	2550	0.55	2650	2.25	2700	1.00
G1542	800	0.00	1150	0.45	2550	0.55	2700	2.25	2750	1.00
G1543	800	0.00	1150	0.45	2550	0.60	2600	2.15	2650	1.00
G1546	800	0.00	1150	0.45	2550	0.60	2600	2.20	2650	1.00
G1547	800	0.00	1150	0.45	2550	0.60	2600	2.20	2700	1.00
G1549	800	0.00	1150	0.45	2550	0.60	2600	2.25	2650	1.00
G1550	800	0.00	1150	0.45	2550	0.60	2600	2.25	2700	1.00
G1553	800	0.00	1150	0.45	2550	0.60	2650	2.15	2750	1.00
G1554	800	0.00	1150	0.45	2550	0.60	2650	2.20	2700	1.00
G1555	800	0.00	1150	0.45	2550	0.60	2650	2.20	2750	1.00
G1557	800	0.00	1150	0.45	2550	0.60	2650	2.25	2750	1.00
G1559	800	0.00	1150	0.45	2550	0.60	2700	2.15	2750	1.00
G1563	800	0.00	1150	0.45	2550	0.60	2700	2.25	2750	1.00
G1565	800	0.00	1150	0.45	2550	0.65	2600	2.15	2700	1.00
G1567	800	0.00	1150	0.45	2550	0.65	2600	2.20	2650	1.00
G1570	800	0.00	1150	0.45	2550	0.65	2600	2.25	2650	1.00
G1571	800	0.00	1150	0.45	2550	0.65	2600	2.25	2700	1.00
G1573	800	0.00	1150	0.45	2550	0.65	2650	2.15	2700	1.00
G1574	800	0.00	1150	0.45	2550	0.65	2650	2.15	2750	1.00
G1575	800	0.00	1150	0.45	2550	0.65	2650	2.20	2700	1.00
G1577	800	0.00	1150	0.45	2550	0.65	2650	2.25	2700	1.00
G1578	800	0.00	1150	0.45	2550	0.65	2650	2.25	2750	1.00
G1585	800	0.00	1150	0.45	2600	0.55	2650	2.15	2700	1.00
G1586	800	0.00	1150	0.45	2600	0.55	2650	2.15	2750	1.00
G1587	800	0.00	1150	0.45	2600	0.55	2650	2.20	2700	1.00
G1589	800	0.00	1150	0.45	2600	0.55	2650	2.25	2700	1.00
G1590	800	0.00	1150	0.45	2600	0.55	2650	2.25	2750	1.00
G1597	800	0.00	1150	0.45	2600	0.60	2650	2.15	2700	1.00
G3128A	850	0.00	1150	0.50	2500	0.65	2600	2.25	2700	1.00
G312A	750	0.00	1150	0.50	2600	0.60	2700	2.25	2750	1.00
G3132A	850	0.00	1150	0.50	2500	0.65	2650	2.20	2700	1.00
G3152A	850	0.00	1150	0.50	2550	0.55	2650	2.15	2750	1.00
G3156A	850	0.00	1150	0.50	2550	0.55	2650	2.25	2750	1.00
G3164A	850	0.00	1150	0.50	2550	0.60	2600	2.15	2700	1.00
G316A	750	0.00	1150	0.50	2600	0.65	2650	2.20	2750	1.00
G3172A	850	0.00	1150	0.50	2550	0.60	2650	2.15	2700	1.00
G3176A	850	0.00	1150	0.50	2550	0.60	2650	2.25	2700	1.00
G3184A	850	0.00	1150	0.50	2550	0.65	2600	2.15	2650	1.00
G3188A	850	0.00	1150	0.50	2550	0.65	2600	2.20	2700	1.00
G3196A	850	0.00	1150	0.50	2550	0.65	2650	2.20	2750	1.00
G3200A	850	0.00	1150	0.50	2550	0.65	2700	2.15	2750	1.00
G3204A	850	0.00	1150	0.50	2550	0.65	2700	2.25	2750	1.00
G3208A	850	0.00	1150	0.50	2600	0.55	2650	2.20	2750	1.00
G3212A	850	0.00	1150	0.50	2600	0.55	2700	2.15	2750	1.00
G3216A	850	0.00	1150	0.50	2600	0.55	2700	2.25	2750	1.00
G3220A	850	0.00	1150	0.50	2600	0.60	2650	2.20	2750	1.00
G3224A	850	0.00	1150	0.50	2600	0.60	2700	2.15	2750	1.00
G3228A	850	0.00	1150	0.50	2600	0.60	2700	2.25	2750	1.00
G3232A	850	0.00	1150	0.50	2600	0.65	2650	2.20	2750	1.00
G3236A	850	0.00	1150	0.50	2600	0.65	2700	2.15	2750	1.00
G3240A	850	0.00	1150	0.50	2600	0.65	2700	2.25	2750	1.00
G3244A	850	0.00	1150	0.55	2500	0.55	2600	2.20	2650	1.00
G3248A	850	0.00	1150	0.55	2500	0.55	2600	2.25	2700	1.00
G324A	750	0.00	1150	0.50	2600	0.65	2700	2.25	2750	1.00
G3252A	850	0.00	1150	0.55	2500	0.55	2650	2.20	2700	1.00
G3268A	850	0.00	1150	0.55	2500	0.60	2600	2.25	2650	1.00
G3272A	850	0.00	1150	0.55	2500	0.60	2650	2.15	2750	1.00
G3276A	850	0.00	1150	0.55	2500	0.60	2650	2.25	2750	1.00
G3284A	850	0.00	1150	0.55	2500	0.65	2600	2.15	2700	1.00
G328A	750	0.00	1150	0.55	2500	0.55	2600	2.20	2650	1.00
G3292A	850	0.00	1150	0.55	2500	0.65	2650	2.15	2700	1.00
G3296A	850	0.00	1150	0.55	2500	0.65	2650	2.25	2700	1.00
G32A	750	0.00	1150	0.45	2500	0.60	2650	2.15	2750	1.00
G3304A	850	0.00	1150	0.55	2550	0.55	2600	2.15	2650	1.00
G3308A	850	0.00	1150	0.55	2550	0.55	2600	2.20	2700	1.00
G3316A	850	0.00	1150	0.55	2550	0.55	2650	2.20	2750	1.00
G3320A	850	0.00	1150	0.55	2550	0.55	2700	2.15	2750	1.00
G3324A	850	0.00	1150	0.55	2550	0.55	2700	2.25	2750	1.00
G332A	750	0.00	1150	0.55	2500	0.55	2600	2.25	2700	1.00
G3332A	850	0.00	1150	0.55	2550	0.60	2600	2.25	2700	1.00
G3336A	850	0.00	1150	0.55	2550	0.60	2650	2.20	2700	1.00
G3356A	850	0.00	1150	0.55	2550	0.65	2650	2.15	2750	1.00
G3360A	850	0.00	1150	0.55	2550	0.65	2650	2.25	2750	1.00
G3368A	850	0.00	1150	0.55	2600	0.55	2650	2.15	2750	1.00
G336A	750	0.00	1150	0.55	2500	0.55	2650	2.20	2700	1.00
G3380A	850	0.00	1150	0.55	2600	0.60	2650	2.15	2750	1.00
G3384A	850	0.00	1150	0.55	2600	0.60	2650	2.25	2750	1.00
G3392A	850	0.00	1150	0.55	2600	0.65	2650	2.15	2750	1.00
G3404A	850	0.00	1200	0.45	2500	0.55	2600	2.15	2700	1.00
G3412A	850	0.00	1200	0.45	2500	0.55	2650	2.15	2700	1.00
G3416A	850	0.00	1200	0.45	2500	0.55	2650	2.25	2700	1.00
G3424A	850	0.00	1200	0.45	2500	0.60	2600	2.15	2650	1.00
G3428A	850	0.00	1200	0.45	2500	0.60	2600	2.20	2700	1.00
G3436A	850	0.00	1200	0.45	2500	0.60	2650	2.20	2750	1.00
G3448A	850	0.00	1200	0.45	2500	0.65	2600	2.20	2650	1.00
G3456A	850	0.00	1200	0.45	2500	0.65	2650	2.20	2700	1.00
G3476A	850	0.00	1200	0.45	2550	0.55	2650	2.15	2750	1.00
G3480A	850	0.00	1200	0.45	2550	0.55	2650	2.25	2750	1.00
G3484A	850	0.00	1200	0.45	2550	0.55	2700	2.20	2750	1.00
G3488A	850	0.00	1200	0.45	2550	0.60	2600	2.15	2700	1.00
G3496A	850	0.00	1200	0.45	2550	0.60	2650	2.15	2700	1.00
G3500A	850	0.00	1200	0.45	2550	0.60	2650	2.25	2700	1.00
G3508A	850	0.00	1200	0.45	2550	0.65	2600	2.15	2650	1.00
G3512A	850	0.00	1200	0.45	2550	0.65	2600	2.20	2700	1.00
G3520A	850	0.00	1200	0.45	2550	0.65	2650	2.20	2750	1.00
G3528A	850	0.00	1200	0.45	2550	0.65	2700	2.25	2750	1.00
G352A	750	0.00	1150	0.55	2500	0.60	2600	2.25	2650	1.00
G3536A	850	0.00	1200	0.45	2600	0.55	2700	2.15	2750	1.00
G3540A	850	0.00	1200	0.45	2600	0.55	2700	2.25	2750	1.00
G3544A	850	0.00	1200	0.45	2600	0.60	2650	2.20	2750	1.00
G3548A	850	0.00	1200	0.45	2600	0.60	2700	2.15	2750	1.00
G3552A	850	0.00	1200	0.45	2600	0.60	2700	2.25	2750	1.00
G3560A	850	0.00	1200	0.45	2600	0.65	2700	2.15	2750	1.00
G3564A	850	0.00	1200	0.45	2600	0.65	2700	2.25	2750	1.00
G3568A	850	0.00	1200	0.50	2500	0.55	2600	2.20	2650	1.00
G356A	750	0.00	1150	0.55	2500	0.60	2650	2.15	2750	1.00
G3572A	850	0.00	1200	0.50	2500	0.55	2600	2.25	2700	1.00
G3576A	850	0.00	1200	0.50	2500	0.55	2650	2.20	2700	1.00
G3592A	850	0.00	1200	0.50	2500	0.60	2600	2.25	2650	1.00
G3596A	850	0.00	1200	0.50	2500	0.60	2650	2.15	2750	1.00
G3600A	850	0.00	1200	0.50	2500	0.60	2650	2.25	2750	1.00
G3608A	850	0.00	1200	0.50	2500	0.65	2600	2.15	2700	1.00
G360A	750	0.00	1150	0.55	2500	0.60	2650	2.25	2750	1.00
G3616A	850	0.00	1200	0.50	2500	0.65	2650	2.15	2700	1.00
G3620A	850	0.00	1200	0.50	2500	0.65	2650	2.25	2700	1.00
G3632A	850	0.00	1200	0.50	2550	0.55	2600	2.20	2700	1.00
G3640A	850	0.00	1200	0.50	2550	0.55	2650	2.20	2750	1.00
G3644A	850	0.00	1200	0.50	2550	0.55	2700	2.15	2750	1.00
G3648A	850	0.00	1200	0.50	2550	0.55	2700	2.25	2750	1.00
G3656A	850	0.00	1200	0.50	2550	0.60	2600	2.25	2700	1.00
G3660A	850	0.00	1200	0.50	2550	0.60	2650	2.20	2700	1.00
G3680A	850	0.00	1200	0.50	2550	0.65	2650	2.15	2750	1.00
G3684A	850	0.00	1200	0.50	2550	0.65	2650	2.25	2750	1.00
G3688A	850	0.00	1200	0.50	2550	0.65	2700	2.20	2750	1.00
G368A	750	0.00	1150	0.55	2500	0.65	2600	2.15	2700	1.00
G3692A	850	0.00	1200	0.50	2600	0.55	2650	2.15	2750	1.00
G36A	750	0.00	1150	0.45	2500	0.60	2650	2.25	2750	1.00
G3704A	850	0.00	1200	0.50	2600	0.60	2650	2.15	2750	1.00
G3708A	850	0.00	1200	0.50	2600	0.60	2650	2.25	2750	1.00
G3712A	850	0.00	1200	0.50	2600	0.60	2700	2.20	2750	1.00
G3724A	850	0.00	1200	0.50	2600	0.65	2700	2.20	2750	1.00
G3728A	850	0.00	1200	0.55	2500	0.55	2600	2.15	2700	1.00
G3736A	850	0.00	1200	0.55	2500	0.55	2650	2.15	2700	1.00
G3740A	850	0.00	1200	0.55	2500	0.55	2650	2.25	2700	1.00
G3748A	850	0.00	1200	0.55	2500	0.60	2600	2.15	2650	1.00
G3752A	850	0.00	1200	0.55	2500	0.60	2600	2.20	2700	1.00
G3760A	850	0.00	1200	0.55	2500	0.60	2650	2.20	2750	1.00

G1599	800	0.00	1150	0.45	2600	0.60	2650	2.20	2700	1.00
G1601	800	0.00	1150	0.45	2600	0.60	2650	2.25	2700	1.00
G1602	800	0.00	1150	0.45	2600	0.60	2650	2.25	2750	1.00
G1606	800	0.00	1150	0.45	2600	0.60	2700	2.20	2750	1.00
G1609	800	0.00	1150	0.45	2600	0.65	2650	2.15	2700	1.00
G1611	800	0.00	1150	0.45	2600	0.65	2650	2.20	2700	1.00
G1613	800	0.00	1150	0.45	2600	0.65	2650	2.25	2700	1.00
G1614	800	0.00	1150	0.45	2600	0.65	2650	2.25	2750	1.00
G1618	800	0.00	1150	0.45	2600	0.65	2700	2.20	2750	1.00
G1622	800	0.00	1150	0.50	2500	0.55	2600	2.15	2700	1.00
G1625	800	0.00	1150	0.50	2500	0.55	2600	2.20	2700	1.00
G1627	800	0.00	1150	0.50	2500	0.55	2600	2.25	2650	1.00
G1630	800	0.00	1150	0.50	2500	0.55	2650	2.15	2700	1.00
G1633	800	0.00	1150	0.50	2500	0.55	2650	2.20	2750	1.00
G1634	800	0.00	1150	0.50	2500	0.55	2650	2.25	2700	1.00
G1643	800	0.00	1150	0.50	2500	0.60	2600	2.15	2700	1.00
G1645	800	0.00	1150	0.50	2500	0.60	2600	2.20	2650	1.00
G1646	800	0.00	1150	0.50	2500	0.60	2600	2.20	2700	1.00
G1651	800	0.00	1150	0.50	2500	0.60	2650	2.15	2700	1.00
G1653	800	0.00	1150	0.50	2500	0.60	2650	2.20	2700	1.00
G1654	800	0.00	1150	0.50	2500	0.60	2650	2.20	2750	1.00
G1655	800	0.00	1150	0.50	2500	0.60	2650	2.25	2700	1.00
G1666	800	0.00	1150	0.50	2500	0.65	2600	2.20	2650	1.00
G1667	800	0.00	1150	0.50	2500	0.65	2600	2.20	2700	1.00
G1669	800	0.00	1150	0.50	2500	0.65	2600	2.25	2650	1.00
G1670	800	0.00	1150	0.50	2500	0.65	2600	2.25	2700	1.00
G1673	800	0.00	1150	0.50	2500	0.65	2650	2.15	2750	1.00
G1674	800	0.00	1150	0.50	2500	0.65	2650	2.20	2700	1.00
G1675	800	0.00	1150	0.50	2500	0.65	2650	2.20	2750	1.00
G1677	800	0.00	1150	0.50	2500	0.65	2650	2.25	2750	1.00
G1685	800	0.00	1150	0.50	2550	0.55	2600	2.15	2700	1.00
G1687	800	0.00	1150	0.50	2550	0.55	2600	2.20	2650	1.00
G1690	800	0.00	1150	0.50	2550	0.55	2600	2.25	2650	1.00
G1691	800	0.00	1150	0.50	2550	0.55	2600	2.25	2700	1.00
G1693	800	0.00	1150	0.50	2550	0.55	2650	2.15	2700	1.00
G1694	800	0.00	1150	0.50	2550	0.55	2650	2.15	2750	1.00
G1695	800	0.00	1150	0.50	2550	0.55	2650	2.20	2700	1.00
G1697	800	0.00	1150	0.50	2550	0.55	2650	2.25	2700	1.00
G1698	800	0.00	1150	0.50	2550	0.55	2650	2.25	2750	1.00
G1705	800	0.00	1150	0.50	2550	0.60	2600	2.15	2650	1.00
G1706	800	0.00	1150	0.50	2550	0.60	2600	2.15	2700	1.00
G1709	800	0.00	1150	0.50	2550	0.60	2600	2.20	2700	1.00
G1711	800	0.00	1150	0.50	2550	0.60	2600	2.25	2650	1.00
G1714	800	0.00	1150	0.50	2550	0.60	2650	2.15	2700	1.00
G1715	800	0.00	1150	0.50	2550	0.60	2650	2.15	2750	1.00
G1717	800	0.00	1150	0.50	2550	0.60	2650	2.20	2750	1.00
G1718	800	0.00	1150	0.50	2550	0.60	2650	2.25	2700	1.00
G1719	800	0.00	1150	0.50	2550	0.60	2650	2.25	2750	1.00
G1721	800	0.00	1150	0.50	2550	0.60	2700	2.15	2750	1.00
G1725	800	0.00	1150	0.50	2550	0.60	2700	2.25	2750	1.00
G1726	800	0.00	1150	0.50	2550	0.65	2600	2.15	2650	1.00
G1727	800	0.00	1150	0.50	2550	0.65	2600	2.15	2700	1.00
G1729	800	0.00	1150	0.50	2550	0.65	2600	2.20	2650	1.00
G1730	800	0.00	1150	0.50	2550	0.65	2600	2.20	2700	1.00
G1733	800	0.00	1150	0.50	2550	0.65	2600	2.25	2700	1.00
G1735	800	0.00	1150	0.50	2550	0.65	2650	2.15	2700	1.00
G1737	800	0.00	1150	0.50	2550	0.65	2650	2.20	2700	1.00
G1739	800	0.00	1150	0.50	2550	0.65	2650	2.25	2700	1.00
G1742	800	0.00	1150	0.50	2550	0.65	2700	2.15	2750	1.00
G1746	800	0.00	1150	0.50	2550	0.65	2700	2.25	2750	1.00
G1747	800	0.00	1150	0.50	2600	0.55	2650	2.15	2700	1.00
G1749	800	0.00	1150	0.50	2600	0.55	2650	2.20	2700	1.00
G1750	800	0.00	1150	0.50	2600	0.55	2650	2.20	2750	1.00
G1751	800	0.00	1150	0.50	2600	0.55	2650	2.25	2700	1.00
G1754	800	0.00	1150	0.50	2600	0.55	2700	2.15	2750	1.00
G1758	800	0.00	1150	0.50	2600	0.55	2700	2.25	2750	1.00
G1759	800	0.00	1150	0.50	2600	0.60	2650	2.15	2700	1.00
G1761	800	0.00	1150	0.50	2600	0.60	2650	2.20	2700	1.00
G1762	800	0.00	1150	0.50	2600	0.60	2650	2.20	2750	1.00
G1763	800	0.00	1150	0.50	2600	0.60	2650	2.25	2700	1.00
G1766	800	0.00	1150	0.50	2600	0.60	2700	2.15	2750	1.00
G1770	800	0.00	1150	0.50	2600	0.60	2700	2.25	2750	1.00
G1771	800	0.00	1150	0.50	2600	0.65	2650	2.15	2700	1.00
G1773	800	0.00	1150	0.50	2600	0.65	2650	2.20	2700	1.00
G1774	800	0.00	1150	0.50	2600	0.65	2650	2.20	2750	1.00
G1775	800	0.00	1150	0.50	2600	0.65	2650	2.25	2700	1.00
G1778	800	0.00	1150	0.50	2600	0.65	2700	2.15	2750	1.00
G1782	800	0.00	1150	0.50	2600	0.65	2700	2.25	2750	1.00
G1786	800	0.00	1150	0.55	2500	0.55	2600	2.20	2650	1.00
G1787	800	0.00	1150	0.55	2500	0.55	2600	2.20	2700	1.00
G1789	800	0.00	1150	0.55	2500	0.55	2600	2.25	2650	1.00
G1790	800	0.00	1150	0.55	2500	0.55	2600	2.25	2700	1.00
G1792	800	0.00	1150	0.55	2500	0.55	2650	2.15	2700	1.00
G1793	800	0.00	1150	0.55	2500	0.55	2650	2.15	2750	1.00
G1794	800	0.00	1150	0.55	2500	0.55	2650	2.20	2700	1.00
G1795	800	0.00	1150	0.55	2500	0.55	2650	2.20	2750	1.00
G1796	800	0.00	1150	0.55	2500	0.55	2650	2.25	2700	1.00
G1797	800	0.00	1150	0.55	2500	0.55	2650	2.25	2750	1.00
G1805	800	0.00	1150	0.55	2500	0.60	2600	2.15	2700	1.00
G1807	800	0.00	1150	0.55	2500	0.60	2600	2.20	2650	1.00
G1808	800	0.00	1150	0.55	2500	0.60	2600	2.20	2700	1.00
G1810	800	0.00	1150	0.55	2500	0.60	2600	2.25	2650	1.00
G1813	800	0.00	1150	0.55	2500	0.60	2650	2.15	2700	1.00
G1814	800	0.00	1150	0.55	2500	0.60	2650	2.15	2750	1.00
G1815	800	0.00	1150	0.55	2500	0.60	2650	2.20	2700	1.00
G1816	800	0.00	1150	0.55	2500	0.60	2650	2.20	2750	1.00
G1817	800	0.00	1150	0.55	2500	0.60	2650	2.25	2700	1.00
G1818	800	0.00	1150	0.55	2500	0.60	2650	2.25	2750	1.00
G1826	800	0.00	1150	0.55	2500	0.65	2600	2.15	2700	1.00
G1831	800	0.00	1150	0.55	2500	0.65	2600	2.25	2650	1.00
G1832	800	0.00	1150	0.55	2500	0.65	2600	2.25	2700	1.00
G1834	800	0.00	1150	0.55	2500	0.65	2650	2.15	2700	1.00
G1835	800	0.00	1150	0.55	2500	0.65	2650	2.15	2750	1.00
G1836	800	0.00	1150	0.55	2500	0.65	2650	2.20	2700	1.00
G1837	800	0.00	1150	0.55	2500	0.65	2650	2.20	2750	1.00
G1838	800	0.00	1150	0.55	2500	0.65	2650	2.25	2700	1.00
G1839	800	0.00	1150	0.55	2500	0.65	2650	2.25	2750	1.00
G1846	800	0.00	1150	0.55	2550	0.55	2600	2.15	2650	1.00
G1847	800	0.00	1150	0.55	2550	0.55	2600	2.15	2700	1.00
G376A	750	0.00	1150	0.55	2500	0.65	2650	2.15	2700	1.00
G3772A	850	0.00	1200	0.55	2500	0.65	2600	2.20	2650	1.00
G3776A	850	0.00	1200	0.55	2500	0.65	2600	2.25	2700	1.00
G3780A	850	0.00	1200	0.55	2500	0.65	2650	2.20	2700	1.00
G3800A	850	0.00	1200	0.55	2550	0.55	2650	2.15	2750	1.00
G3804A	850	0.00	1200	0.55	2550	0.55	2650	2.25	2750	1.00
G3808A	850	0.00	1200	0.55	2550	0.55	2700	2.20	2750	1.00
G380A	750	0.00	1150	0.55	2500	0.65	2650	2.25	2700	1.00
G3812A	850	0.00	1200	0.55	2550	0.60	2600	2.15	2700	1.00
G3820A	850	0.00	1200	0.55	2550	0.60	2650	2.15	2700	1.00
G3824A	850	0.00	1200	0.55	2550	0.60	2650	2.25	2700	1.00
G3832A	850	0.00	1200	0.55	2550	0.65	2600	2.15	2650	1.00
G3836A	850	0.00	1200	0.55	2550	0.65	2600	2.20	2700	1.00
G3844A	850	0.00	1200	0.55	2550	0.65	2650	2.20	2750	1.00
G3848A	850	0.00	1200	0.55	2550	0.65	2700	2.15	2750	1.00
G3852A	850	0.00	1200	0.55	2550	0.65	2700	2.25	2750	1.00
G3856A	850	0.00	1200	0.55	2600	0.55	2650	2.20	2750	1.00
G392A	750	0.00	1150	0.55	2550	0.55	2600	2.20	2700	1.00
G400A	750	0.00	1150	0.55	2550	0.55	2650	2.20	2750	1.00
G404A	750	0.00	1150	0.55	2550	0.55	2700	2.15	2750	1.00
G408A	750	0.00	1150	0.55	2550	0.55	2700	2.25	2750	1.00
G412A	750	0.00	1150	0.55	2550	0.60	2600	2.20	2650	1.00
G416A	750	0.00	1150	0.55	2550	0.60	2600	2.25	2700	1.00
G420A	750	0.00	1150	0.55	2550	0.60	2650	2.20	2700	1.00
G440A	750	0.00	1150	0.55	2550	0.65	2650	2.15	2750	1.00
G444A	750	0.00	1150	0.55	2550	0.65	2650	2.25	2750	1.00
G44A	750	0.00	1150	0.45	2500	0.65	2600	2.15	2700	1.00
G452A	750	0.00	1150	0.55	2600	0.55	2650	2.15	2750	1.00
G460A	750	0.00	1150	0.55	2600	0.55	2700	2.20	2750	1.00
G464A	750	0.00	1150	0.55	2600	0.60	2650	2.15	2750	1.00
G468A	750	0.00	1150	0.55	2600	0.60	2650	2.25	2750	1.00
G476A	750	0.00	1150	0.55	2600	0.65	2650	2.15	2750	1.00
G496A	750	0.00	1200	0.45	2500	0.55	2650	2.15	2700	1.00
G4A	750	0.00	1150	0.45	2500	0.55	2600	2.20	2650	1.00
G500A	750	0.00	1200	0.45	2500	0.55	2650	2.25	2700	1.00
G508A	750	0.00	1200	0.45	2500	0.60	2600	2.15	2650	1.00
G512A	750	0.00	1200	0.45	2500	0.60	2600	2.20	2700	1.00
G520A	750	0.00	1200	0.45	2500	0.60	2650	2.20	2750	1.00
G52A	750	0.00	1150	0.45	2500	0.65	2650	2.15	2700	1.00
G532A	750	0.00	1200	0.45	2500	0.65	2600	2.20	2650	1.00
G536A	750	0.00	1200	0.45	2500	0.65	2600	2.25	2700	1.00
G540A	750	0.00	1200	0.45	2500	0.65	2650	2.20	2700	1.00
G560A	750	0.00	1200	0.45	2550	0.55	2650	2.15	2750	1.00
G564A	750	0.00	1200	0.45	2550	0.55	2650	2.25	2750	1.00
G568A	750	0.00	1200	0.45	2550	0.55	2700	2.20	2750	1.00
G56A	750	0.00	1150	0.45	2500	0.65	2650	2.25	2700	1.00
G572A	750	0.00	1200	0.45	2550	0.60	2600	2.15	2700	1.00
G580A	750	0.00	1200	0.45	2550	0.60	2650	2.15	2700	1.00
G584A	750	0.00	1200	0.45	2550	0.60	2650	2.25	2700	1.00
G592A	750	0.00	1200	0.45	2550	0.65	2600	2.15	2650	1.00
G596A	750	0.00	1200	0.45	2550	0.65	2600	2.20	2700	1.00
G604A	750	0.00	1200	0.45	2550	0.65	2650	2.20	2750	1.00
G608A	750	0.00	1200	0.45	2550	0.65	2700	2.15	2750	1.00
G612A	750	0.00	1200	0.45	2550	0.65	2700	2.25	2750	1.00
G616A	750	0.00	1200	0.45	2600	0.55	2650	2.20	2750	1.00
G620A	750	0.00	1200	0.45	2600	0.55	2700	2.15	2750	1.00
G624A	750	0.00	1200	0.45	2600	0.55	2700	2.25	2750	1.00
G628A	750	0.00	1200	0.45	2600	0.60	2650	2.20	2750	1.00
G632A	750	0.00	1200	0.45	2600	0.60	2700	2.15	2750	1.00
G636A	750	0.00	1200	0.45	2600	0.60	2700	2.25	2750	1.00
G644A	750	0.00	1200	0.45	2600	0.65	2700	2.15	2750	1.00
G648A	750	0.00	1200	0.45	2600	0.65	2700	2.25	2750	1.00
G64A	750	0.00	1150	0.45	2550	0.55	2600	2.15	2650	1.00
G652A	750	0.00	1200	0.50	2500	0.55	2600	2.20	2650	1.00
G656A	750	0.00	1200	0.50	2500	0.55	2600	2.25	2700	1.00
G660A	750	0.00	1200	0.50	2500	0.55	2650	2.20	2700	1.00
G676A	750	0.00	1200	0.50	2500	0.60	2600	2.25	2650	1.00
G680A	750	0.00	1200	0.50	2500	0.60	2650	2.15	2750	1.00
G684A	750	0.00	1200	0.50	2500	0.60	2650	2.25	2750	1.00
G692A	750	0.00	1200	0.50	2500	0.65	2600	2.15	2700	1.00
G700A	750	0.00	1200	0.50	2500	0.65	2650	2.15	2700	1.00
G704A	750	0.00	1200	0.50	2500	0.65	2650	2.25	2700	1.00
G716A	750	0.00	1200	0.50	2550	0.55	2600	2.20	2700	1.00
G724A	750	0.00	1200	0.50	2550	0.55	2650	2.20	2750	1.00
G732A	750	0.00	1200	0.50	2550	0.55	2700	2.25	2750	1.00
G740A	750	0.00	1200	0.50	2550	0.60	2600	2.25	2700	1.00
G744A	750	0.00	1200	0.50	2550	0.60	2650	2.20	2700	1.00
G764A	750	0.00	1200	0.50	2550	0.65	2650	2.15	2750	1.00
G768A	750	0.00	1200	0.50	2550	0.65	2650	2.25	2750	1.00
G76A	750	0.00	1150	0.45	2550	0.55	2650	2.20	2750	1.00
G772A	750	0.00	1200	0.50	2550	0.65	2700	2.20	2750	1.00
G776A	750	0.00	1200	0.50	2600	0.55	2650	2.15	2750	1.00
G784A	750	0.00	1200	0.50	2600	0.55	2700	2.20	2750	1.00
G788A	750	0.00	1200	0.50	2600	0.60	2650	2.15	2750	1.00
G792A	750	0.00	1200	0.50	2600	0.60	2650	2.25	2750	1.00
G796A	750	0.00	1200	0.50	2600	0.60	2700	2.20	2750	1.00
G808A	750	0.00	1200	0.50	2600	0.65	2700	2.20	2750	1.00
G80A	750	0.00	1150	0.45	2550	0.55	2700	2.15	2750	1.00
G812A	750	0.00	1200	0.55	2500	0.55	2600	2.15	2700	1.00
G820A	750	0.00	1200	0.55	2500	0.55	2650	2.15	2700	1.00
G824A	750	0.00	1200	0.55	2500	0.55	2650	2.25	2700	1.00
G832A	750	0.00	1200	0.55	2500	0.60	2600	2.15	2650	1.00
G836A	750	0.00	1200	0.55	2500	0.60	2600	2.20	2700	1.00
G844A	750	0.00	1200	0.55	2500	0.60	2650	2.20	2750	1.00
G84A	750	0.00	1150	0.45	2550	0.55	2700	2.25	2750	1.00
G856A	750	0.00	1200	0.55	2500	0.65	2600	2.20	2650	1.00
G860A	750	0.00	1200	0.55	2500	0.65	2600	2.25	2700	1.00
G864A	750	0.00	1200	0.55	2500	0.65	2650	2.20	2700	1.00
G884A	750	0.00	1200	0.55	2550	0.55	2650	2.15	2750	1.00
G888A	750	0.00	1200	0.55	2550	0.55	2650	2.25	2750	1.00
G892A	750	0.00	1200	0.55	2550	0.55	2700	2.20	2750	1.00
G896A	750	0.00	1200	0.55	2550	0.60	2600	2.15	2700	1.00
G8A	750	0.00	1150	0.45	2500	0.55	2600	2.25	2700	1.00
G904A	750	0.00	1200	0.55	2550	0.60	2650	2.15	2700	1.00
G908A	750	0.00	1200	0.55	2550	0.60	2650	2.25	2700	1.00
G916A	750	0.00	1200	0.55	2550	0.65	2600	2.15	2650	1.00
G920A	750	0.00	1200	0.55	2550	0.65	2600	2.20	2700	1.00
G928A	750	0.00	1200	0.55	2550	0.65	2650	2.20	2750	1.00
G92A	750	0.00	1150	0.45	2550	0.60	2600	2.25	2700	1.00

G1849	800	0.00	1150	0.55	2550	0.55	2600	2.20	2650	1.00
G1850	800	0.00	1150	0.55	2550	0.55	2600	2.20	2700	1.00
G1852	800	0.00	1150	0.55	2550	0.55	2600	2.25	2650	1.00
G1853	800	0.00	1150	0.55	2550	0.55	2600	2.25	2700	1.00
G1855	800	0.00	1150	0.55	2550	0.55	2650	2.15	2700	1.00
G1856	800	0.00	1150	0.55	2550	0.55	2650	2.15	2750	1.00
G1857	800	0.00	1150	0.55	2550	0.55	2650	2.20	2700	1.00
G1858	800	0.00	1150	0.55	2550	0.55	2650	2.20	2750	1.00
G1859	800	0.00	1150	0.55	2550	0.55	2650	2.25	2700	1.00
G1860	800	0.00	1150	0.55	2550	0.55	2650	2.25	2750	1.00
G1862	800	0.00	1150	0.55	2550	0.55	2700	2.15	2750	1.00
G1866	800	0.00	1150	0.55	2550	0.55	2700	2.25	2750	1.00
G932A	750	0.00	1200	0.55	2550	0.65	2700	2.15	2750	1.00
G936A	750	0.00	1200	0.55	2550	0.65	2700	2.25	2750	1.00
G940A	750	0.00	1200	0.55	2600	0.55	2650	2.20	2750	1.00
G944A	750	0.00	1200	0.55	2600	0.55	2700	2.15	2750	1.00
G948A	750	0.00	1200	0.55	2600	0.55	2700	2.25	2750	1.00
G952A	750	0.00	1200	0.55	2600	0.60	2650	2.20	2750	1.00
G956A	750	0.00	1200	0.55	2600	0.60	2700	2.15	2750	1.00
G960A	750	0.00	1200	0.55	2600	0.60	2700	2.25	2750	1.00
G964A	750	0.00	1200	0.55	2600	0.65	2650	2.20	2750	1.00
G968A	750	0.00	1200	0.55	2600	0.65	2700	2.15	2750	1.00
G96A	750	0.00	1150	0.45	2550	0.60	2650	2.20	2700	1.00
G972A	750	0.00	1200	0.55	2600	0.65	2700	2.25	2750	1.00

Appendix D Software Engineering and Object Orientated Analysis

D.1 Introduction

Software analysis is distinct from software design in that it models the application problem rather than its solution. It is thus divorced from any particular implementation language, but may still follow a particular software paradigm. The ray-tracer application uses object-orientated principles in both its analysis and code. A complete discussion of the Shlaer-Mellor object-orientated software analysis methodology used here lies well beyond the scope of this thesis and the reader is referred to Shlaer & Mellor [Shl88] and Shlaer & Mellor [Shl92]. A few salient details are presented below to facilitate comprehension of the diagrams and tables which follow.

D.2 Information Model and Data Dictionary

Complex problems are first divided into several domains for manageability, abstraction and reuse. The objects in each domain are identified and isolated, together with their attributes, in an information model. Attributes preceded by an '@' are keys - these are attributes whose unique values distinguish between different instances of the same object. Associations between objects are depicted as relationships which may exhibit both conditionality (indicated by a 'c') and multiplicity (indicated by a '1' or 'M' for 'many'). An initial identification of objects and relations may be made by noting the nouns and verbs in a description of the problem domain. Objects are represented by rectangles and have a unique number and key letters, while attributes are shown as text within the rectangles and relationships by labelled diamonds. The relationship of object A to object B is indicated on the *far* side of the diamond connecting object A to object B (i.e. the relationship of object A to object B is indicated next to object B and vice versa). Relationships are formalised if an attribute of one object refers to a key attribute of the other via that relationship. This is indicated by writing the relationship in brackets after the attribute. For one to many relationships, this referential attribute always occurs on the 'many' side. Many to many relationships are formalised by an additional associative object which links together specific instances of those objects participating in the relationship. Associative objects are connected to the diamond relationship by an arrow. Inheritance is indicated by a subtype/supertype relationship labelled '---isa---' and the ownership by one object of another by 'has a' relationships. General information about every object and each of its attributes (such as its purpose and range of permissible values) is stored in a data dictionary.

D.3 State Model and Event List

Objects in the information model which have a STATE attribute behave dynamically by existing in different states at different times. Such lifecycle behaviour is captured in a state model. A separate state model is created for each dynamic or active object in the information model, including those which have a single temporal state. States are indicated by rectangles and the events which cause legal transitions between them are indicated by labelled arrows. Event names are prefixed by the key-letters of the object that they belong to. These events may be called or generated by the object's own states or those of totally different objects. Events may carry data, corresponding to some of the object's attributes, into a state. The actions which a state executes when it is entered are written within each rectangle, largely in free-format text. Keywords which are preceded by a percentage sign (such as

%create, %delete and %generate) have special significance when the CASE tool performs error checking. The actions of a state may perform any calculation, generate an event to any instance of any object within or external to the scope of the analysis, and read and/or write attributes of instances of its own or other objects. An action must leave the instance consistent, ensure the consistency of relationships and subtypes and update the current state attribute. Only one action of a given state machine can execute at any instant. Initiated actions must be completed before another event can be received by the same instance of the state machine. Actions in different state machines though, can be executing simultaneously. Events are never lost - if an event is generated to an instance that is busy, the event will not be accepted until the current action is completed. All generated events are consumed when submitted to a state machine. A state is known as transient if its action list always generates an event which takes the object into a new state. All the processing of the system is contained within the actions of its state models. Technically the state machines described here are of the Moore form [Moo56]. The events generated by all the dynamic objects are collated into an event list which lists the event's name, parameters, owner and users.

D.4 Process Model and Process Specifications

A more detailed process model may be created for each state in the state model. The process model expands on the actions carried out by an object when it enters a particular state. A process is a fundamental unit of operation and is represented by a circle which contains the process name, prefixed with the key-letters of the parent object. The data flow diagrams are fairly similar to those presented by DeMarco [DeM78]. The flow of data between processes is indicated by labelled solid arrows. Control flows are represented by dashed arrows and may be either conditional (labelled) or unconditional. All the data that an event introduces to an object's state is available to every process that belongs to the object. Events that a process may generate are indicated as flows, labelled with the event name and parameters, that are directed into empty space. Processes may additionally access object data stores if they possess the object's key attribute(s). Data stores are represented by horizontal parallel lines and store only persistent data, (data which continues to exist after an action is complete). Such data corresponds to the object attributes in the Information Model. Transient data produced by processes is annotated as '(transient)' and conditional data as '(conditional)'. Data that is known by different names to different objects is shown using a producer-consumer notation. A process may only execute when it has received all its indicated data and control inputs. The flow of control is thus often implicit in the flow of data and need only be explicitly shown where necessary. External devices, such as sensors and interfaces, are represented by rectangles known as terminators. Processes may communicate with such terminators via informally labelled flows.

There are four general types of processes:

(a) Accessors, further divided into read, write, create and delete forms.
(b) Event Generators, which generate a single event.
(c) Transformations, which compute, output, and optionally store, new data and
(d) Tests, which test a condition and make one of several possible conditional control outputs.

A process specification formalises the interface of a process by listing its input and output data, generated events, conditional output data and conditional output control as well as any read and write database accesses. Conditional and unconditional controls into a process are not shown on process specifications to enhance reuse. Read and write accessor processes are provided for each object. These show all the possible data that may be read from or written to the object, but particular instances of the process are at liberty to use only those data that they need - i.e. the process specification interface may be a superset. The process specification body includes an informal description of the process, its algorithms, formulae and any additional information that may be useful during the coding phase. It is at this level that opportunities for reuse are most readily identified.

D.5 Translation to Code

In the Shlaer-Mellor methodology the distinction between analysis and design starts to blur at the process model. The analysis is so detailed at this level that it is possible to omit the design phase and produce code directly by using a transformational approach [Hil90].

Strict enforcement of legal state transitions can be accomplished by examining the existing state of an object instance before attempting to transit to (i.e. execute the actions of) a new one. One scheme, attributable to Hill [Hil90], is to code the events and state actions of the state model as separate procedures. State actions would be private and hidden from external callers. The visible events would call the corresponding state actions only if the object instance they were considering had the correct state. Semaphore protection of shared data from simultaneous access would also be handled in the event procedure.

```
procedure DRAW_EVENT( var INSTANCE : GRAPH );
begin
  SEMAPHORE.LOCK;
  case INSTANCE.STATE of
      CLEARED      : DRAWN_ACTION( INSTANCE );
      DRAWN, DONE  : CLEAR_ACTION( INSTANCE );
      TEXT_MODE    : SET_GRAPHICS_MODE( INSTANCE );
      else ERROR_MESSAGE;
  end; {case}
  SEMAPHORE.RELEASE;
end; {DRAW_EVENT}

procedure DRAWN_ACTION( var INSTANCE : GRAPH );
begin
  ...
  INSTANCE.STATE := DRAWN;
end; {DRAWN_ACTION}
```

Such rigour is standard practice for systems with multi-tasking features, embedded control responsibilities, hard real-time constraints or mission critical robustness requirements. Since the ray-tracer has none of these requirements we take the liberty of interpreting the legal state transitions as instructions for the proper use of the objects by calling routines, and enforce them only in certain cases (such as setting the graphics mode before trying to draw graphs).

In general each object in the analysis becomes an object in the code although objects with inheritance or ownership relationships may be merged. The attributes of each analysed object become private or public object data in the code. Each visible analysis event becomes a public object method accepting only that data described in the event interface. Some local events may be coded as private methods. Processes from the process model may also be coded as private methods or as local procedures within the object. Procedure calls and data exchange between objects are governed by the event generations indicated in the analysis.

The coded objects and supplementary packages are graphically represented by structure graphs conforming to the Bhur [Bhu84] notation but limited to Pascal-type structures. The Bhur notation was derived for the Ada programming language, but since Ada is based on Pascal we can still utilise a subset of the notation in this project. In these diagrams, rectangles represent units, procedures or functions; arrows into procedures or functions represent invocations and couples with directed arrowheads on the invocation represent variable or value parameters. Dashed rhombuses represent data types and solid rhombuses variables or constants of some data type. Invocations with double arrowheads represent the use by the unit concerned of other units via the Pascal 'uses' keyword. Rectangles representing such 'used' units have a diagonal slash (a disposition) if they are provided by the compiler rather than the user. Dashed invocations represent a specific instantiation of some generic mechanism - in our case the creation of an instance of an object class. The CASE tool uses Bhur structure graphs to automatically generate Ada code. In practice, additional, more detailed, structure graphs are created for each procedure indicated on the package or unit level diagram.

D.6 Software Engineering and Neural Networks

It is interesting to note that some researchers are extracting finite state machines, similar to those presented in this appendix, from the internal states of neural networks. Kolen [Kol94] reviews some of these attempts and discusses their shortcomings.

D.7 Symbology

Table D.1 Symbology

SYMBOL	ERD	STD	DFD
□	object	state	terminator
◇	relationship	-	-
————————	connection	-	-
————————>	associative object connection	event	data flow or generated event
○	-	-	process
════════	-	-	data store
— — — — — — —>	-	-	unconditional control flow
M	multiplicity	-	-
c	conditionality	-	-
R	relation	-	-
@	key attribute	-	-
E(X,Y) ————————>	-	event	event
X+Y ————————>	-	-	data flow
X (transient) ————————>	-	-	transient data
X (conditional) ————————>	-	-	conditional data
X — — — — — — —>	-	-	conditional control flow
A.X = B.Y	-	-	producer-consumer

D.8 Information Model and Entity Relationship Diagram

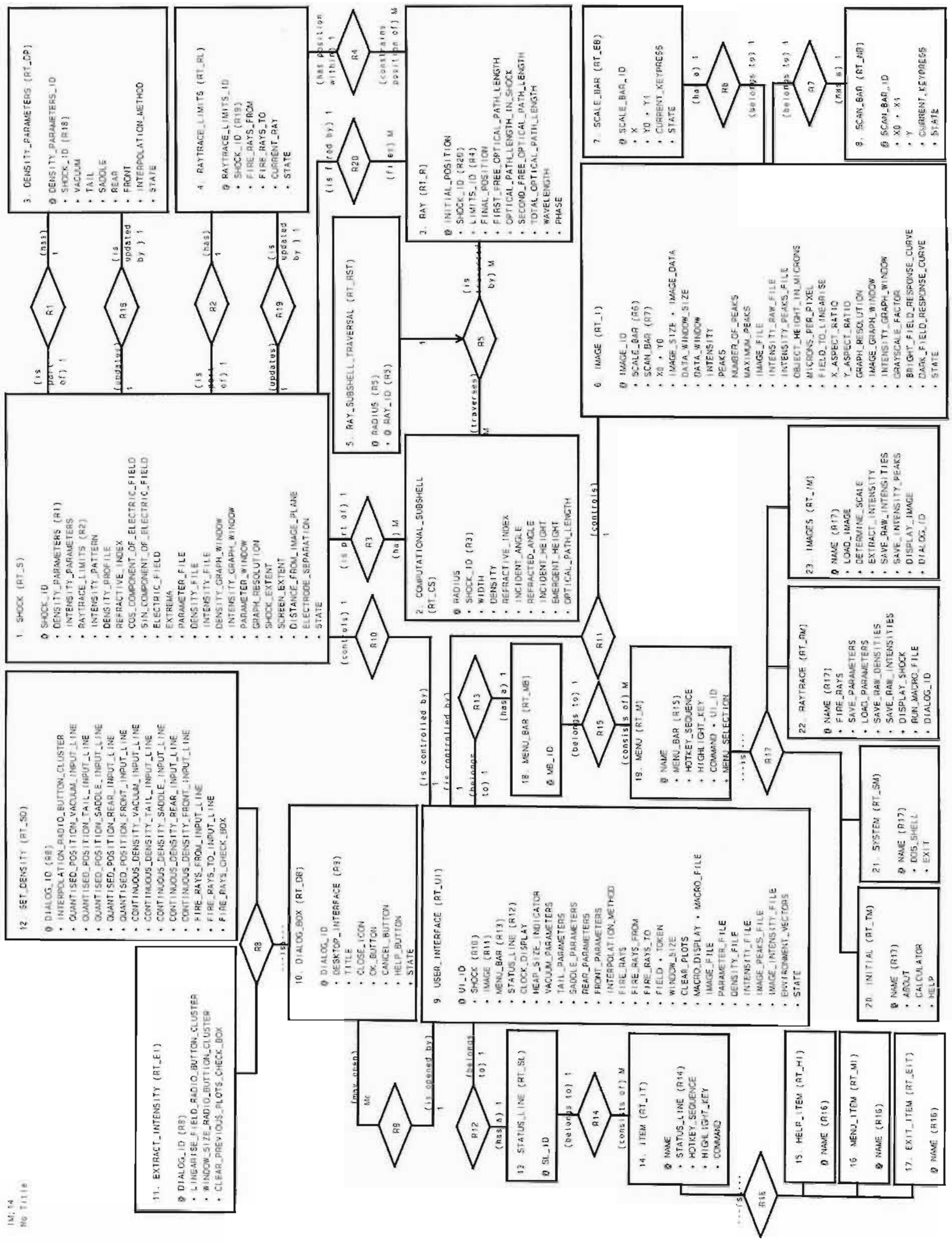


Figure D.1 Entity Relationship Diagram

D.9 Data Dictionary

1. SHOCK

Attributes:
SHOCK_ID
DENSITY_PARAMETERS
INTENSITY_PARAMETERS
RAYTRACE_LIMITS
INTENSITY_PATTERN
DENSITY_PROFILE
REFRACTIVE_INDEX
COS_COMPONENT_OF_ELECTRIC_FIELD
SIN_COMPONENT_OF_ELECTRIC_FIELD
ELECTRIC_FIELD
EXTREMA
PARAMETER_FILE
DENSITY_FILE
INTENSITY_FILE
DENSITY_GRAPH_WINDOW
INTENSITY_GRAPH_WINDOW
PARAMETER_WINDOW
GRAPH_RESOLUTION
SHOCK_EXTENT
SCREEN_EXTENT
DISTANCE_FROM_IMAGE_PLANE
ELECTRODE_SEPARATION
STATE

Primary Identifiers: SHOCK_ID

Description: This object models various physical properties of a spherical shock in air, its graphical representation and the theoretical interference pattern produced by the raytracing engine.

Key Letter: RT_S

1.1 SHOCK.SHOCK_ID

Description: The identity of a particular instance of a shock. In practice there shall be only one, so this requirement will fall away in the coding phase.

Domain: Natural.

1.2 SHOCK.DENSITY_PARAMETERS

Description: The identity of the object storing the shock's density parameters.

Relationship: R1:DENSITY_PARAMETERS_ID

Domain: Same as DENSITY_PARAMETERS.DENSITY_PARAMETERS_ID.

1.3 SHOCK.INTENSITY_PARAMETERS

Description: These are the peaks and troughs of the intensity pattern.

Domain: An array of points storing quantised position in microns and continuous absolute calculated intensity to double (64 bit) precision.

1.4 SHOCK.RAYTRACE_LIMITS

Description: The identity of the object storing the shock's raytrace limits - i.e. the positions into the shock at which to start and stop firing rays.

Relationship: R2:RAYTRACE_LIMITS_ID

Domain: Same as RAYTRACE_LIMITS.RAYTRACE_LIMITS_ID.

1.5 SHOCK.INTENSITY_PATTERN

Description: This is the calculated intensity for each quantised position on the screen.

Domain: An array of points storing continuous absolute calculated intensity to double (64 bit) precision.

1.6 SHOCK.DENSITY_PROFILE
Description: The postulated density profile underlying the shock.
Domain: An array of points storing continuous density in atmospheres to double (64 bit) precision.

1.7 SHOCK.REFRACTIVE_INDEX
Description: The refractive index of the shocked air at each quantised position within the shock.
Domain: An array of points storing continuous absolute refractive index to double (64 bit) precision.

1.8 SHOCK.COS_COMPONENT_OF_ELECTRIC_FIELD
Description: The cos component of the electric field used to calculate the intensity pattern.
Domain: An array of real values stored to double (64 bit) precision.

1.9 SHOCK.SIN_COMPONENT_OF_ELECTRIC_FIELD
Description: The sin component of the electric field used to calculate the intensity pattern.
Domain: An array of real values stored to double (64 bit) precision.

1.10 SHOCK.ELECTRIC_FIELD
Description: The electric field of the intensity pattern formed by the combination of its sin and cos components.
Domain: An array of real values stored to double (64 bit) precision.

1.11 SHOCK.EXTREMA
Description: The number of extrema (peaks and troughs) in the calculated intensity pattern.
Domain: Natural, 0..100.

1.12 SHOCK.PARAMETER_FILE
Description: The name of a file storing shock parameter details (the density and intensity parameters). The name of the file may be used interchangeably with the file itself.
Domain: ASCII pathname string.

1.13 SHOCK.DENSITY_FILE
Description: The name of a file storing the entire shock density profile. The name of the file may be used interchangeably with the file itself.
Domain: ASCII pathname string.

1.14 SHOCK.INTENSITY_FILE
Description: The name of a file storing the entire shock intensity pattern. The name of the file may be used interchangeably with the file itself.
Domain: ASCII pathname string.

1.15 SHOCK.DENSITY_GRAPH_WINDOW
Description: The identity of the graphics window in which a graph of the density profile will be displayed.
Domain: Natural.

1.16 SHOCK.INTENSITY_GRAPH_WINDOW
Description: The identity of the graphics window in which a graph of the intensity pattern will be displayed.
Domain: Natural.

1.17 SHOCK.PARAMETER_WINDOW
Description: The identity of the graphics window in which the user-selected density parameters, raytrace limits and progress indicator will be displayed.
Domain: Natural.

1.18 SHOCK.GRAPH_RESOLUTION

Description: The resolution at which to display graphs in the graphics windows. Every GRAPH_RESOLUTION'th data point will be plotted.
Domain: Natural.

1.19 SHOCK.SHOCK_EXTENT
Description: The maximum diameter of the spherical shock.
Domain: Natural (quantised into microns).

1.20 SHOCK.SCREEN_EXTENT
Description: The maximum extent of the calculated intensity pattern that will be considered.
Domain: Natural (quantised into microns).

1.21 SHOCK.DISTANCE_FROM_IMAGE_PLANE
Description: The distance from the centre of the shock to the image plane. This information is used in calculating the intensity pattern.
Domain: Natural (quantised into microns).

1.22 SHOCK.ELECTRODE_SEPARATION
Description: The distance separating the electrode pins. A spark discharge across this gap is what gives rise to the spherical shock. This information is used in scaling the image.
Domain: Natural (quantised into microns).

1.23 SHOCK.STATE
Description: The current state of the object when considered as a state machine.
Domain: See State Transition Diagram.

2. COMPUTATIONAL_SUBSHELL

Attributes: RADIUS
SHOCK_ID
WIDTH
DENSITY
REFRACTIVE_INDEX
INCIDENT_ANGLE
REFRACTED_ANGLE
INCIDENT_HEIGHT
EMERGENT_HEIGHT
OPTICAL_PATH_LENGTH
Primary Identifiers: RADIUS
Description: For computational purposes, the shock is divided into numerous subshells, each one micron apart. This object models one such subshell and the way it affects a single ray traversing it.
Key Letter: RT_CS

2.1 COMPUTATIONAL_SUBSHELL.RADIUS
Description: The outer radius of the computational subshell. This attribute uniquely identifies the instance of the computational subshell.
Domain: Natural (microns).

2.2 COMPUTATIONAL_SUBSHELL.SHOCK_ID
Description: The identity of the shock to which the computational subshell belongs.
Relationship: R3:SHOCK
Domain: same as SHOCK.SHOCK_ID

2.3 COMPUTATIONAL_SUBSHELL.WIDTH
Description: The width of the computational subshell - set to one micron.

Domain: Natural (microns).

2.4 COMPUTATIONAL_SUBSHELL.DENSITY

Description: The density of the air within the computational subshell.
Domain: Real value stored to double (64 bit) precision in units of atmospheres.

2.5 COMPUTATIONAL_SUBSHELL.REFRACTIVE_INDEX

Description: The refractive index of the air within the computational subshell - this depends empirically on the density of the air.
Domain: Real value stored to double (64 bit) precision

2.6 COMPUTATIONAL_SUBSHELL.INCIDENT_ANGLE

Description: The angle at which an incident ray enters the subshell, measured with respect to the normal to the subshell surface in the same plane as the normal and incident ray.
Domain: Real value stored to double (64 bit) precision

2.7 COMPUTATIONAL_SUBSHELL.REFRACTED_ANGLE

Description: The angle at which the incident ray leaves the boundary defined by the subshell's outer radius once it has been refracted at that boundary, measured with respect to the normal to the subshell surface in the same plane as the normal and incident ray.
Domain: Real value stored to double (64 bit) precision

2.8 COMPUTATIONAL_SUBSHELL.INCIDENT_HEIGHT

Description: The perpendicular height above a horizontal line through the centre of the shock at which the incident ray enters the subshell.
Domain: Natural (quantised into microns).

2.9 COMPUTATIONAL_SUBSHELL.EMERGENT_HEIGHT

Description: The perpendicular height above a horizontal line through the centre of the shock at which the emergent ray leaves the subshell. This is also the incident height of the ray into the next subshell.
Domain: Natural (quantised into microns).

2.10 COMPUTATIONAL_SUBSHELL.OPTICAL_PATH_LENGTH

Description: The optical distance travelled by the ray within the computational subshell, calculated from the previous attributes.
Domain: Real value stored to double (64 bit) precision.

3. DENSITY_PARAMETERS

Attributes: DENSITY_PARAMETERS_ID
SHOCK_ID
VACUUM
TAIL
SADDLE
REAR
FRONT
INTERPOLATION_METHOD
STATE

Primary Identifiers: DENSITY_PARAMETERS_ID
Description: This object represents the parameters of the density profile underlying the shock.
Key Letter: RT_DP

3.1 DENSITY_PARAMETERS.DENSITY_PARAMETERS_ID

Description: The identity of a particular instance of the density parameters object. In practice there shall be only one, so this requirement will fall away in the coding phase.
Domain: Natural.

3.2 DENSITY_PARAMETERS.SHOCK_ID

Description: The identity of the shock whose density parameters this object holds.
Relationship: R18:SHOCK
Domain: Same as SHOCK.SHOCK_ID.

3.3 DENSITY_PARAMETERS.VACUUM

Description: The position and density of the end of the shock vacuum region (moving outwards).
Domain: Natural quantised into microns, Real value double (64 bit) precision in atmospheres.

3.4 DENSITY_PARAMETERS.TAIL

Description: The position and density of the end of the shock tail region (moving outwards).
Domain: Natural quantised into microns, Real value double (64 bit) precision in atmospheres.

3.5 DENSITY_PARAMETERS.SADDLE

Description: The position and density of the end of the shock saddle region (moving outwards).
Domain: Natural quantised into microns, Real value double (64 bit) precision in atmospheres.

3.6 DENSITY_PARAMETERS.REAR

Description: The position and density of the end of the shock rear region (moving outwards).
Domain: Natural quantised into microns, Real value double (64 bit) precision in atmospheres.

3.7 DENSITY_PARAMETERS.FRONT

Description: The position and density of the end of the shock front region (moving outwards).
Domain: Natural quantised into microns, Real value double (64 bit) precision in atmospheres.

3.8 DENSITY_PARAMETERS.INTERPOLATION_METHOD

Description: The method to be used to reconstruct the full density profile from the density parameters.
Domain: [Linear ¦ Cubic Spline ¦ Lagrange ¦ Junction].

3.9 DENSITY_PARAMETERS.STATE

Description: The current state of the object when considered as a state machine.
Domain: See State Transition Diagram.

4. RAY

Attributes:
INITIAL_POSITION
SHOCK_ID
LIMITS_ID
FINAL_POSITION
FIRST_FREE_OPTICAL_PATH_LENGTH
OPTICAL_PATH_LENGTH_IN_SHOCK
SECOND_FREE_OPTICAL_PATH_LENGTH
TOTAL_OPTICAL_PATH_LENGTH
WAVELENGTH
PHASE

Primary Identifiers: INITIAL_POSITION
Description: Numerous rays of laser light, bounded by the raytrace limits and each one micron apart traverse the entire shock and terminate at the image plane. This object models one such ray.
Key Letter: RT_R

4.1 RAY.INITIAL_POSITION

Description: The height, above a horizontal line through the centre of the shock, from which the ray is fired. This attribute is sufficient to uniquely identify the ray.
Domain: 0 .. SHOCK_EXTENT quantised into microns.

4.2 RAY.SHOCK_ID

Description: The identity of the shock which the ray will traverse.

Relationship: R20:SHOCK

Domain: Same as SHOCK.SHOCK_ID.

4.3 RAY.LIMITS_ID

Description: The identity of the raytrace limits constraining the position within which the rays will be fired.

Relationship: R19:RAYTRACE_LIMITS

Domain: Same as RAYTRACE_LIMITS.RAYTRACE_LIMITS_ID

4.4 RAY.FINAL_POSITION

Description: The height, above a horizontal line through the centre of the shock, at which the ray intersects the screen after traversing the shock.

Domain: 0 .. SCREEN_EXTENT quantised into microns.

4.5 RAY.FIRST_FREE_OPTICAL_PATH_LENGTH

Description: The optical distance travelled by the ray, from a plane tangential to the shock front and perpendicular to the horizontal line through the centre of the shock, to when it enters the shock at the shock front.

Domain: Natural in units of microns.

4.6 RAY.OPTICAL_PATH_LENGTH_IN_SHOCK

Description: The optical distance travelled by the ray within the shock, through each of its computational subshells. Most of the ray-tracer's computational effort is expended in computing this attribute.

Domain: Natural in units of microns.

4.7 RAY.SECOND_FREE_OPTICAL_PATH_LENGTH

Description: The optical distance travelled by the ray, from the point at which it exits the shock to the image plane or screen.

Domain: Natural in units of microns.

4.8 RAY.TOTAL_OPTICAL_PATH_LENGTH

Description: Sum of FIRST_FREE_OPTICAL_PATH_LENGTH & OPTICAL_PATH_LENGTH_IN_SHOCK & SECOND_FREE_OPTICAL_PATH_LENGTH.

Domain: Natural in units of microns.

4.9 RAY.WAVELENGTH

Description: The wavelength of the ray of light.

Domain: Natural in units of microns.

4.10 RAY.PHASE

Description: The phase of the ray of light when it arrives at the image plane.

Domain: Real stored to double (64 bit) precision.

5. RAYTRACE_LIMITS

Attributes: RAYTRACE_LIMITS_ID
SHOCK_ID
FIRE_RAYS_FROM
FIRE_RAYS_TO
CURRENT_RAY
STATE

Primary Identifiers: RAYTRACE_LIMITS_ID

Description: This object storing the limits bounding the raytrace of the shock.

Key Letter: RT_RL

5.1 RAYTRACE_LIMITS.RAYTRACE_LIMITS_ID

Description: The identity of a particular instance of the raytrace limits object. In practice there shall be only one so this requirement will fall away in the coding phase.
Domain: Natural.

5.2 RAYTRACE_LIMITS.SHOCK_ID
Description: The identity of the shock which will be raytraced.
Relationship: R19:SHOCK
Domain: Same as SHOCK.SHOCK_ID.

5.3 RAYTRACE_LIMITS.FIRE_RAYS_FROM
Description: The height, above a horizontal line through the centre of the shock, at which to start firing rays. This defaults to the first position.
Domain: 0 .. SHOCKEXTENT quantised into microns.

5.4 RAYTRACE_LIMITS.FIRE_RAYS_TO
Description: The height, above a horizontal line through the centre of the shock, at which to stop firing rays. This defaults to the last position.
Domain: 0 .. SHOCKEXTENT quantised into microns and greater than FIRE_RAYS_FROM.

5.5 RAYTRACE_LIMITS.CURRENT_RAY
Description: The identity of the ray currently being fired. This is its height as previously defined.
Domain: 0 .. SHOCKEXTENT quantised into microns and within the limits.

5.6 RAYTRACE_LIMITS.STATE
Description: The current state of the object when considered as a state machine.
Domain: See State Transition Diagram.

6. RAY_SUBSHELL_TRAVERSAL

Attributes: RADIUS
RAY_ID
Primary Identifiers: RADIUS, RAY_ID
Description: The association of a particular ray with a particular computational subshell is formalised by this object.
Key Letter: RT_RST

6.1 RAY_SUBSHELL_TRAVERSAL.RADIUS
Description: The identity of the computational subshell being traversed.
Relationship: R5:COMPUTATIONAL_SUBSHELL
Domain: Same as COMPUTATIONAL_SUBSHELL.RADIUS

6.2 RAY_SUBSHELL_TRAVERSAL.RAY_ID
Description: The identity of the traversing ray.
Relationship: R5:RAY
Domain: Same as RAY.INITIAL_POSITION

7. IMAGE

Attributes: IMAGE_ID
SCALE_BAR
SCAN_BAR
X0
Y0
IMAGE_SIZE
IMAGE_DATA
DATA_WINDOW_SIZE

DATA_WINDOW
INTENSITY
PEAKS
NUMBER_OF_PEAKS
MAXIMUM_PEAKS
IMAGE_FILE
INTENSITY_RAW_FILE
INTENSITY_PEAKS_FILE
OBJECT_HEIGHT_IN_MICRONS
MICRONS_PER_PIXEL
FIELD_TO_LINEARISE
X_ASPECT_RATIO
Y_ASPECT_RATIO
GRAPH_RESOLUTION
IMAGE_GRAPH_WINDOW
INTENSITY_GRAPH_WINDOW
GRAYSCALE_FACTOR
BRIGHT_FIELD_RESPONSE_CURVE
DARK_FIELD_RESPONSE_CURVE
STATE

Primary Identifiers: IMAGE_ID
Description: Physical laser rays traversing the shock produce an experimental interference pattern at the image plane. This pattern is captured on camera and modelled by this object. This experimental image is distinct from the theoretical pattern object.
Key Letter: RT_I

7.1 IMAGE.IMAGE_ID
Description: The identity of a specific instance of the image object. In practice there shall be only one, so this requirement will fall away in the coding phase.
Domain: Natural.

7.2 IMAGE.SCALE_BAR
Description: The identity of the scale bar attached to the image.
Relationship: R6:SCALE_BAR
Domain: Same as SCALE_BAR.SCALE_BAR_ID.

7.3 IMAGE.SCAN_BAR
Description: The identity of the scan bar attached to the image.
Relationship: R7:SCAN_BAR
Domain: Same as SCAN_BAR.SCAN_BAR_ID.

7.4 IMAGE.X0
Description: The screen row at which to position the image for display.
Domain: Natural.

7.5 IMAGE.Y0
Description: The screen column at which to position the image for display.
Domain: Natural.

7.6 IMAGE.IMAGE_SIZE
Description: The size of the image (it will always be square).
Domain: 0..512 pixels.

7.7 IMAGE.IMAGE_DATA
Description: The rows and columns of pixel values in the entire image.

Domain: Image Size x Image Size array of grayscale pixel values in range 0 ..127.

7.8 IMAGE.DATA_WINDOW_SIZE

Description: The number of rows, on either side of the scan bar, from which image intensities are extracted for further processing.

Domain: 0..2.

7.9 IMAGE.DATA_WINDOW

Description: The rows and columns of pixel values in the image area under the data window.

Domain: Data Window Size x Image Size array of grayscale pixel values in range 0 ..127.

7.10 IMAGE.INTENSITY

Description: The intensity values extracted from the data window. The values of the different rows are averaged. The values may be raw pixel values or actual absolute intensities if the data window has been linearised using a response curve.

Domain: 1 x image size array of real values stored to double (64 bit) precision with range 0.0 .. 127.0.

7.11 IMAGE.PEAKS

Description: The peaks and troughs (extrema) present in the extracted intensity pattern.

Domain: Array of points storing the position (natural) and intensity (real) of the extrema.

7.12 IMAGE.NUMBER_OF_PEAKS

Description: The number of peaks and troughs (extrema) present in the extracted intensity pattern.

Domain: Natural.

7.13 IMAGE.MAXIMUM_PEAKS

Description: The maximum number of peaks that will be considered, starting from the position in the image that marks the centre of the shock and moving to the right.

Domain: Natural (60).

7.14 IMAGE.IMAGE_FILE

Description: The name of the file storing the image. This attribute is used interchangeably to refer to the physical file itself.

Domain: ASCII pathname.

7.15 IMAGE.INTENSITY_RAW_FILE

Description: The name of the file storing the raw extracted intensities (whether linearised or not). This attribute is used interchangeably to refer to the physical file itself.

Domain: ASCII pathname.

7.16 IMAGE.INTENSITY_PEAKS_FILE

Description: The name of the file storing the extracted intensity peaks (whether linearised or not). This attribute is used interchangeably to refer to the physical file itself.

Domain: ASCII pathname.

7.17 IMAGE.OBJECT_HEIGHT_IN_MICRONS

Description: The height in microns of the vertical object that the scale bar is scaled to match. This exercise enables the determination of a micron/pixel scale for the image.

Domain: Natural quantised into microns.

7.18 IMAGE.MICRONS_PER_PIXEL

Description: The micron/pixel scale determined by using the scale bar.

Domain: Real stored to double (64 bit) precision.

7.19 IMAGE.FIELD_TO_LINEARISE

Description: The image consists of two interleaved fields, referred to as the bright and dark fields. This attribute determined which of the two is to be selected for linearisation and/or extraction within the data window.
Domain: [BRIGHT ¦ DARK ¦ NONE].

7.20 IMAGE.X_ASPECT_RATIO
Description: Specifies the X Aspect ratio with which to display the image. Every X Aspect Ratio'ed column will be displayed.
Domain: Natural range 1..Image Size.

7.21 IMAGE.Y_ASPECT_RATIO
Description: Specifies the Y Aspect ratio with which to display the image. Every Y Aspect Ratio'ed row will be displayed.
Domain: Natural range 1..Image Size.

7.22 IMAGE.GRAPH_RESOLUTION
Description: The resolution at which to display a graph of the extracted intensities (linearised or not). Every Graph Resolution'th point will be plotted.
Domain: Natural.

7.23 IMAGE.IMAGE_GRAPH_WINDOW
Description: The identity of the graphics window in which the image will be displayed.
Domain: Natural.

7.24 IMAGE.INTENSITY_GRAPH_WINDOW
Description: The identity of the graphics window in which a graph of the extracted intensities will be displayed.
Domain: Natural.

7.25 IMAGE.GRAYSCALE_FACTOR
Description: The number of grayscales that will be used to display the image.
Domain: Natural (16, 64, 128).

7.26 IMAGE.BRIGHT_FIELD_RESPONSE_CURVE
Description: An empirically determined logarithmic curve used to convert pixels values in the bright field of the image to actual absolute intensities. This is a compensation for the logarithmic response of the camera capturing the image. The curve is interpolated between the points defining it.
Domain: Array of mapping co-ordinates: pixel (natural) - intensity (real).

7.27 IMAGE.DARK_FIELD_RESPONSE_CURVE
Description: An empirically determined logarithmic curve used to convert pixels values in the dark field of the image to actual absolute intensities. This is a compensation for the logarithmic response of the camera capturing the image. The curve is interpolated between the points defining it.
Domain: Array of mapping co-ordinates: pixel (natural) - intensity (real).

7.28 IMAGE.STATE
Description: The current state of the object when considered as a state machine.
Domain: See State Transition Diagram.

8. SCALE_BAR

Attributes: SCALE_BAR_ID
X
Y0
Y1
CURRENT_KEYPRESS
STATE
Primary Identifiers: SCALE_BAR_ID
Description: The scale bar is used to scale the image by measuring the pixel length of a feature of known dimensions.

Key Letter: RT_EB

8.1 SCALE_BAR.SCALE_BAR_ID
Description: The identity of a specific instance of the scale bar object. In practice there shall be only one, so this requirement will fall away in the coding phase.
Domain: Natural.

8.2 SCALE_BAR.X
Description: The column at which the scale bar is positioned.
Domain: 1..Image Size.

8.3 SCALE_BAR.Y0
Description: The row at which the scale bar starts.
Domain: 1..Image Size.

8.4 SCALE_BAR.Y1
Description: The row at which the scale bar ends.
Domain: 1..Image Size.

8.5 SCALE_BAR.CURRENT_KEYPRESS
Description: The current key pressed by the operator.
Domain: Character (including control characters).

8.6 SCALE_BAR.STATE
Description: The current state of the object when considered as a state machine.
Domain: See State Transition Diagram.

9. SCAN_BAR

Attributes: SCAN_BAR_ID
X0
X1
Y
CURRENT_KEYPRESS
STATE
Primary Identifiers: SCAN_BAR_ID
Description: The scan bar is used to select the central row of a data window from which image pixel intensities are to be extracted and linearised.
Key Letter: RT_NB

9.1 SCAN_BAR.SCAN_BAR_ID
Description: The identity of a specific instance of the scan bar object. In practice there shall be only one, so this requirement will fall away in the coding phase.
Domain: Natural.

9.2 SCAN_BAR.X0
Description: The column at which the scan bar starts.
Domain: 1..Image Size.

9.3 SCAN_BAR.X1
Description: The column at which the scan bar ends.
Domain: 1..Image Size.

9.4 SCAN_BAR.Y
Description: The row at which the scan bar is positioned.

Domain: 1..Image Size.

9.5 SCAN_BAR.CURRENT_KEYPRESS
Description: The current key pressed by the operator.
Domain: Character (including control characters).

9.6 SCAN_BAR.STATE
Description: The current state of the object when considered as a state machine.
Domain: See State Transition Diagram.

10. USER_INTERFACE

Attributes:
UI_ID
SHOCK
IMAGE
MENU_BAR
STATUS_LINE
CLOCK_DISPLAY
HEAP_SIZE_INDICATOR
VACUUM_PARAMETERS
TAIL_PARAMETERS
SADDLE_PARAMETERS
REAR_PARAMETERS
FRONT_PARAMETERS
INTERPOLATION_METHOD
FIRE_RAYS
FIRE_RAYS_FROM
FIRE_RAYS_TO
FIELD
TOKEN
WINDOW_SIZE
CLEAR_PLOTS
MACRO_DISPLAY
MACRO_FILE
IMAGE_FILE
PARAMETER_FILE
DENSITY_FILE
INTENSITY_FILE
IMAGE_PEAKS_FILE
IMAGE_INTENSITY_FILE
ENVIRONMENT_VECTORS
STATE

Primary Identifiers: UI_ID
Description: This represents the mouse-driven, menued and dialog-boxed user interface. This object ties all the others together.
Key Letter: RT_UI

10.1 USER_INTERFACE.UI_ID
Description: The identity of a specific instance of the user interface object. In practice there shall be only one, so this requirement will fall away in the coding phase.
Domain: Natural.

10.2 USER_INTERFACE.SHOCK
Description: The identity of the shock which the user interface controls.
Relationship: R10:SHOCK

Domain: Same as SHOCK.SHOCK_ID.

10.3 USER_INTERFACE.IMAGE
Description: The identity of the image which the user interface controls.
Relationship: R11:IMAGE
Domain: Same as IMAGE.IMAGE_ID.

10.4 USER_INTERFACE.MENU_BAR
Description: The identity of the menu bar at the top of the user interface desktop.
Relationship: R13:MENU_BAR
Domain: Same as MENU_BAR.MB_ID.

10.5 USER_INTERFACE.STATUS_LINE
Description: The identity of the status line at the bottom of the user interface desktop.
Relationship: R12:STATUS_LINE
Domain: Same as STATUS_LINE.SL_ID.

10.6 USER_INTERFACE.CLOCK_DISPLAY
Description: A readout of the current time to the right of the menu bar.
Domain: ASCII text.

10.7 USER_INTERFACE.HEAP_SIZE_INDICATOR
Description: A readout of the free heap space to the right of the status line.
Domain: ASCII text.

10.8 USER_INTERFACE.VACUUM_PARAMETERS
Description: Vacuum parameters of the shock density received from, or displayed to, the operator.
Domain: Same as DENSITY_PARAMETERS.VACUUM.

10.9 USER_INTERFACE.TAIL_PARAMETERS
Description: Tail parameters of the shock density received from, or displayed to, the operator.
Domain: Same as DENSITY_PARAMETERS.TAIL.

10.10 USER_INTERFACE.SADDLE_PARAMETERS
Description: Saddle parameters of the shock density received from, or displayed to, the operator.
Domain: Same as DENSITY_PARAMETERS.SADDLE.

10.11 USER_INTERFACE.REAR_PARAMETERS
Description: Rear parameters of the shock density received from, or displayed to, the operator.
Domain: Same as DENSITY_PARAMETERS.REAR.

10.12 USER_INTERFACE.FRONT_PARAMETERS
Description: Front parameters of the shock density received from, or displayed to, the operator.
Domain: Same as DENSITY_PARAMETERS.FRONT.

10.13 USER_INTERFACE.INTERPOLATION_METHOD
Description: Shock parameter interpolation method received from, or displayed to, the operator.
Domain: Same as DENSITY_PARAMETERS.INTERPOLATION_METHOD.

10.14 USER_INTERFACE.FIRE_RAYS
Description: Indication of the operator's intent to commence ray tracing within the specified limits.
Domain: Boolean.

10.15 USER_INTERFACE.FIRE_RAYS_FROM
Description: Fire Rays From parameter of Raytrace Limits received from, or displayed to, the operator.

Domain: Same as RAYTRACE_LIMITS.FIRE_RAYS_FROM.

10.16 USER_INTERFACE.FIRE_RAYS_TO

Description: Fire Rays To parameter of Raytrace Limits received from, or displayed to, the operator.

Domain: Same as RAYTRACE_LIMITS.FIRE_RAYS_FROM.

10.17 USER_INTERFACE.FIELD

Description: Field To Linearise parameter of the Image received from, or displayed to, the operator.

Domain: Same as IMAGE.FIELD_TO_LINEARISE.

10.18 USER_INTERFACE.TOKEN

Description: A token parsed from a User Interface Macro File.

Domain: [NEW_KEYWORD ¦ VACUUM_KEYWORD ¦ TAIL_KEYWORD ¦ SADDLE_KEYWORD ¦ REAR_KEYWORD ¦ FRONT_KEYWORD ¦ FIRE_RAYS_KEYWORD ¦ INTERPOLATE_KEYWORD ¦ DISPLAY_KEYWORD ¦ PARAMETER_FILE_KEYWORD ¦DENSITY_FILE_KEYWORD ¦INTENSITY_FILE_KEYWORD ¦END_KEYWORD ¦ REAL_VALUE ¦ INTEGER_VALUE ¦ STRING_VALUE]

10.19 USER_INTERFACE.WINDOW_SIZE

Description: The Data Window Size parameter of the Image received from, or displayed to, the operator.

Domain: Same as IMAGE.DATA_WINDOW_SIZE.

10.20 USER_INTERFACE.CLEAR_PLOTS

Description: Indication of the operator's intent to clear the Image Graph Window of the Image object each time the Image Intensity is extracted via the Scan Bar.

Domain: Boolean.

10.21 USER_INTERFACE.MACRO_DISPLAY

Description: Instruction to display the Shock Density and Intensity Graphs and Parameter Window during processing of a User Interface Macro File. The instruction is set upon detection of a DISPLAY_KEYWORD within the Macro File being processed.

Domain: Boolean.

10.22 USER_INTERFACE.MACRO_FILE

Description: The name of the Macro File storing instructions for non-interactive, batch processing by the ray-tracer. This attribute is used interchangeably to refer to the physical file itself.

Domain: ASCII pathname.

10.23 USER_INTERFACE.IMAGE_FILE

Description: The name of the file containing the image from which to extract intensities. This attribute is used interchangeably to refer to the physical file itself.

Domain: ASCII pathname.

10.24 USER_INTERFACE.PARAMETER_FILE

Description: The name of the file storing the shock's theoretical density and intensity parameters - this forms the basis of the training data input to the neural network. This attribute is used interchangeably to refer to the physical file itself.

Domain: ASCII pathname.

10.25 USER_INTERFACE.DENSITY_FILE

Description: The name of the file storing the raw (unparametrised) density profile. This attribute is used interchangeably to refer to the physical file itself.

Domain: ASCII pathname.

10.26 USER_INTERFACE.INTENSITY_FILE

Description: The name of the file storing the raw (unparametrised) theoretical intensity pattern. This attribute is used interchangeably to refer to the physical file itself.

Domain: ASCII pathname.

10.27 USER_INTERFACE.IMAGE_PEAKS_FILE

Description: The name of the file storing the peaks and troughs (parametrised data) of the extracted experimental intensity pattern. This attribute is used interchangeably to refer to the physical file itself.

Domain: ASCII pathname.

10.28 USER_INTERFACE.IMAGE_INTENSITY_FILE

Description: The name of the file storing the raw (unparametrised) experimental intensity profile extracted from an image. This attribute is used interchangeably to refer to the physical file itself.

Domain: ASCII pathname.

10.29 USER_INTERFACE.ENVIRONMENT_VECTORS

Description: A representation of the operating system environment parameters that need to saved and restored when opening a shell to the operating system and when returning to the main application.

Domain: Operating system dependent.

10.30 USER_INTERFACE.STATE

Description: The current state of the object when considered as a state machine.

Domain: See State Transition Diagram.

11. DIALOG_BOX

Attributes: DIALOG_ID
DESKTOP_INTERFACE
TITLE
CLOSE_ICON
OK_BUTTON
CANCEL_BUTTON
HELP_BUTTON
STATE

Primary Identifiers: DIALOG_ID

Description: This is a non-functional ancestor dialog box from which successors may inherit.

Key Letter: RT_DB

11.1 DIALOG_BOX.DIALOG_ID

Description: This is the identity of a particular instance of the dialog box object. The representation of this attribute, but not its nature, may change during the coding phase.

Domain: Natural.

11.2 DIALOG_BOX.DESKTOP_INTERFACE

Description: The identity of the user interface which owns the dialog box.

Relationship: R9:USER_INTERFACE

Domain: Same as USER_INTERFACE.UI_ID.

11.3 DIALOG_BOX.TITLE

Description: The displayed title of the dialog box.

Domain: ASCII text.

11.4 DIALOG_BOX.CLOSE_ICON

Description: The status of a displayed symbol which will cause the dialog box to be closed when it is selected.

Domain: Boolean.

11.5 DIALOG_BOX.OK_BUTTON

Description: The status of a labelled button in the dialog box which, when depressed, will execute the actions of the dialog box.

Domain: Boolean.

11.6 DIALOG_BOX.CANCEL_BUTTON

Description: The status of a labelled button in the dialog box which, when depressed, will dismiss the dialog box without taking any further action.

Domain: Boolean.

11.7 DIALOG_BOX.HELP_BUTTON

Description: The status of a labelled button in the dialog box which, when depressed, will open a help window explaining the functionality of the dialog box.

Domain: Boolean.

11.8 DIALOG_BOX.STATE

Description: The current state of the object when considered as a state machine.

Domain: See State Transition Diagram of descendants.

12. EXTRACT_INTENSITY

Attributes: DIALOG_ID
LINEARISE_FIELD_RADIO_BUTTON_CLUSTER
WINDOW_SIZE_RADIO_BUTTION_CLUSTER
CLEAR_PREVIOUS_PLOTS_CHECK_BOX

Primary Identifiers: DIALOG_ID

Description: This dialog box is used to get information from the operator prior to extracting image intensities.

Key Letter: RT_EI

12.1 EXTRACT_INTENSITY.DIALOG_ID

Description: The identity of the dialog box.

Relationship: R8:DIALOG_BOX

Domain: Same as DIALOG_BOX.DIALOG_ID.

12.2 EXTRACT_INTENSITY.LINEARISE_FIELD_RADIO_BUTTON_CLUSTER

Description: The value of a selection by the operator of various Image Field To Linearise options displayed in the dialog box.

Domain: Natural value with bit encoding.

12.3 EXTRACT_INTENSITY.WINDOW_SIZE_RADIO_BUTTION_CLUSTER

Description: The value of a selection by the operator of various Image Data Window Size options displayed in the dialog box.

Domain: Natural value with bit encoding.

12.4 EXTRACT_INTENSITY.CLEAR_PREVIOUS_PLOTS_CHECK_BOX

Description: The value of a check by the operator setting the User Interface Clear Plots attribute displayed in the dialog box.

Domain: Boolean.

13. SET_DENSITY

Attributes: DIALOG_ID
INTERPOLATION_RADIO_BUTTON_CLUSTER
QUANTISED_POSITION_VACUUM_INPUT_LINE
QUANTISED_POSITION_TAIL_INPUT_LINE
QUANTISED_POSITION_SADDLE_INPUT_LINE
QUANTISED_POSITION_REAR_INPUT_LINE
QUANTISED_POSITION_FRONT_INPUT_LINE
CONTINUOUS_DENSITY_VACUUM_INPUT_LINE
CONTINUOUS_DENSITY_TAIL_INPUT_LINE
CONTINUOUS_DENSITY_SADDLE_INPUT_LINE

CONTINUOUS_DENSITY_REAR_INPUT_LINE
CONTINUOUS_DENSITY_FRONT_INPUT_LINE
FIRE_RAYS_FROM_INPUT_LINE
FIRE_RAYS_TO_INPUT_LINE
FIRE_RAYS_CHECK_BOX

Primary Identifiers: DIALOG_ID
Description: This dialog box is used to get information from the operator prior to setting a density profile and raytracing.
Key Letter: RT_SD

13.1 SET_DENSITY.DIALOG_ID
Description: The identity of the dialog box.
Relationship: R8:DIALOG_BOX
Domain: Same as DIALOG_BOX.DIALOG_ID.

13.2 SET_DENSITY.INTERPOLATION_RADIO_BUTTON_CLUSTER
Description: The value of a selection by the operator of an Image Interpolation Method option.
Domain: Natural value with bit encoding.

13.3 SET_DENSITY.QUANTISED_POSITION_VACUUM_INPUT_LINE
Description: A displayed field in the dialog box containing the position component (in microns) of the Vacuum Shock parameter as input by the operator.
Domain: ASCII string representation of a natural number.

13.4 SET_DENSITY.QUANTISED_POSITION_TAIL_INPUT_LINE
Description: A displayed field in the dialog box containing the position component (in microns) of the Tail Shock parameter as input by the operator.
Domain: ASCII string representation of a natural number.

13.5 SET_DENSITY.QUANTISED_POSITION_SADDLE_INPUT_LINE
Description: A displayed field in the dialog box containing the position component (in microns) of the Saddle Shock parameter as input by the operator.
Domain: ASCII string representation of a natural number.

13.6 SET_DENSITY.QUANTISED_POSITION_REAR_INPUT_LINE
Description: A displayed field in the dialog box containing the position component (in microns) of the Rear Shock parameter as input by the operator.
Domain: ASCII string representation of a natural number.

13.7 SET_DENSITY.QUANTISED_POSITION_FRONT_INPUT_LINE
Description: A displayed field in the dialog box containing the position component (in microns) of the Front Shock parameter as input by the operator.
Domain: ASCII string representation of a natural number.

13.8 SET_DENSITY.CONTINUOUS_DENSITY_VACUUM_INPUT_LINE
Description: A displayed field in the dialog box containing the density component (in atmospheres) of the Vacuum Shock parameter as input by the operator.
Domain: ASCII string representation of a real number.

13.9 SET_DENSITY.CONTINUOUS_DENSITY_TAIL_INPUT_LINE
Description: A displayed field in the dialog box containing the density component (in atmospheres) of the Tail Shock parameter as input by the operator.
Domain: ASCII string representation of a real number.

13.10 SET_DENSITY.CONTINUOUS_DENSITY_SADDLE_INPUT_LINE

Description: A displayed field in the dialog box containing the density component (in atmospheres) of the Saddle Shock parameter as input by the operator.
Domain: ASCII string representation of a real number.

13.11 SET_DENSITY.CONTINUOUS_DENSITY_REAR_INPUT_LINE
Description: A displayed field in the dialog box containing the density component (in atmospheres) of the Rear Shock parameter as input by the operator.
Domain: ASCII string representation of a real number.

13.12 SET_DENSITY.CONTINUOUS_DENSITY_FRONT_INPUT_LINE
Description: A displayed field in the dialog box containing the density component (in atmospheres) of the Front Shock parameter as input by the operator.
Domain: ASCII string representation of a real number.

13.13 SET_DENSITY.FIRE_RAYS_FROM_INPUT_LINE
Description: A displayed field in the dialog box containing a value for the Raytrace Limits Fire Rays From attribute (in microns) input by the operator.
Domain: ASCII string representation of a natural number.

13.14 SET_DENSITY.FIRE_RAYS_TO_INPUT_LINE
Description: A displayed field in the dialog box containing a value for the Raytrace Limits Fire Rays To attribute (in microns) input by the operator.
Domain: ASCII string representation of a natural number.

13.15 SET_DENSITY.FIRE_RAYS_CHECK_BOX
Description: The value of a check by the operator setting the User Interface Fire Rays attribute, displayed in the dialog box.
Domain: Boolean.

14. STATUS_LINE

Attributes: SL_ID
Primary Identifiers: SL_ID
Description: The user interface desktop has a status line at the bottom.
Key Letter: RT_SL

14.1 STATUS_LINE.SL_ID
Description: The identity of a specific instance of the Status Line object. In practice there shall be only one, so this requirement will fall away in the coding phase.
Domain: Natural.

15. ITEM

Attributes: NAME
STATUS_LINE
HOTKEY_SEQUENCE
HIGHLIGHT_KEY
COMMAND
Primary Identifiers: NAME
Description: The status line has selectable indexed entries tied to certain commands.
Key Letter: RT_IT

15.1 ITEM.NAME
Description: The displayed name of the item entry. Mouse clicks on this activate the item.
Domain: ASCII Text.

15.2 ITEM.STATUS_LINE
Description: The identity of the Status Line to which the item is attached.
Relationship: R14:STATUS_LINE
Domain: Same as STATUS_LINE.SL_ID.

15.3 ITEM.HOTKEY_SEQUENCE
Description: A keyboard shortcut which will activate the item.
Domain: ASCII characters including control characters/function keys.

15.4 ITEM.HIGHLIGHT_KEY
Description: A highlit letter in the item name. A <Ctrl><Highlight Key> sequence activates the item.
Domain: ASCII character.

15.5 ITEM.COMMAND
Description: The command executed when the item is activated.
Domain: n/a.

16. HELP_ITEM

Attributes: NAME
Primary Identifiers: NAME
Description: This is the help entry on the status line. It invokes the context-sensitive help facility.
Key Letter: RT_HI

16.1 HELP_ITEM.NAME
Description: The name of the item.
Relationship: R16:ITEM
Domain: Same as ITEM.NAME.

17. MENU_ITEM

Attributes: NAME
Primary Identifiers: NAME
Description: This is the menu entry on the status line. It invokes the pull-down menu system.
Key Letter: RT_MI

17.1 MENU_ITEM.NAME
Description: The name of the item.
Relationship: R16:ITEM
Domain: Same as ITEM.NAME.

18. EXIT_ITEM
Attributes: NAME
Primary Identifiers: NAME
Description: This is the exit entry in the status line. It invokes the application exit sequence.
Key Letter: RT_EIT

18.1 EXIT_ITEM.NAME
Description: The name of the item.
Relationship: R16:ITEM
Domain: Same as ITEM.NAME.

19. MENU_BAR

Attributes: MB_ID

Primary Identifiers: MB_ID
Description: The user interface desktop has a top-level menu bar at the top of the screen.
Key Letter: RT_MB

19.1 MENU_BAR.MB_ID
Description: The identity of a specific instance of the Menu Bar object. In practice there shall be only one, so this requirement will fall away in the coding phase.
Domain: Natural:

20. MENU

Attributes: NAME
MENU_BAR
HOTKEY_SEQUENCE
HIGHLIGHT_KEY
COMMAND
UI_ID
MENU_SELECTION
Primary Identifiers: NAME
Description: Each entry in the menu bar is a pull-down submenu. This object is a non-functional ancestor from which successors may inherit.
Key Letter: RT_M

20.1 MENU.NAME
Description: The displayed name of the menu entry. Mouse clicks on this activate the menu.
Domain: ASCII text.

20.2 MENU.MENU_BAR
Description: The menu bar to which the item belongs.
Relationship: R15:MENU_BAR
Domain: Same as MENU_BAR.MB_ID.

20.3 MENU.HOTKEY_SEQUENCE
Description: A keyboard shortcut which will activate the menu.
Domain: ASCII characters including control characters/function keys.

20.4 MENU.HIGHLIGHT_KEY
Description: A highlit letter in the item name. A <Ctrl><Highlight Key> sequence activates the menu.
Domain: ASCII character.

20.5 MENU.COMMAND
Description: The command executed when the item is activated.
Domain: n/a.

20.6 MENU.UI_ID
Description: The identity of the user interface owning the menu bar to which the menu is attached.
Domain: Same as USER_INTERFACE.UI_ID.

20.7 MENU.MENU_SELECTION
Description: The item entry selected from the menu.
Domain: See descendent object fields.

21. INITIAL

Attributes: NAME
ABOUT
CALCULATOR
HELP
Primary Identifiers: NAME
Description: The INITIAL submenu provides access to tools, information and help.
Key Letter: RT_TM

21.1 INITIAL.NAME
Description: The name of the menu.
Relationship: R17:MENU
Domain: Same as MENU.NAME.

21.2 INITIAL.ABOUT
Description: A label indicating a menu option that displays credits/system information.
Domain: ASCII text.

21.3 INITIAL.CALCULATOR
Description: A label indicating a menu option that launches a simple arithmetic calculator.
Domain: ASCII text.

21.4 INITIAL.HELP
Description: A label indicating a menu option that invokes context-sensitive, cross-referenced help.
Domain: ASCII text.

22. SYSTEM
Attributes: NAME
DOS_SHELL
EXIT
Primary Identifiers: NAME
Description: The SYSTEM submenu providing access to the operating system.
Key Letter: RT_SM

22.1 SYSTEM.NAME
Description: The name of the menu.
Relationship: R17:MENU
Domain: Same as MENU.NAME.

22.2 SYSTEM.DOS_SHELL
Description: A label indicating a menu option that invokes a DOS operating system shell session.
Domain: ASCII text.

22.3 SYSTEM.EXIT
Description: A label indicating a menu option that exits the application and returns to the DOS operating system.
Domain: ASCII text.

23. RAYTRACE

Attributes: NAME
FIRE_RAYS
SAVE_PARAMETERS
LOAD_PARAMETERS
SAVE_RAW_DENSITIES

SAVE_RAW_INTENSITIES
DISPLAY_SHOCK
RUN_MACRO_FILE
DIALOG_ID

Primary Identifiers: NAME

Description: The RAYTRACE submenu provides access to the raytracing functions.

Key Letter: RT_RM

23.1 RAYTRACE.NAME

Description: The name of the menu.
Relationship: R17:MENU
Domain: Same as MENU.NAME.

23.2 RAYTRACE.FIRE_RAYS

Description: A label indicating a menu option that opens the Set Density dialog.
Domain: ASCII text.

23.3 RAYTRACE.SAVE_PARAMETERS

Description: A label indicating a menu option that opens a filename dialog to save a file.
Domain: ASCII text.

23.4 RAYTRACE.LOAD_PARAMETERS

Description: A label indicating a menu option that opens a filename dialog to load a file.
Domain: ASCII text.

23.5 RAYTRACE.SAVE_RAW_DENSITIES

Description: A label indicating a menu option that opens a filename dialog to save a file.
Domain: ASCII text.

23.6 RAYTRACE.SAVE_RAW_INTENSITIES

Description: A label indicating a menu option that opens a filename dialog to save a file.
Domain: ASCII text.

23.7 RAYTRACE.DISPLAY_SHOCK

Description: A label indicating a menu option that will redisplay the shock density and intensity graphs.
Domain: ASCII text.

23.8 RAYTRACE.RUN_MACRO_FILE

Description: A label indicating a menu option that will open a filename dialog to select a macro file to run.
Domain: ASCII text.

23.9 RAYTRACE.DIALOG_ID

Description: The identity of the dialog the Fire Rays menu entry will invoke.
Domain: Same as DIALOG_BOX.DIALOG_ID.

24. IMAGES

Attributes:
NAME
LOAD_IMAGE
DETERMINE_SCALE
EXTRACT_INTENSITY
SAVE_RAW_INTENSITIES
SAVE_INTENSITY_PEAKS
DISPLAY_IMAGE
DIALOG_ID

Primary Identifiers: NAME
Description: The IMAGES submenu provides access to the image manipulation functions.
Key Letter: RT_IM

24.1 IMAGES.NAME
Description: The name of the menu.
Relationship: R17:MENU
Domain: Same as MENU.NAME.

24.2 IMAGES.LOAD_IMAGE
Description: A label indicating a menu option that opens a filename dialog to select an image to load.
Domain: ASCII text.

24.3 IMAGES.DETERMINE_SCALE
Description: A label indicating a menu option that starts the image scaling exercise by displaying the scale bar over the image.
Domain: ASCII text.

24.4 IMAGES.EXTRACT_INTENSITY
Description: A label indicating a menu option that invokes the Extract Intensity dialog box.
Domain: ASCII text.

24.5 IMAGES.SAVE_RAW_INTENSITIES
Description: A label indicating a menu option that opens a filename dialog to save a file.
Domain: ASCII text.

24.6 IMAGES.SAVE_INTENSITY_PEAKS
Description: A label indicating a menu option that opens a filename dialog to save a file.
Domain: ASCII text.

24.7 IMAGES.DISPLAY_IMAGE
Description: A label indicating a menu option that will redisplay the image and the extracted intensity graph (if any).
Domain: ASCII text.

24.8 IMAGES.DIALOG_ID
Description: The identity of the dialog the Extract Intensity menu entry will invoke.
Domain: Same as DIALOG_BOX.DIALOG_ID.

D.10 Relationship Descriptions

R1.	SHOCK (1) has DENSITY_PARAMETERS (1) DENSITY_PARAMETERS (1) is part of SHOCK (1)
Description:	The shock has density parameters.
Formalisation:	SHOCK.DENSITY_PARAMETERS
R2.	RAYTRACE_LIMITS (1) is part of SHOCK (1) SHOCK (1) has RAYTRACE_LIMITS (1)
Description:	The shock has raytrace limits.
Formalisation:	SHOCK.RAYTRACE_LIMITS
R3.	COMPUTATIONAL_SUBSHELL (M) is part of SHOCK (1) SHOCK (1) has COMPUTATIONAL_SUBSHELL (M)
Description:	The shock contains many computation subshells.
Formalisation:	COMPUTATIONAL_SUBSHELL.SHOCK_ID
R4.	RAY (M) has position within RAYTRACE_LIMITS (1) RAYTRACE_LIMITS (1) constrains position of RAY (M)
Description:	The raytrace limits constrain the tracing of each ray.
Formalisation:	RAY.LIMITS_ID
R5.	COMPUTATIONAL_SUBSHELL (M) is traversed by RAY (M) RAY (M) traverses COMPUTATIONAL_SUBSHELL (M)
Description:	The traversal of a particular subshell by a particular ray.
Formalisation:	RAY_SUBSHELL_TRAVERSAL.RADIUS RAY_SUBSHELL_TRAVERSAL.RAY_ID
R6.	SCALE_BAR (1) belongs to IMAGE (1) IMAGE (1) has a SCALE_BAR (1)
Description:	The image has a scale bar.
Formalisation:	IMAGE.SCALE_BAR
R7.	SCAN_BAR (1) belongs to IMAGE (1) IMAGE (1) has a SCAN_BAR (1)
Description:	The image has a scan bar.
Formalisation:	IMAGE.SCAN_BAR
R8.	DIALOG_BOX is a supertype of EXTRACT_INTENSITY EXTRACT_INTENSITY is a subtype of DIALOG_BOX DIALOG_BOX is a supertype of SET_DENSITY SET_DENSITY is a subtype of DIALOG_BOX
Description:	Subtype-supertype inheritance relationship for dialog boxes.
Formalisation:	EXTRACT_INTENSITY.DIALOG_ID SET_DENSITY.DIALOG_ID
R9.	USER_INTERFACE (1) may open DIALOG_BOX (Mc) DIALOG_BOX (Mc) is opened by USER_INTERFACE (1)
Description:	The user interface may open many dialog boxes.
Formalisation:	DIALOG_BOX.DESKTOP_INTERFACE

R10.	USER_INTERFACE (1) controls SHOCK (1)
	SHOCK (1) is controlled by USER_INTERFACE (1)
Description:	The shock is manipulated from the user interface.
Formalisation:	USER_INTERFACE.SHOCK
R11.	IMAGE (1) is controlled by USER_INTERFACE (1)
	USER_INTERFACE (1) controls IMAGE (1)
Description:	The image is manipulated from the user interface.
Formalisation:	USER_INTERFACE.IMAGE
R12.	STATUS_LINE (1) belongs to USER_INTERFACE (1)
	USER_INTERFACE (1) has a STATUS_LINE (1)
Description:	The user interface has a single status line.
Formalisation:	USER_INTERFACE.STATUS_LINE
R13.	MENU_BAR (1) belongs to USER_INTERFACE (1)
	USER_INTERFACE (1) has a MENU_BAR (1)
Description:	The user interface has a single menu bar.
Formalisation:	USER_INTERFACE.MENU_BAR
R14.	ITEM (M) belongs to STATUS_LINE (1)
	STATUS_LINE (1) consists of ITEM (M)
Description:	The status line contains many selectable items.
Formalisation:	ITEM.STATUS_LINE
R15.	MENU (M) belongs to MENU_BAR (1)
	MENU_BAR (1) consists of MENU (M)
Description:	The menu bar contains many pull-down submenus.
Formalisation:	MENU.MENU_BAR
R16.	ITEM is a supertype of EXIT_ITEM
	EXIT_ITEM is a subtype of ITEM
	ITEM is a supertype of MENU_ITEM
	MENU_ITEM is a subtype of ITEM
	ITEM is a supertype of HELP_ITEM
	HELP_ITEM is a subtype of ITEM
Description:	Subtype-supertype inheritance relationship for status line items.
Formalisation:	EXIT_ITEM.NAME
	MENU_ITEM.NAME
	HELP_ITEM.NAME
R17.	MENU is a supertype of RAYTRACE
	RAYTRACE is a subtype of MENU
	MENU is a supertype of IMAGES
	IMAGES is a subtype of MENU
	MENU is a supertype of INITIAL
	INITIAL is a subtype of MENU
	MENU is a supertype of SYSTEM
	SYSTEM is a subtype of MENU
Description:	Subtype-supertype inheritance relationship for menu bar submenus.
Formalisation:	RAYTRACE.NAME
	IMAGES.NAME
	INITIAL.NAME
	SYSTEM.NAME

R18.	DENSITY_PARAMETERS (1) updates SHOCK (1)
	SHOCK (1) is updated by DENSITY_PARAMETERS (1)
Description:	The density parameters update the density profile of the shock.
Formalisation:	DENSITY_PARAMETERS.SHOCK_ID
R19.	RAYTRACE_LIMITS (1) updates SHOCK (1)
	SHOCK (1) is updated by RAYTRACE_LIMITS (1)
Description:	The raytrace limits update the shock attributes.
Formalisation:	RAYTRACE_LIMITS.SHOCK_ID
R20.	RAY (M) is fired by SHOCK (1)
	SHOCK (1) fires RAY (M)
Description:	The shock object controls the firing of all the rays.
Formalisation:	RAY.SHOCK_ID

D.11 State Transition Diagrams

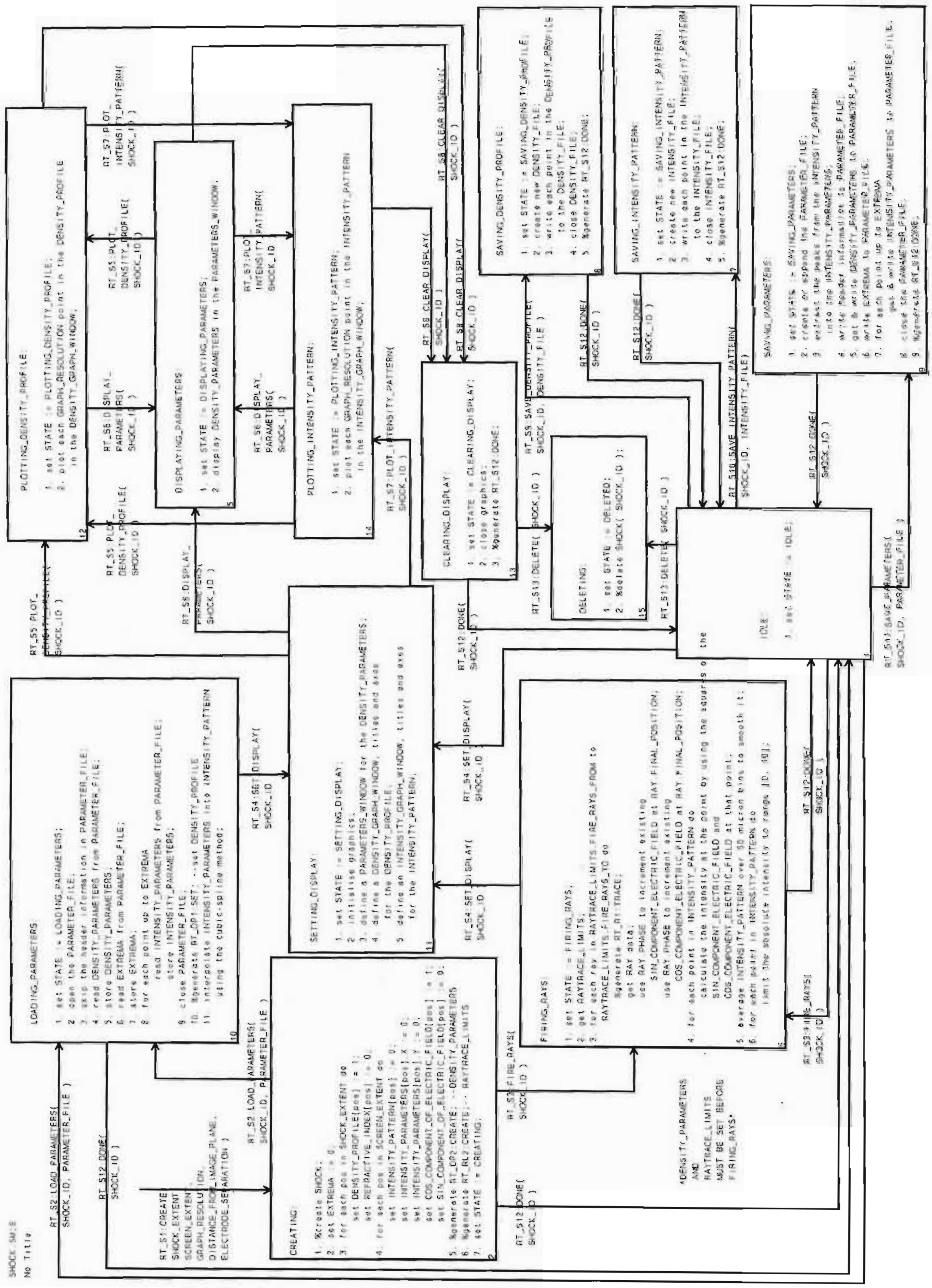


Figure D.2 State Transition Diagram for Object Shock

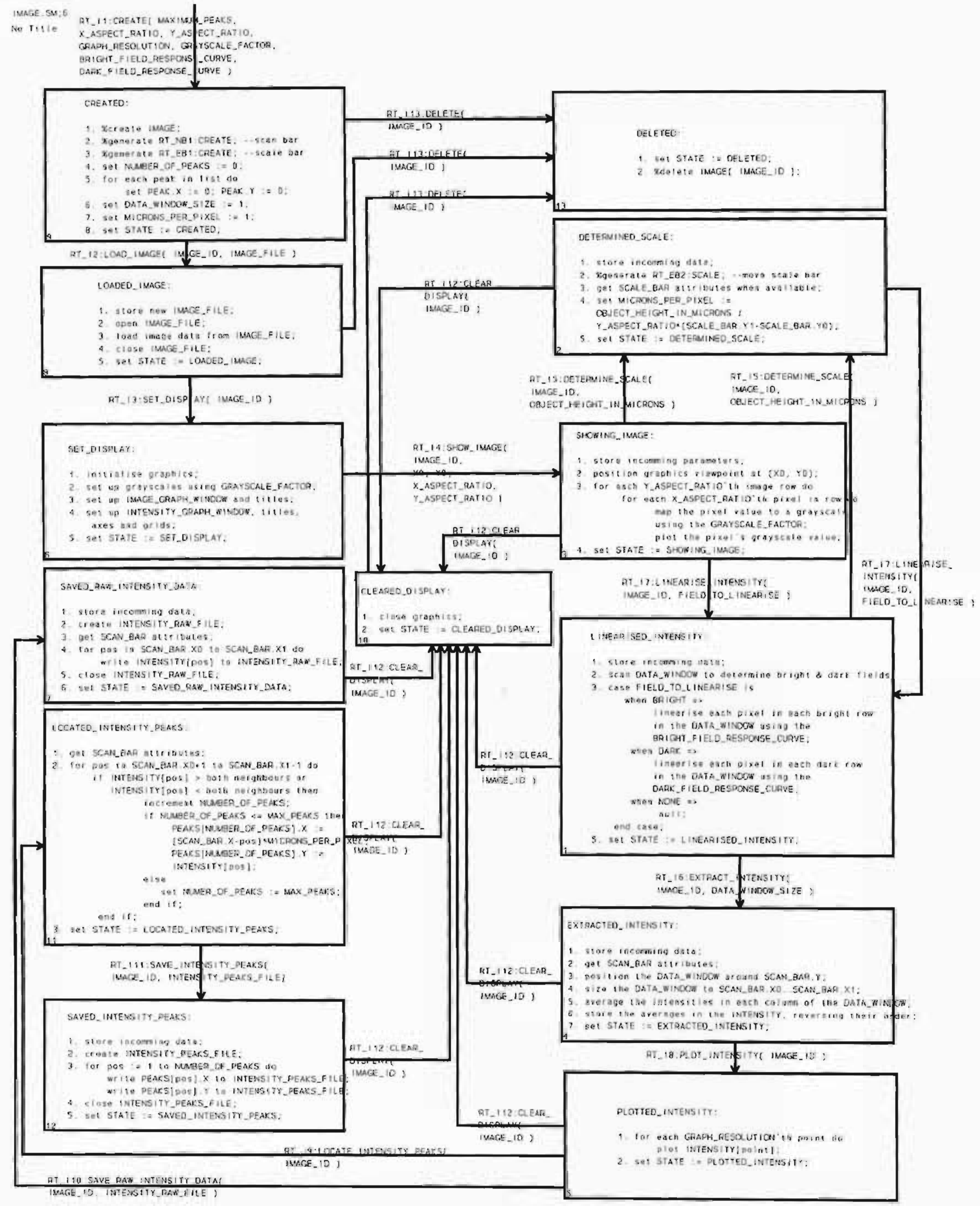


Figure D.3 State Transition Diagram for Object Image

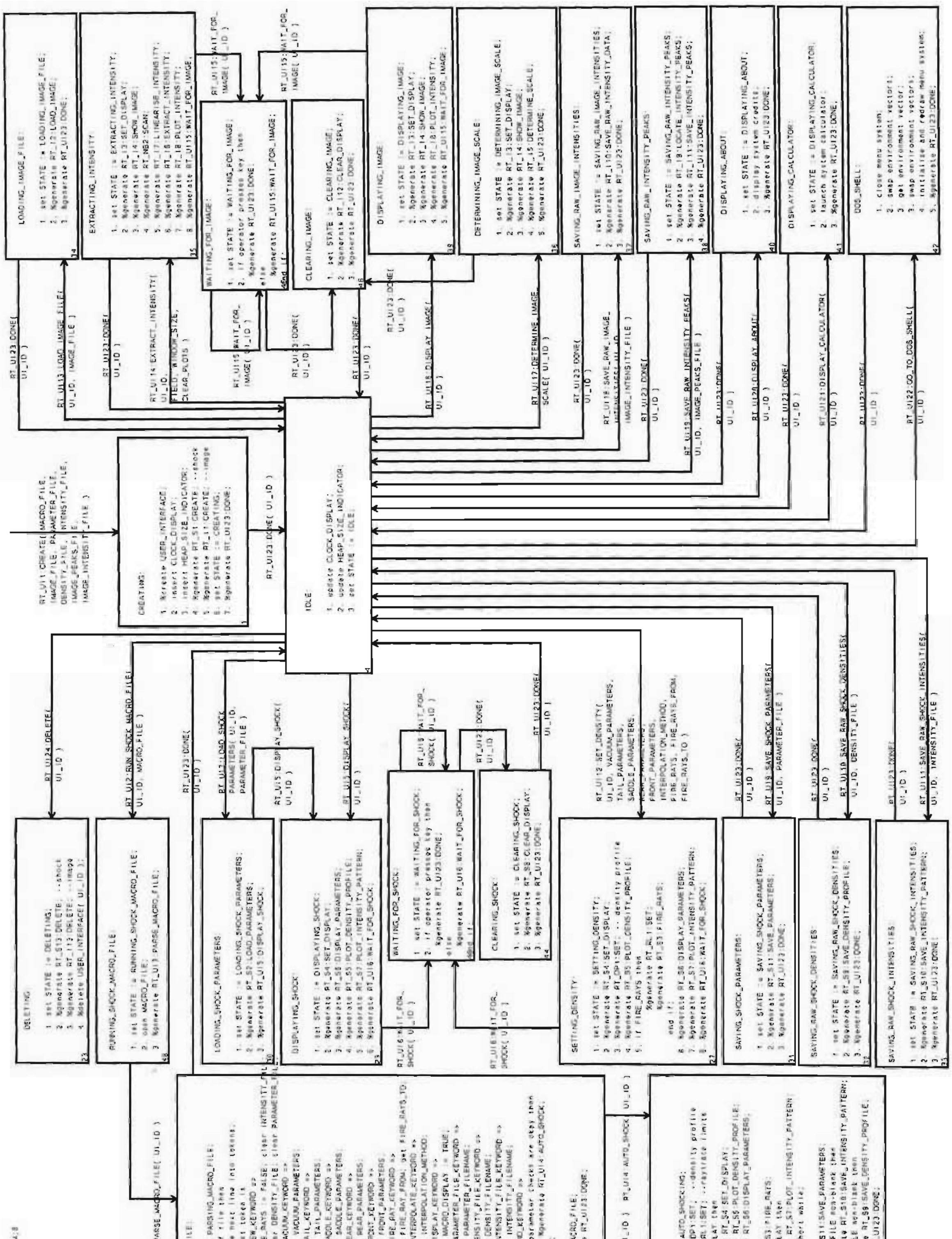


Figure D.4 State Transition Diagram for Object User Interface

```
COMPUTATIONAL_SUBSHELL.SM;5
No Title                          RT_CS1:TRAVERSE( RADIUS,
                                  INCIDENT_ANGLE,
                                  INCIDENT_HEIGHT )

TEMPORAL:

1. use the RADIUS (subshell position) and REFRACTIVE_INDEX,
   the INCIDENT_HEIGHT and INCIDENT_ANGLE of the ray to calculate the
   REFRACTED_ANGLE from Snell's Law;
2. use the REFRACTED_ANGLE and the SUBSHELL WIDTH to calculate the
   OPTICAL_PATH_LENGTH travelled by the ray within the subshell;
3. use the REFRACTED_ANGLE, SUBSHELL_WIDTH and
   OPTICAL_PATH_LENGTH to calculate the EMERGENT_HEIGHT of the
   ray from the subshell.
1
```

Figure D.5 State Transition Diagram for Object Computational Subshell

```
RAY.SM;4
No Title

                    RT_R1:TRACE( INITIAL POSITION )

TEMPORAL:

1. determine the FIRST_FREE_OPTICAL_PATH_LENGTH from a plane normal to the incident
   ray and touching the shock front (boundary of the outermost subshell and free air);
2. for each COMPUTATIONAL_SUBSHELL (indexed by radius) do
      %generate RT_CS1:TRAVERSE;
      get COMPUTATIONAL_SUBSHELL attributes;
      set OPTICAL_PATH_LENGTH_IN_SHOCK :=
          OPTICAL_PATH_LENGTH_IN_SHOCK +
          COMPUTATIONAL_SUBSHELL.OPTICAL_PATH_LENGTH;
3. determine the SECOND_FREE_OPTICAL_PATH_LENGTH from the
   point the ray leaves the shock to the image plane which is positioned
   SHOCK.DISTANCE_FROM_IMAGE_PLANE away from the centre of the shock;
4. set TOTAL_OPTICAL_PATH_LENGTH := FIRST_FREE_OPTICAL_PATH_LENGTH
   + OPTICAL_PATH_LENGTH_IN_SHOCK + SECOND_FREE_OPTICAL_PATH_LENGTH;
5. use the TOTAL_OPTICAL_PATH_LENGTH and WAVELENGTH to
   determine the FINAL_POINT and PHASE of the ray when it arrives at the image plane;
1
```

Figure D.6 State Transition Diagram for Object Ray

```
RAY_SUBSHELL_TRAVERSAL.SM;1
No Title

        RT_RST1:CREATE( RADIUS, RAY_ID )

TEMPORAL:

1. create RAY_SUBSHELL_TRAVERSAL;
1
```

Figure D.7 State Transition Diagram for Object Ray Subshell Traversal

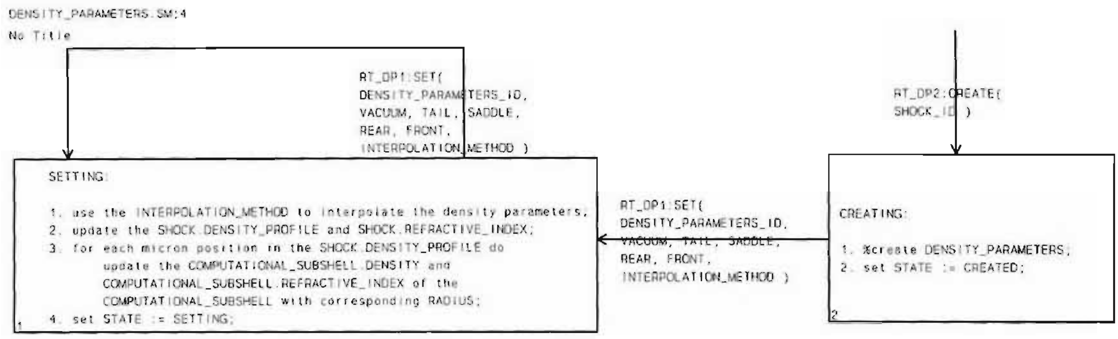


Figure D.8 State Transition Diagram for Object Density Parameters

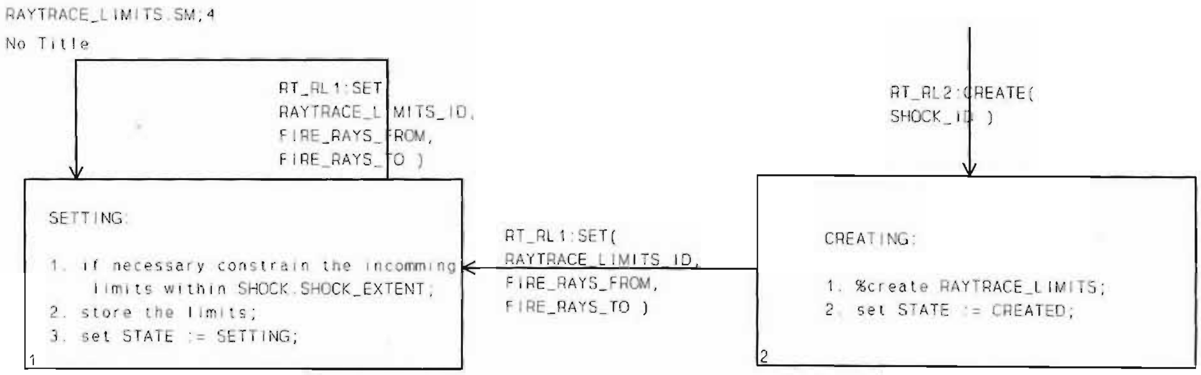


Figure D.9 State Transition Diagram for Object Raytrace Limits

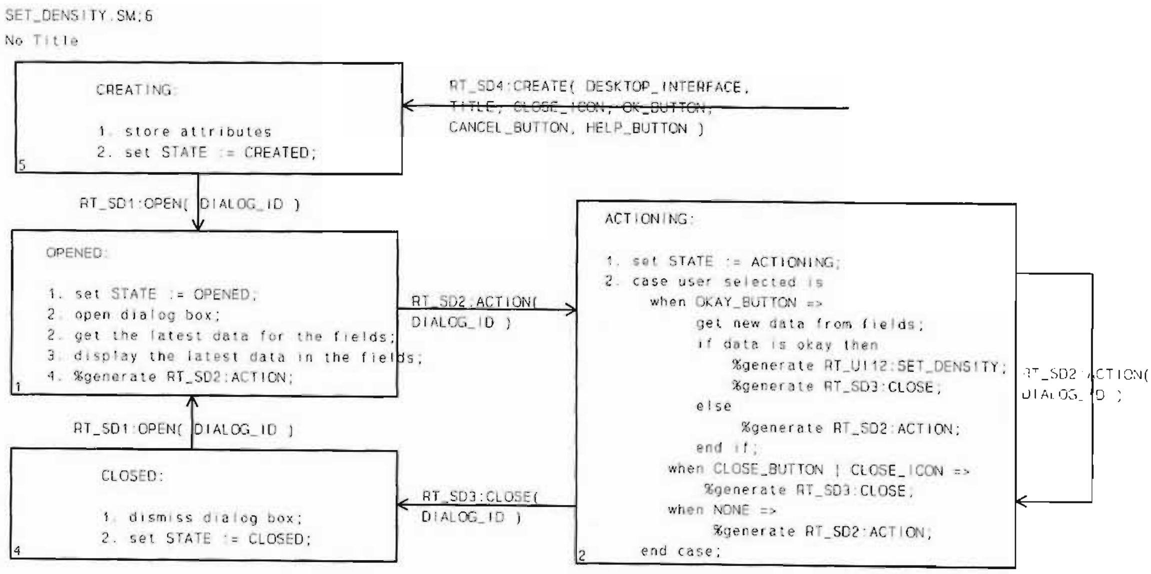


Figure D.10 State Transition Diagram for Object Set Density

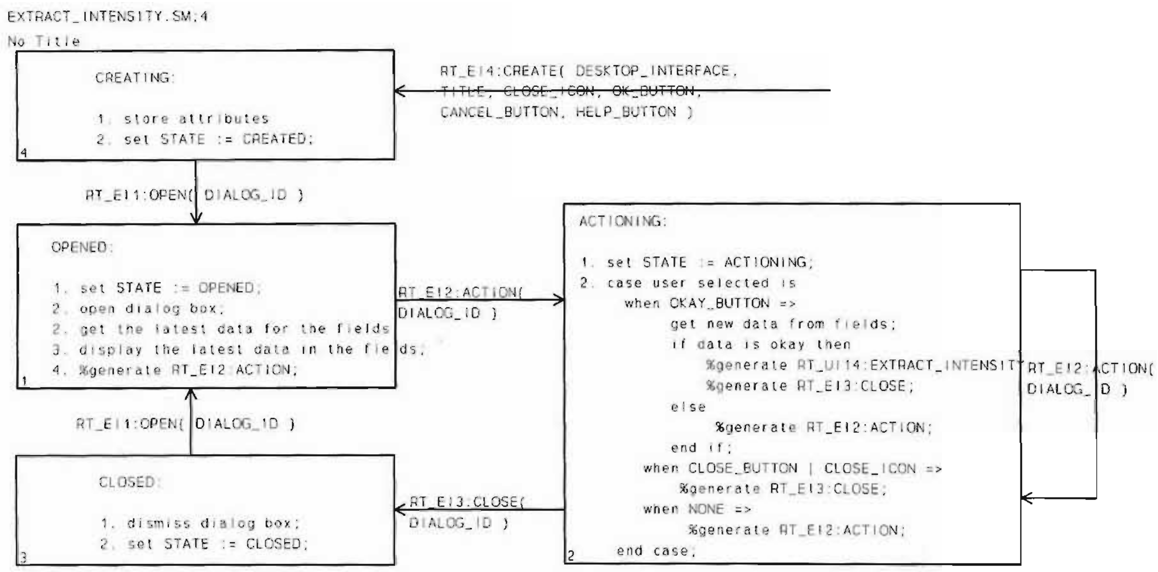


Figure D.11 State Transition Diagram for Object Extract Intensity

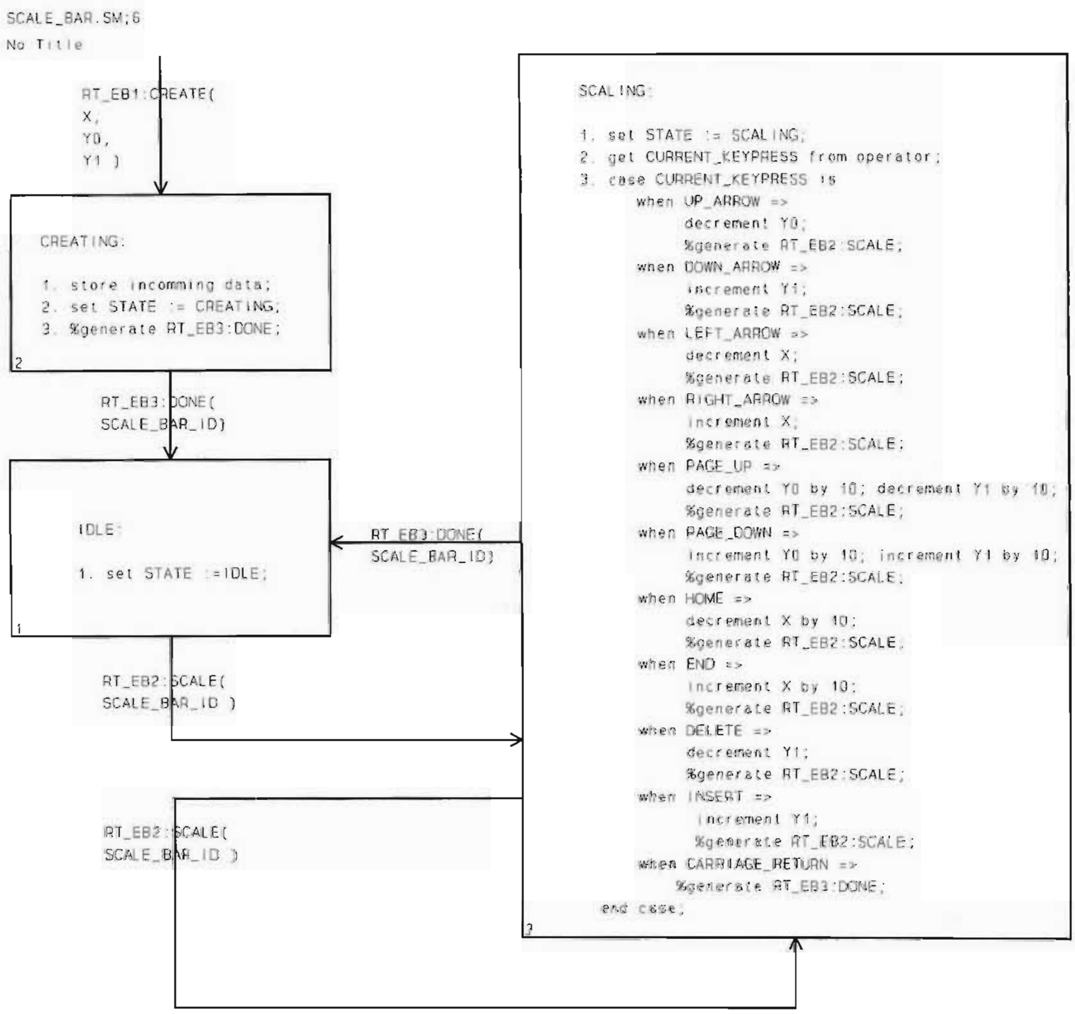


Figure D.12 State Transition Diagram for Object Scale Bar

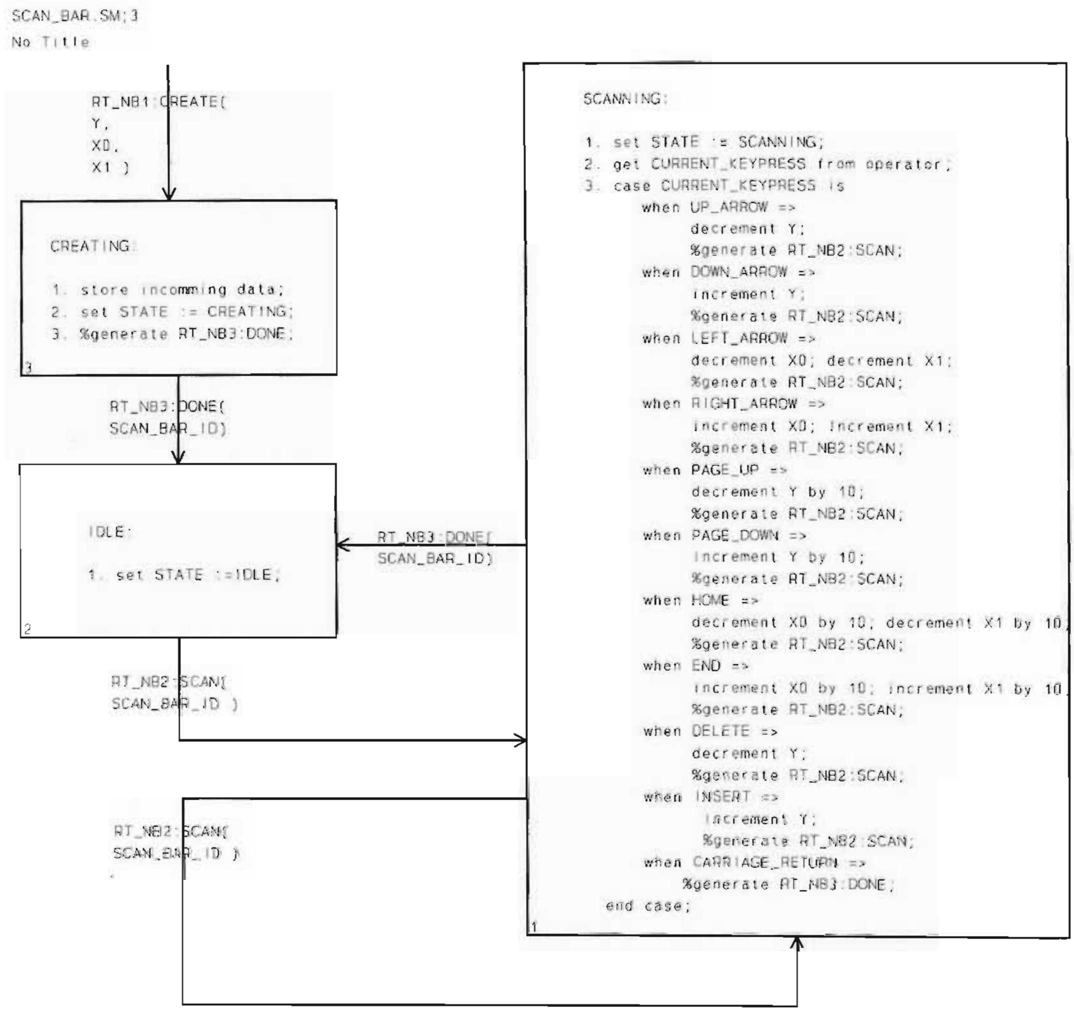


Figure D.13 State Transition Diagram for Object Scan Bar

```
IMAGES.SM;3
No Title
  RT_IM1:SELECT( NAME, MENU_SELECTION, UI_ID, DIALOG_ID )

  TEMPORAL:

  1. case MENU_SELECTION is
         when LOAD_IMAGE =>
             %generate RT_UI13:LOAD_IMAGE_FILE;
         when DETERMINE_SCALE =>
             %generate RT_UI17:DETERMINE_IMAGE_SCALE;
         when EXTRACT_INTENSITY =>
             %generate RT_EI1:OPEN;
         when SAVE_RAW_INTENSITIES =>
             %generate RT_UI18:SAVE_RAW_IMAGE_INTENSITIES;
         when SAVE_INTENSITY_PEAKS =>
             %generate RT_UI19:SAVE_RAW_INTENSITY_PEAKS;
         when DISPLAY_IMAGE =>
             %generate RT_UI16:DISPLAY_IMAGE;
     end case;
1
```

Figure D.14 State Transition Diagram for Object Image Menu

```
RAYTRACE.SM;5
No Title
  RT_RM1:SELECT( NAME, MENU_SELECTION, UI_ID, DIALOG_ID )

  TEMPORAL:

  1. case MENU_SELECTION is
         when FIRE_RAYS =>
             %generate RT_SD1:OPEN;
         when SAVE_PARAMETERS =>
             %generate RT_UI9:SAVE_SHOCK_PARAMETERS;
         when LOAD_PARAMETERS =>
             %generate RT_UI7:LOAD_SHOCK_PARAMETERS;
         when SAVE_RAW_DENSITIES =>
             %generate RT_UI10:SAVE_RAW_SHOCK_DENSITIES;
         when SAVE_RAW_INTENSITIES =>
             %generate RT_UI11:SAVE_RAW_SHOCK_INTENSITIES;
         when DISPLAY_SHOCK =>
             %generate RT_UI5:DISPLAY_SHOCK;
         when RUN_MACRO_FILE =>
             %generate RT_UI2:RUN_SHOCK_MACRO_FILE;
     end case;
1
```

Figure D.15 State Transition Diagram for Object Raytrace Menu

```
SYSTEM.SM;4
No Title
   RT_SM1:SELECT( NAME, MENU_SELECTION, UI_ID )

   TEMPORAL:

   1. case MENU_SELECTION is
         when DOS_SHELL =>
               %generate RT_UI22:GO_TO_DOS_SHELL;
         when EXIT =>
               %generate RT_UI24:DELETE;
       end case;

1
```

Figure D.16 State Transition Diagram for Object System Menu

```
INITIAL.SM;5
No Title
   RT_TM1:SELECT( NAME, MENU_SELECTION, UI_ID )

   TEMPORAL.

   1. case MENU_SELECTION is
         when ABOUT =>
               %generate RT_UI20:DISPLAY_ABOUT;
         when CALCULATOR =>
               %generate RT_UI21:DISPLAY_CALCULATOR;
         when HELP =>
               open context-sensitive help system;
      end case;

1
```

Figure D.17 State Transition Diagram for Object Initial Menu

D.12 Event List

Table D.2 Event List				
LABEL	**MEANING**	**EVENT DATA**	**SOURCE**	**DESTINATION**
RT_CS1	TRAVERSE	RADIUS + INCIDENT_ANGLE + INCIDENT_HEIGHT	RAY	COMPUTATIONAL_ SUBSHELL
RT_DP1	SET	DENSITY_ PARAMETERS_ID VACUUM + TAIL + SADDLE + REAR + FRONT + INTERPOLATION_ METHOD	USER_INTERFACE	DENSITY_ PARAMETERS
RT_DP1	SET	DENSITY_ PARAMETERS_ID + VACUUM + TAIL + SADDLE + REAR + FRONT + INTERPOLATION_ METHOD	SHOCK	DENSITY_ PARAMETERS
RT_DP2	CREATE	SHOCK_ID	SHOCK	DENSITY_ PARAMETERS
RT_EB1	CREATE	X + Y0 + Y1	IMAGE	SCALE_BAR
RT_EB2	SCALE	SCALE_BAR_ID	SCALE_BAR	SCALE_BAR
RT_EB2	SCALE	SCALE_BAR_ID	IMAGE	SCALE_BAR
RT_EB3	DONE	SCALE_BAR_ID	SCALE_BAR	SCALE_BAR
RT_EI1	OPEN	DIALOG_ID	IMAGES	EXTRACT_INTENSIT Y
RT_EI2	ACTION	DIALOG_ID	EXTRACT_INTENSITY	EXTRACT_INTENSIT Y
RT_EI3	CLOSE	DIALOG_ID	EXTRACT_INTENSITY	EXTRACT_INTENSIT Y
RT_EI4	CREATE	DESKTOP_ INTERFACE + TITLE + CLOSE_ICON + OK_BUTTON + CANCEL_BUTTON + HELP_BUTTON	-	EXTRACT_INTENSIT Y

Table D.2 Event List				
LABEL	**MEANING**	**EVENT DATA**	**SOURCE**	**DESTINATION**
RT_I1	CREATE	MAXIMUM_PEAKS + X_ASPECT_RATIO + Y_ASPECT_RATIO + GRAPH_RESOLUTION + GRAYSCALE_FACTOR + BRIGHT_FIELD_ RESPONSE_CURVE + DARK_FIELD_ RESPONSE_CURVE	USER_INTERFACE	IMAGE
RT_I10	SAVE_RAW_ INTENSITY_DATA	IMAGE_ID + INTENSITY_RAW_FIL E	USER_INTERFACE	IMAGE
RT_I11	SAVE_INTENSITY_ PEAKS	IMAGE_ID + INTENSITY_PEAKS_ FILE	USER_INTERFACE	IMAGE
RT_I12	CLEAR_DISPLAY	IMAGE_ID	USER_INTERFACE	IMAGE
RT_I13	DELETE	IMAGE_ID	USER_INTERFACE	IMAGE
RT_I2	LOAD_IMAGE	IMAGE_ID + IMAGE_FILE	USER_INTERFACE	IMAGE
RT_I3	SET_DISPLAY	IMAGE_ID	USER_INTERFACE	IMAGE
RT_I4	SHOW_IMAGE	IMAGE_ID + X0 + Y0 + X_ASPECT_RATIO + Y_ASPECT_RATIO	USER_INTERFACE	IMAGE
RT_I5	DETERMINE_SCALE	IMAGE_ID + OBJECT_HEIGHT_ IN_MICRONS	USER_INTERFACE	IMAGE
RT_I6	EXTRACT_INTENSITY	IMAGE_ID + DATA_WINDOW_SIZE	USER_INTERFACE	IMAGE
RT_I7	LINEARISE_ INTENSITY	IMAGE_ID + FIELD_TO_LINEARISE	USER_INTERFACE	IMAGE
RT_I8	PLOT_INTENSITY	IMAGE_ID	USER_INTERFACE	IMAGE
RT_I9	LOCATE_INTENSITY_ PEAKS	IMAGE_ID	USER_INTERFACE	IMAGE
RT_IM1	SELECT	NAME + MENU_SELECTION + UI_ID + DIALOG_ID	-	IMAGES
RT_NB1	CREATE	Y + X0 + X1	IMAGE	SCAN_BAR
RT_NB2	SCAN	SCAN_BAR_ID	USER_INTERFACE	SCAN_BAR

Table D.2 Event List				
LABEL	MEANING	EVENT DATA	SOURCE	DESTINATION
RT_NB2	SCAN	SCAN_BAR_ID	SCAN_BAR	SCAN_BAR
RT_NB3	DONE	SCAN_BAR_ID	SCAN_BAR	SCAN_BAR
RT_R1	TRACE	INITIAL_POSITION	SHOCK	RAY
RT_RL1	SET	RAYTRACE_LIMITS_ID + FIRE_RAYS_FROM + FIRE_RAYS_TO	USER_INTERFACE	RAYTRACE_LIMITS
RT_RL2	CREATE	SHOCK_ID	SHOCK	RAYTRACE_LIMITS
RT_RM1	SELECT	NAME + MENU_SELECTION + UI_ID + DIALOG_ID	-	RAYTRACE
RT_RST1	CREATE	RADIUS + RAY_ID	-	RAY_SUBSHELL_TRAVERSAL
RT_S1	CREATE	SHOCK_EXTENT + SCREEN_EXTENT + GRAPH_RESOLUTION + DISTANCE_FROM_IMAGE_PLANE + ELECTRODE_SEPARATION	USER_INTERFACE	SHOCK
RT_S10	SAVE_INTENSITY_PATTERN	SHOCK_ID + INTENSITY_FILE	USER_INTERFACE	SHOCK
RT_S11	SAVE_PARAMETERS	SHOCK_ID + PARAMETER_FILE	USER_INTERFACE	SHOCK
RT_S12	DONE	SHOCK_ID	SHOCK	SHOCK
RT_S13	DELETE	SHOCK_ID	USER_INTERFACE	SHOCK
RT_S2	LOAD_PARAMETERS	SHOCK_ID + PARAMETER_FILE	USER_INTERFACE	SHOCK
RT_S3	FIRE_RAYS	SHOCK_ID	USER_INTERFACE	SHOCK
RT_S4	SET_DISPLAY	SHOCK_ID	USER_INTERFACE	SHOCK
RT_S5	PLOT_DENSITY_PROFILE	SHOCK_ID	USER_INTERFACE	SHOCK
RT_S6	DISPLAY_PARAMETERS	SHOCK_ID	USER_INTERFACE	SHOCK
RT_S7	PLOT_INTENSITY_PATTERN	SHOCK_ID	USER_INTERFACE	SHOCK
RT_S8	CLEAR_DISPLAY	SHOCK_ID	USER_INTERFACE	SHOCK

Table D.2 Event List				
LABEL	**MEANING**	**EVENT DATA**	**SOURCE**	**DESTINATION**
RT_S9	SAVE_DENSITY_ PROFILE	SHOCK_ID + DENSITY_FILE	USER_INTERFACE	SHOCK
RT_SD1	OPEN	DIALOG_ID	RAYTRACE	SET_DENSITY
RT_SD2	ACTION	DIALOG_ID	SET_DENSITY	SET_DENSITY
RT_SD3	CLOSE	DIALOG_ID	SET_DENSITY	SET_DENSITY
RT_SD4	CREATE	DESKTOP_ INTERFACE + TITLE + CLOSE_ICON + OK_BUTTON + CANCEL_BUTTON + HELP_BUTTON	-	SET_DENSITY
RT_SM1	SELECT	NAME + MENU_SELECTION + UI_ID	-	SYSTEM
RT_TM1	SELECT	NAME + MENU_SELECTION + UI_ID	-	INITIAL
RT_UI1	CREATE	MACRO_FILE + IMAGE_FILE + PARAMETER_FILE + DENSITY_FILE + INTENSITY_FILE + IMAGE_PEAKS_FILE + IMAGE_INTENSITY_ FILE	-	USER_INTERFACE
RT_UI10	SAVE_RAW_SHOCK_ DENSITIES	UI_ID + DENSITY_FILE	RAYTRACE	USER_INTERFACE
RT_UI11	SAVE_RAW_SHOCK_ INTENSITIES	UI_ID + INTENSITY_FILE	RAYTRACE	USER_INTERFACE

Table D.2 Event List				
LABEL	**MEANING**	**EVENT DATA**	**SOURCE**	**DESTINATION**
RT_UI12	SET_DENSITY	UI_ID + VACUUM_ PARAMETERS + TAIL_PARAMETERS + SADDLE_ PARAMETERS + REAR_PARAMETERS + FRONT_ PARAMETERS + INTERPOLATION_ METHOD + FIRE_RAYS + FIRE_RAYS_FROM + FIRE_RAYS_TO	SET_DENSITY	USER_INTERFACE
RT_UI13	LOAD_IMAGE_FILE	UI_ID + IMAGE_FILE	IMAGES	USER_INTERFACE
RT_UI14	EXTRACT_INTENSITY	UI_ID + FIELD + WINDOW_SIZE + CLEAR_PLOTS	EXTRACT_INTENSITY	USER_INTERFACE
RT_UI15	WAIT_FOR_IMAGE	UI_ID	USER_INTERFACE	USER_INTERFACE
RT_UI16	DISPLAY_IMAGE	UI_ID	IMAGES	USER_INTERFACE
RT_UI17	DETERMINE_IMAGE_ SCALE	UI_ID	IMAGES	USER_INTERFACE
RT_UI18	SAVE_RAW_IMAGE_ INTENSITIES	UI_ID + IMAGE_ INTENSITY_FILE	IMAGES	USER_INTERFACE
RT_UI19	SAVE_RAW_ INTENSITY_PEAKS	UI_ID + IMAGE_PEAKS_FILE	IMAGES	USER_INTERFACE
RT_UI2	RUN_SHOCK_ MACRO_FILE	UI_ID + MACRO_FILE	RAYTRACE	USER_INTERFACE
RT_UI20	DISPLAY_ABOUT	UI_ID	INITIAL	USER_INTERFACE
RT_UI21	DISPLAY_ CALCULATOR	UI_ID	INITIAL	USER_INTERFACE
RT_UI22	GO_TO_DOS_SHELL	UI_ID	SYSTEM	USER_INTERFACE
RT_UI23	DONE	UI_ID	USER_INTERFACE	USER_INTERFACE
RT_UI24	DELETE	UI_ID	SYSTEM	USER_INTERFACE
RT_UI3	PARSE_MACRO_FILE	UI_ID	USER_INTERFACE	USER_INTERFACE
RT_UI4	AUTO_SHOCK	UI_ID	USER_INTERFACE	USER_INTERFACE
RT_UI5	DISPLAY_SHOCK	UI_ID	RAYTRACE	USER_INTERFACE

Table D.2 Event List				
LABEL	**MEANING**	**EVENT DATA**	**SOURCE**	**DESTINATION**
RT_UI5	DISPLAY_SHOCK	UI_ID	USER_INTERFACE	USER_INTERFACE
RT_UI6	WAIT_FOR_SHOCK	UI_ID	USER_INTERFACE	USER_INTERFACE
RT_UI7	LOAD_SHOCK_ PARAMETERS	UI_ID + PARAMETER_FILE	RAYTRACE	USER_INTERFACE
RT_UI9	SAVE_SHOCK_ PARAMETERS	UI_ID + PARAMETER_FILE	RAYTRACE	USER_INTERFACE

D.13 Data Flows

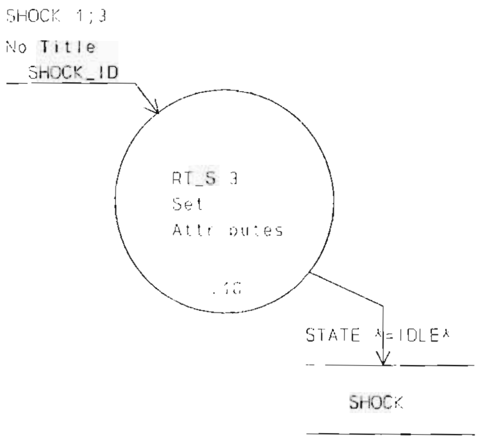


Figure D.18 Data Flow Diagram for Object Shock in State Idle

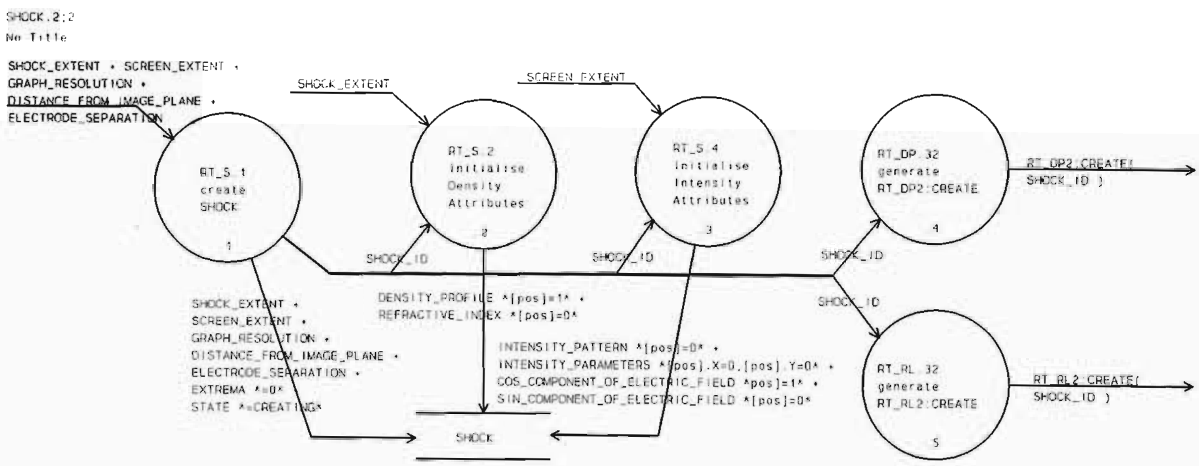


Figure D.19 Data Flow Diagram for Object Shock in State Creating

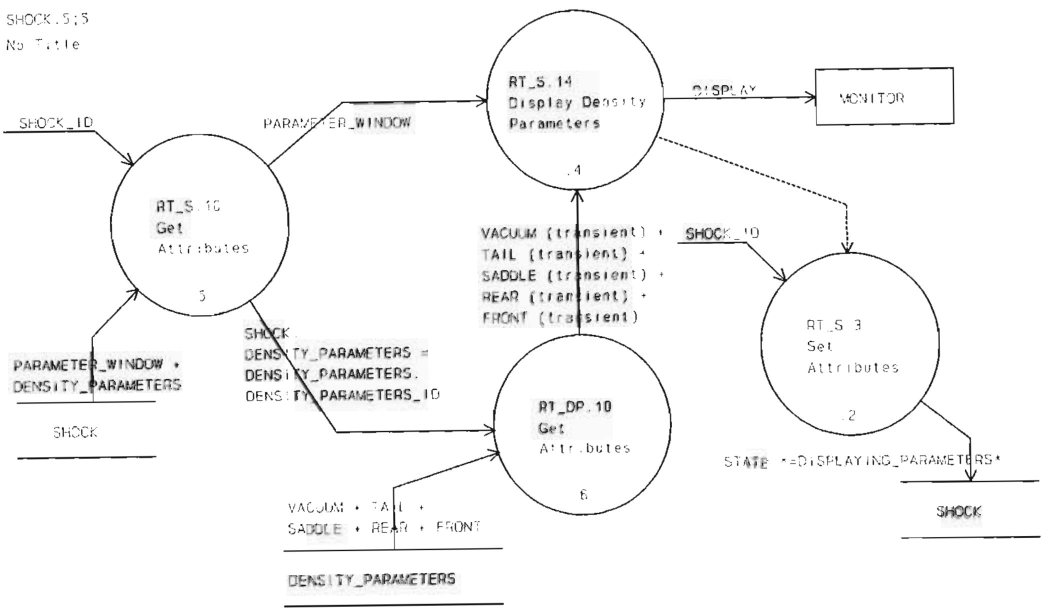


Figure D.20 Data Flow Diagram for Object Shock in State Displaying Parameters

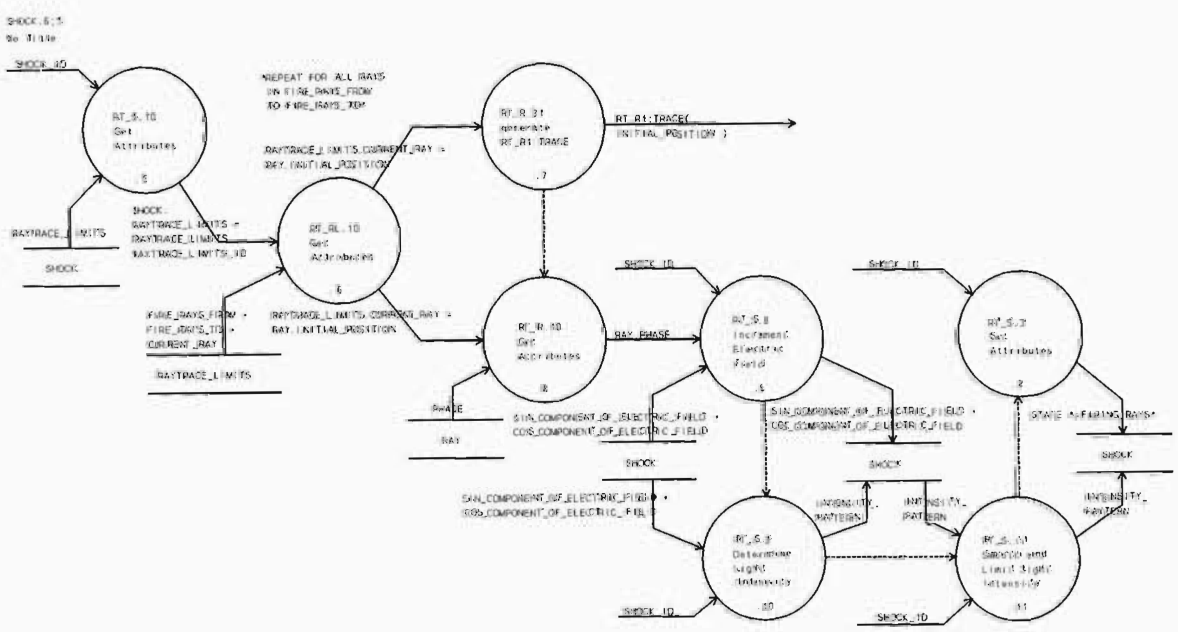


Figure D.21 Data Flow Diagram for Object Shock in State Firing Rays

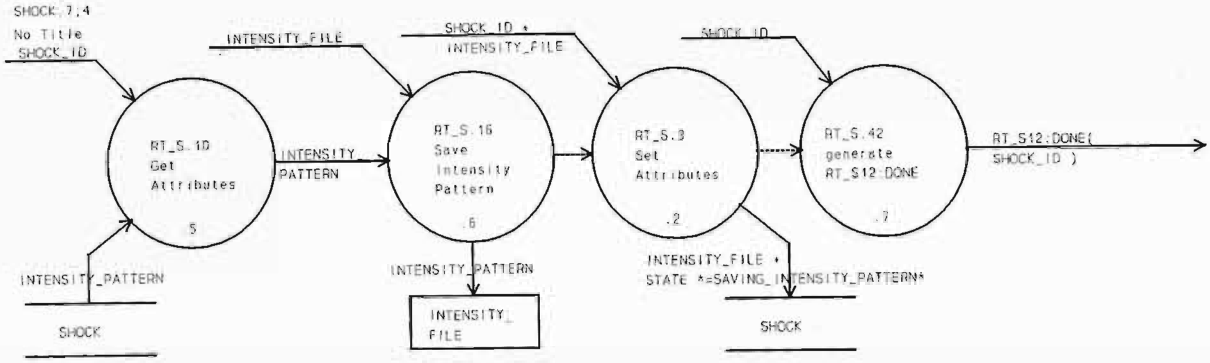


Figure D.22 Data Flow Diagram for Object Shock in State Saving Intensity Pattern

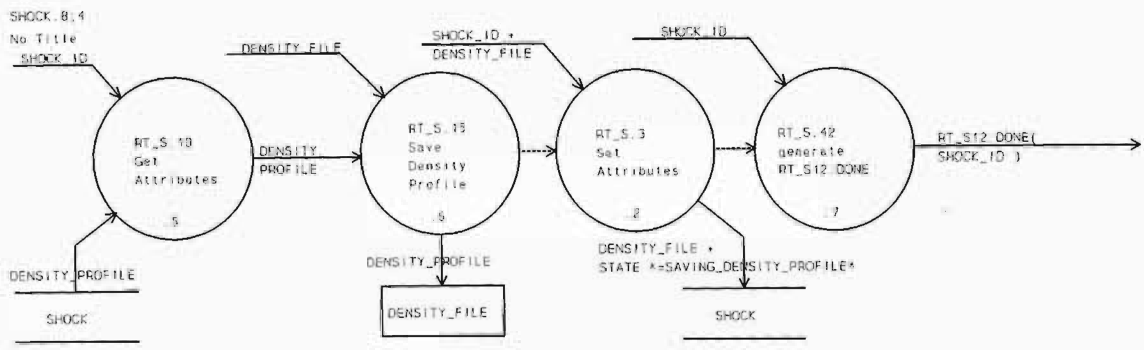


Figure D.23 Data Flow Diagram for Object Shock in State Saving Density Profile

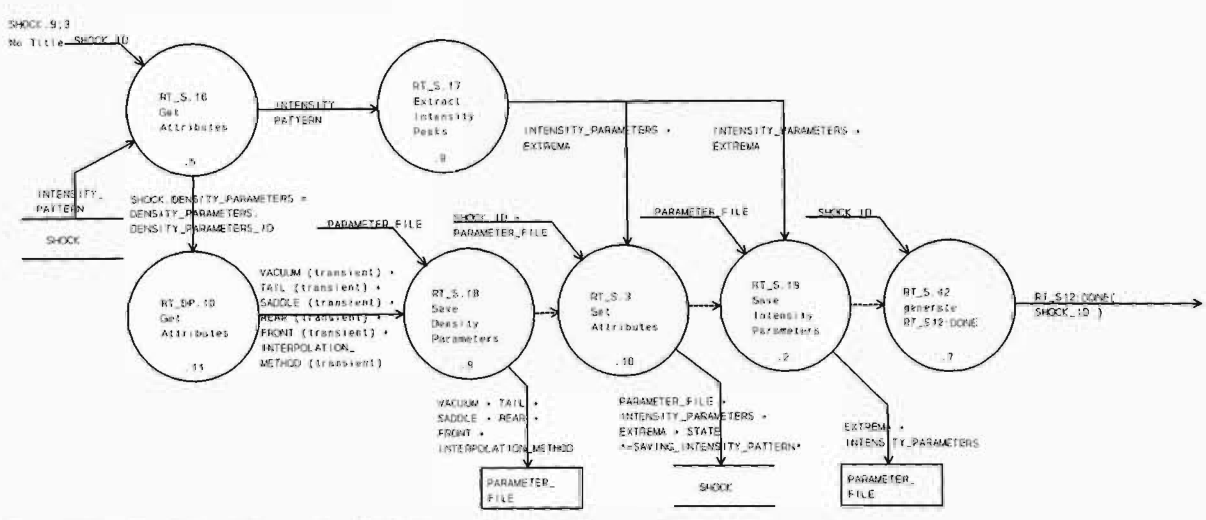


Figure D.24 Data Flow Diagram for Object Shock in State Saving Parameters

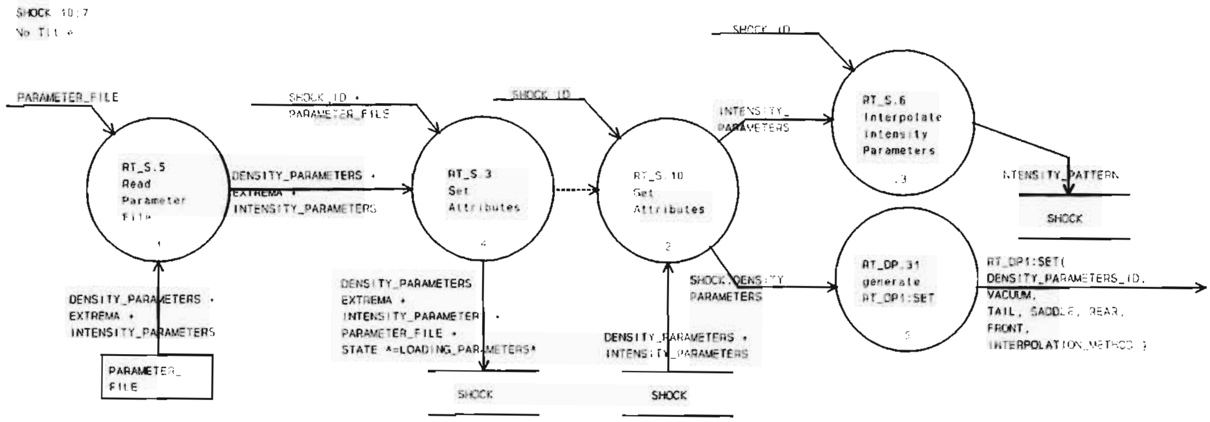


Figure D.25 Data Flow Diagram for Object Shock in State Loading Parameters

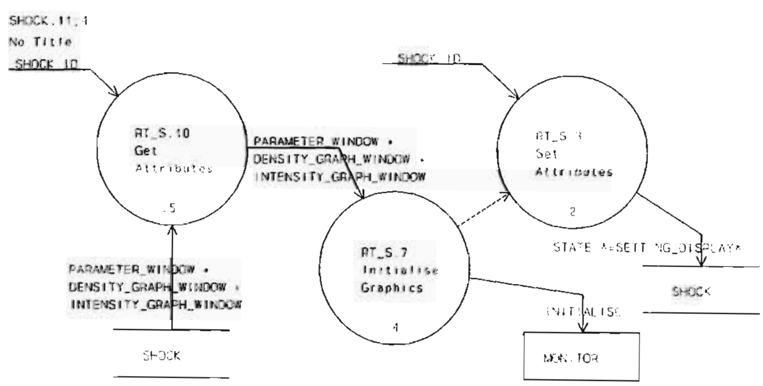


Figure D.26 Data Flow Diagram for Object Shock in State Setting Display

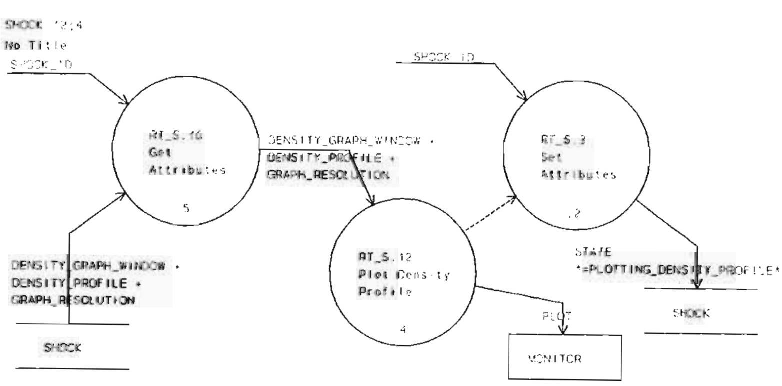


Figure D.27 Data Flow Diagram for Object Shock in State Plotting Density Profile

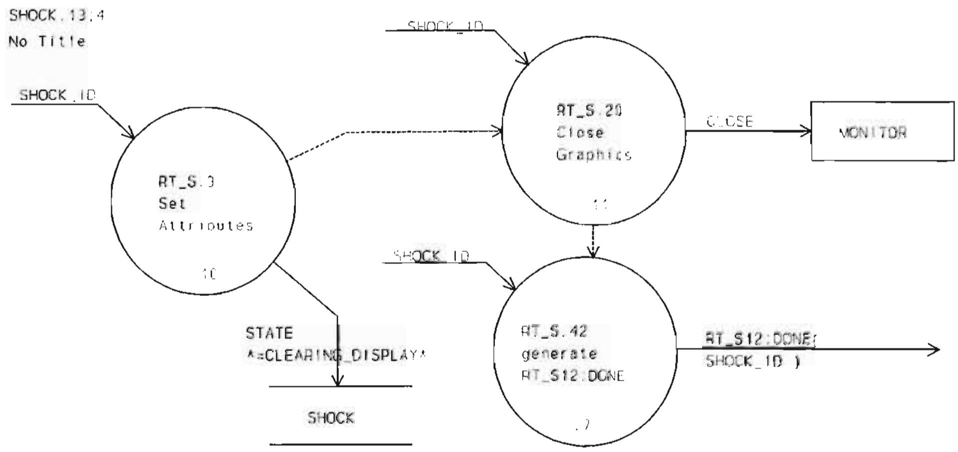


Figure D.28 Data Flow Diagram for Object Shock in State Clearing Display

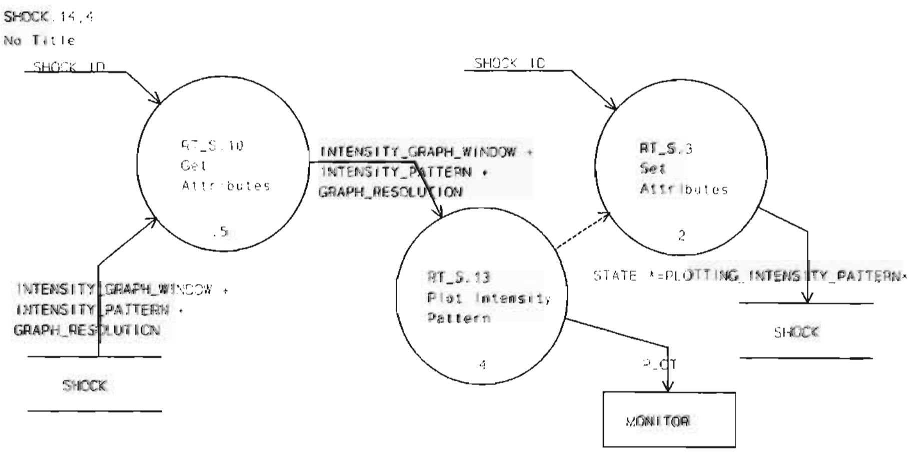


Figure D.29 Data Flow Diagram for Object Shock in State Plotting Intensity Pattern

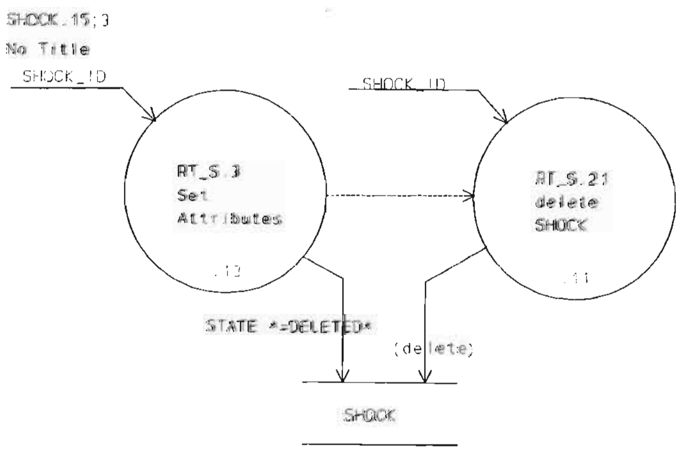


Figure D.30 Data Flow Diagram for Object Shock in State Deleting

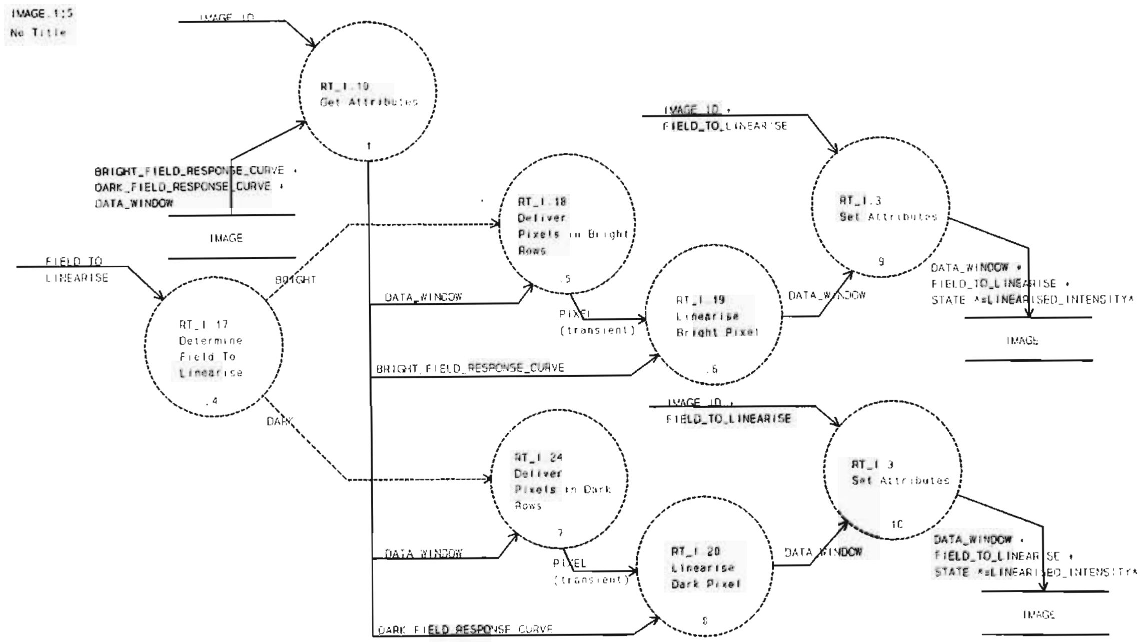


Figure D.31 Data Flow Diagram for Object Image in State Linearised Intensity

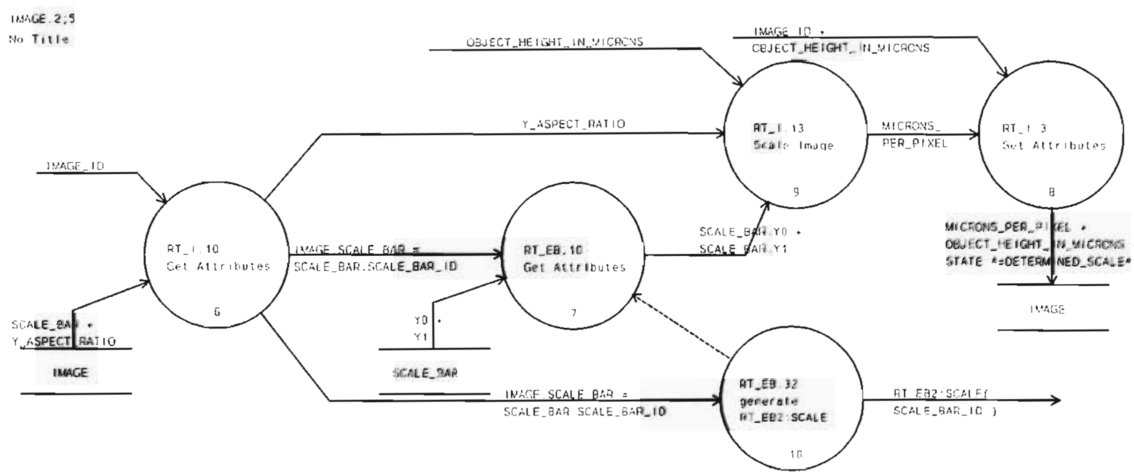


Figure D.32 Data Flow Diagram for Object Image in State Determined Scale

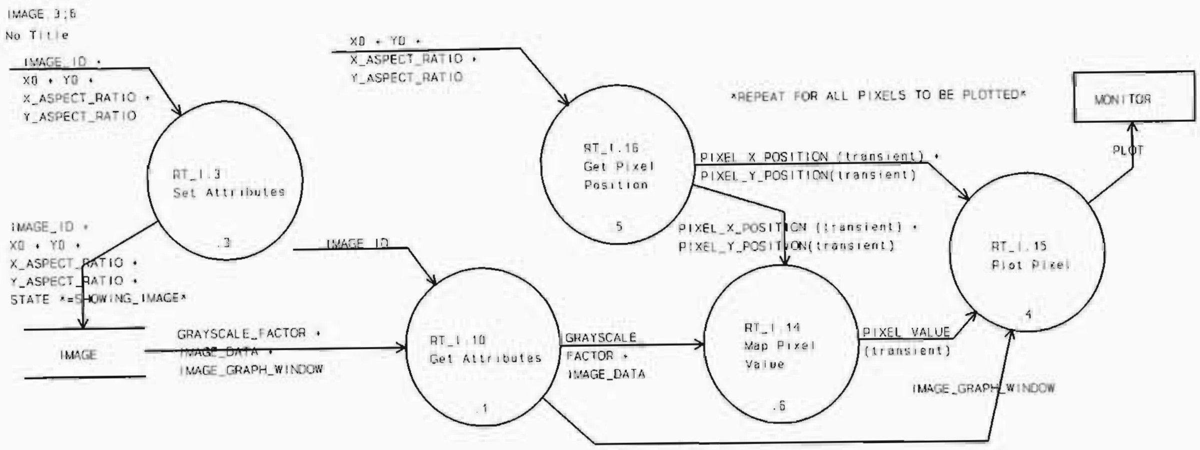


Figure D.33 Data Flow Diagram for Object Image in State Showing Image

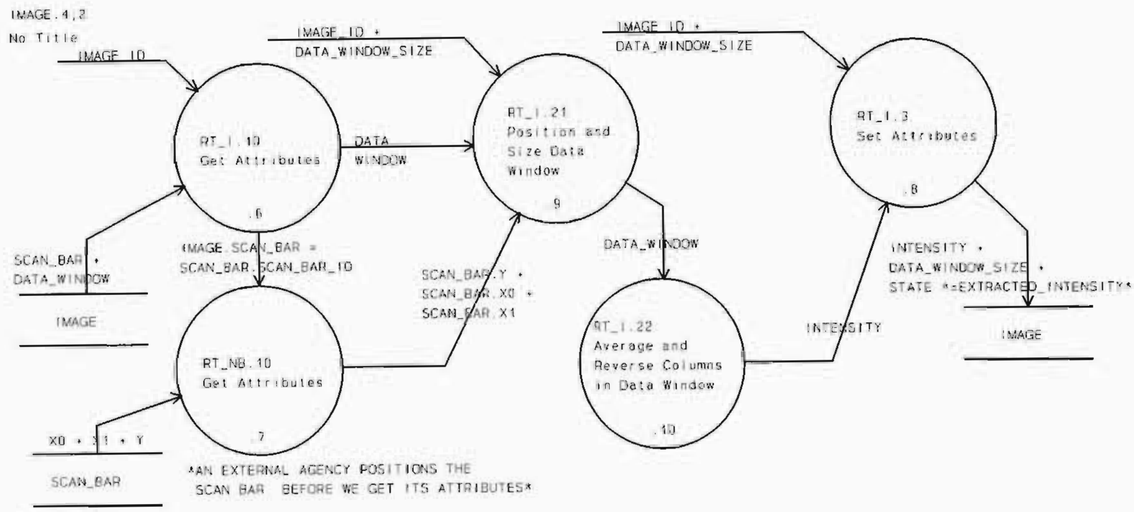


Figure D.34 Data Flow Diagram for Object Image in State Extracted Intensity

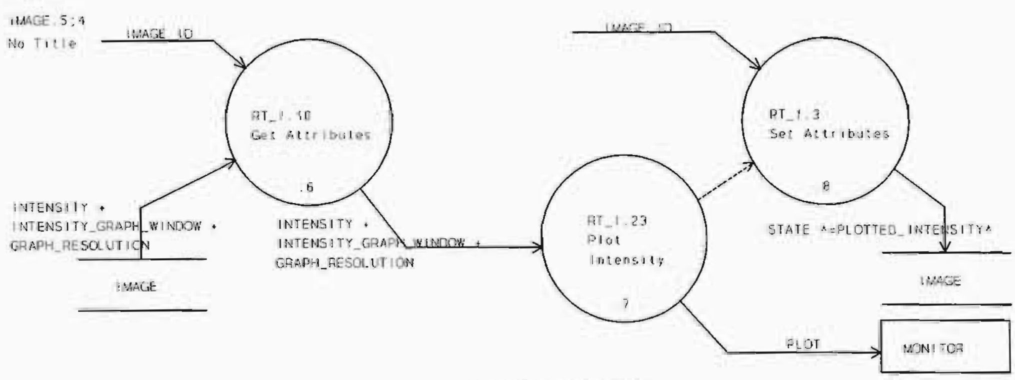


Figure D.35 Data Flow Diagram for Object Image in State Plotted Intensity

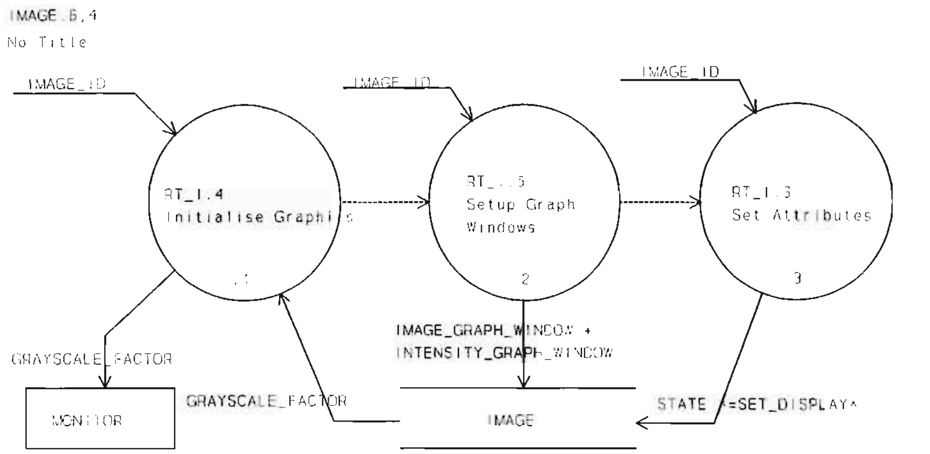


Figure D.36 Data Flow Diagram for Object Image in State Set Display

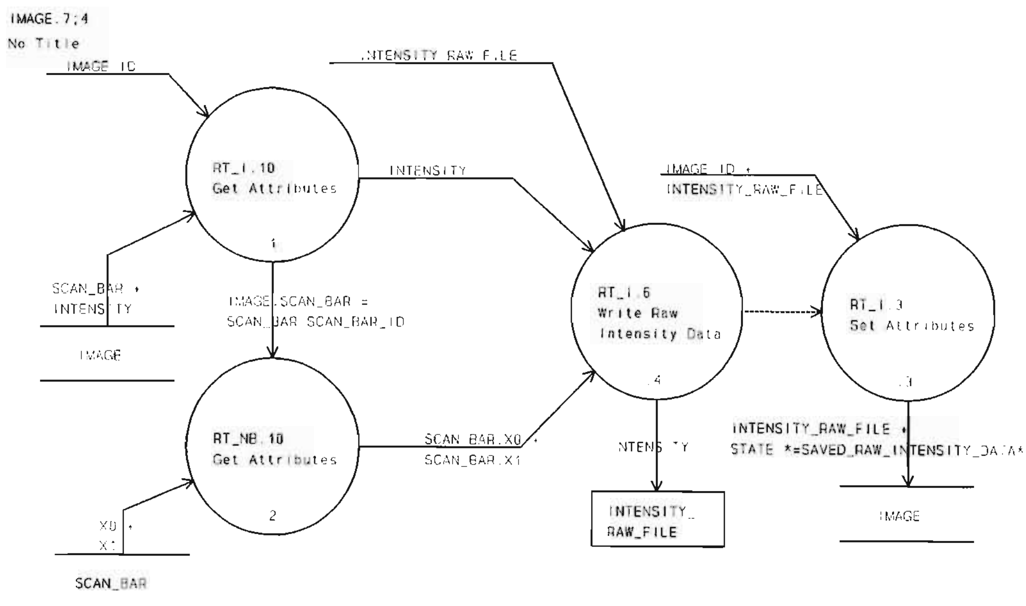


Figure D.37 Data Flow Diagram for Object Image in State Saved Raw Intensity Data

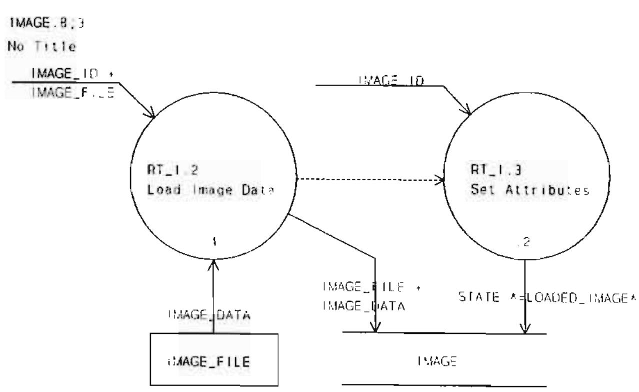


Figure D.38 Data Flow Diagram for Object Image in State Loaded Image

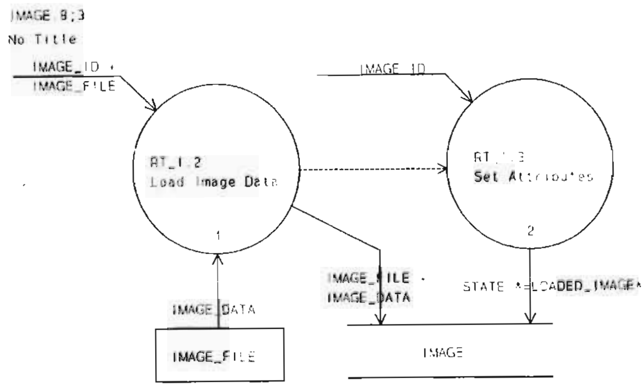


Figure D.38 Data Flow Diagram for Object Image in State Loaded Image

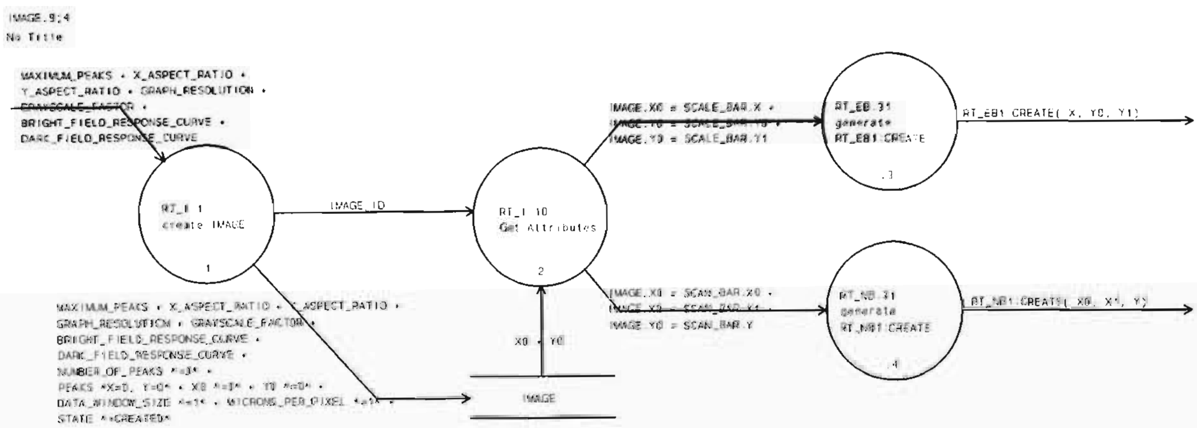


Figure D.39 Data Flow Diagram for Object Image in State Created

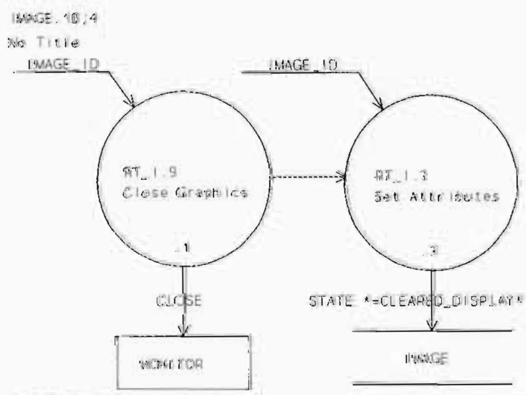


Figure D.40 Data Flow Diagram for Object Image in State Cleared Display

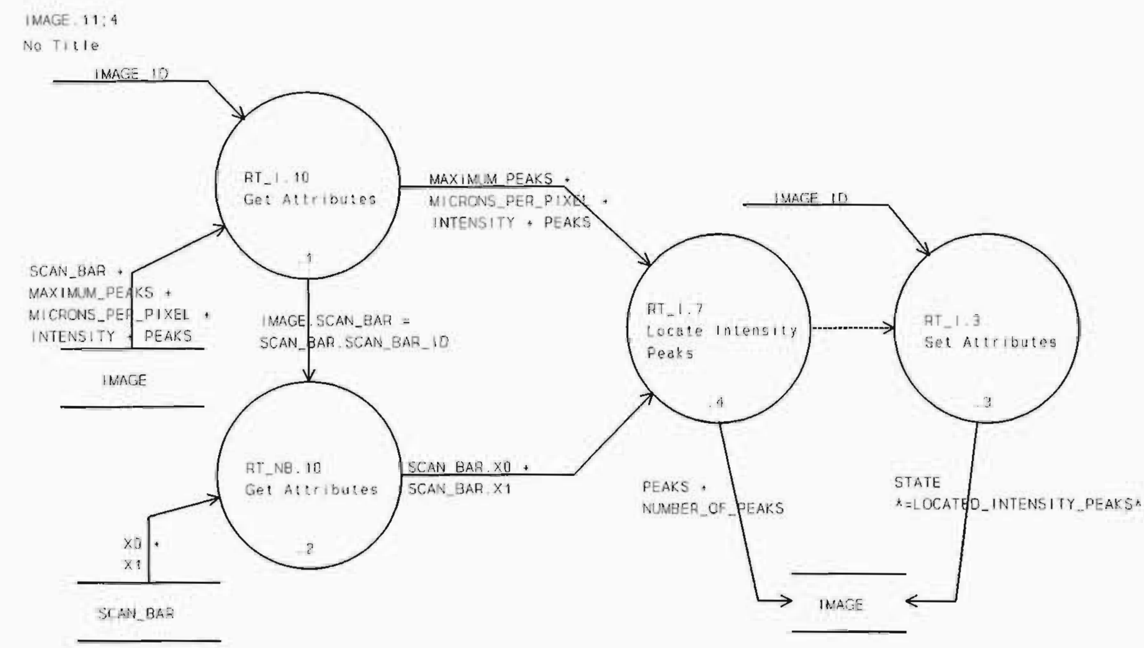


Figure D.41 Data Flow Diagram for Object Image in State Located Intensity Peaks

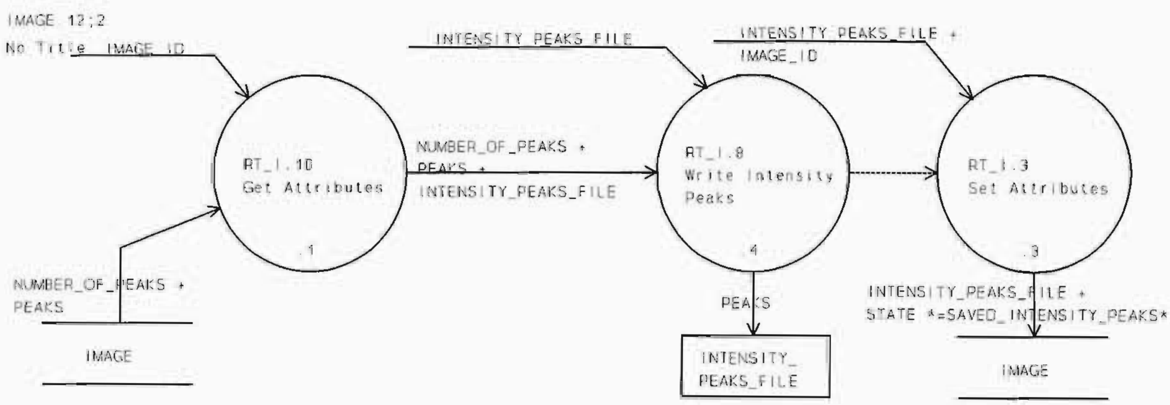


Figure D.42 Data Flow Diagram for Object Image in State Saved Intensity Peaks

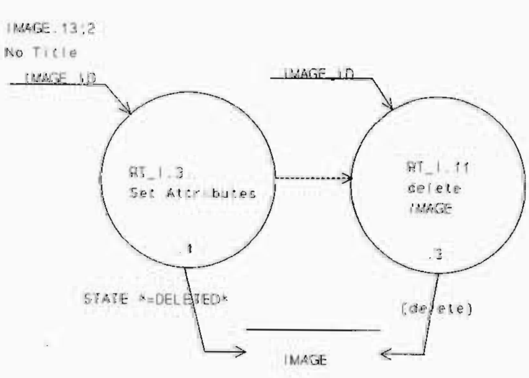


Figure D.43 Data Flow Diagram for Object Image in State Deleted

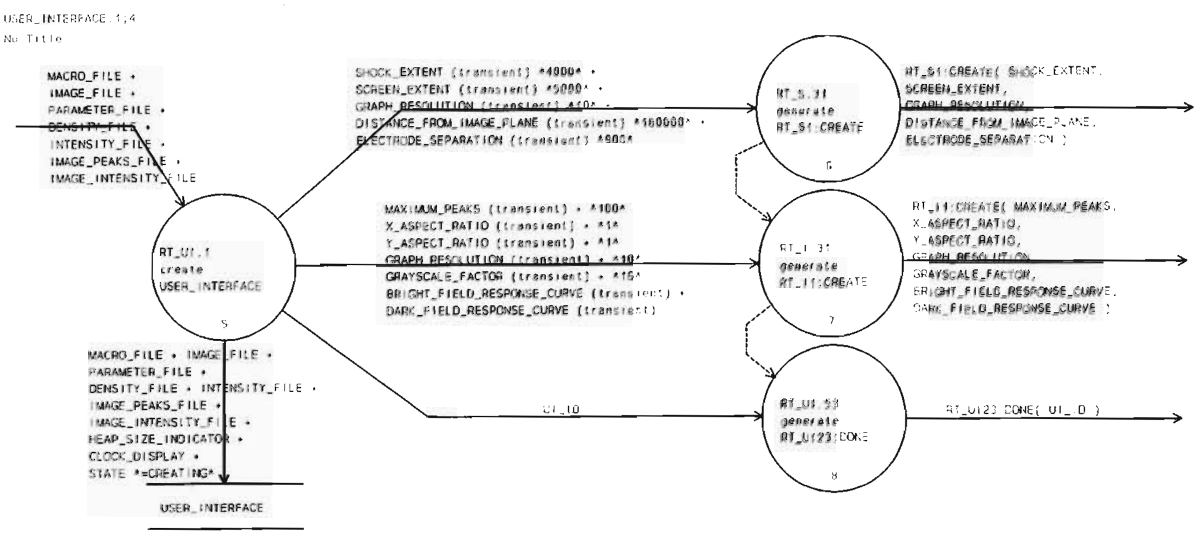


Figure D.44 Data Flow Diagram for Object User Interface in State Creating

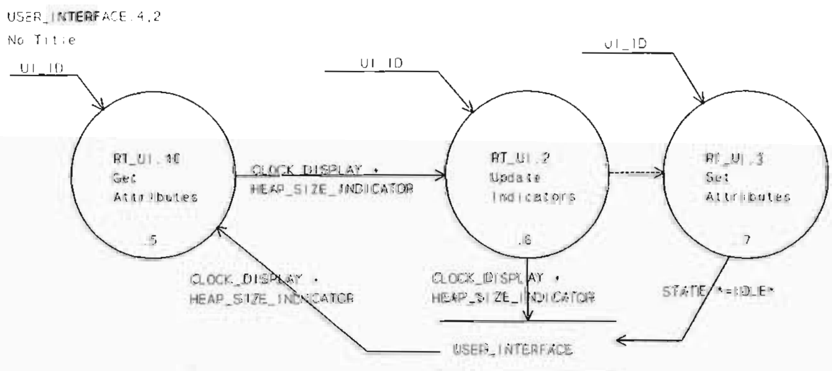


Figure D.45 Data Flow Diagram for Object User Interface in State Idle

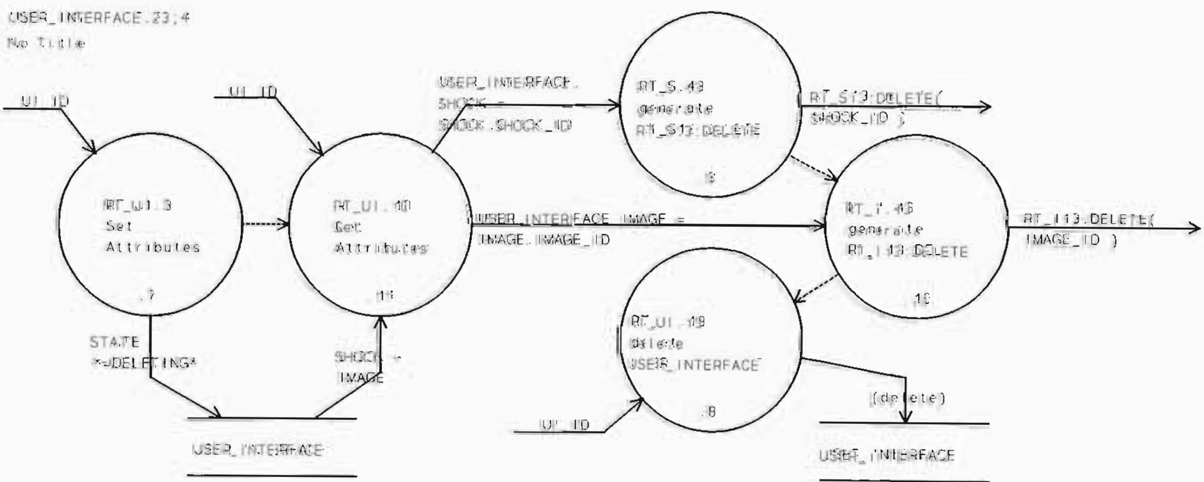


Figure D.46 Data Flow Diagram for Object User Interface in State Deleting

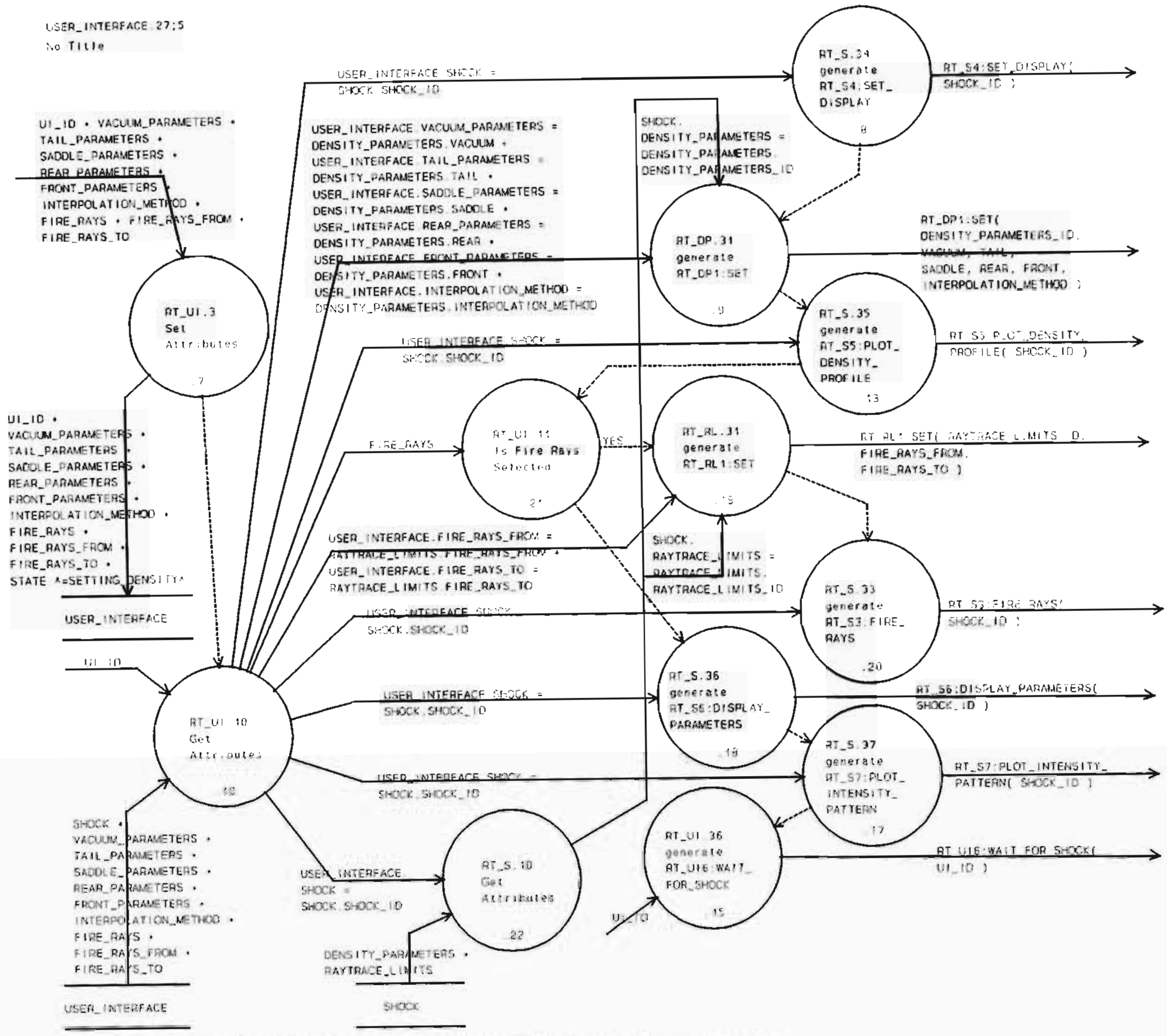


Figure D.47 Data Flow Diagram for Object User Interface in State Setting Density

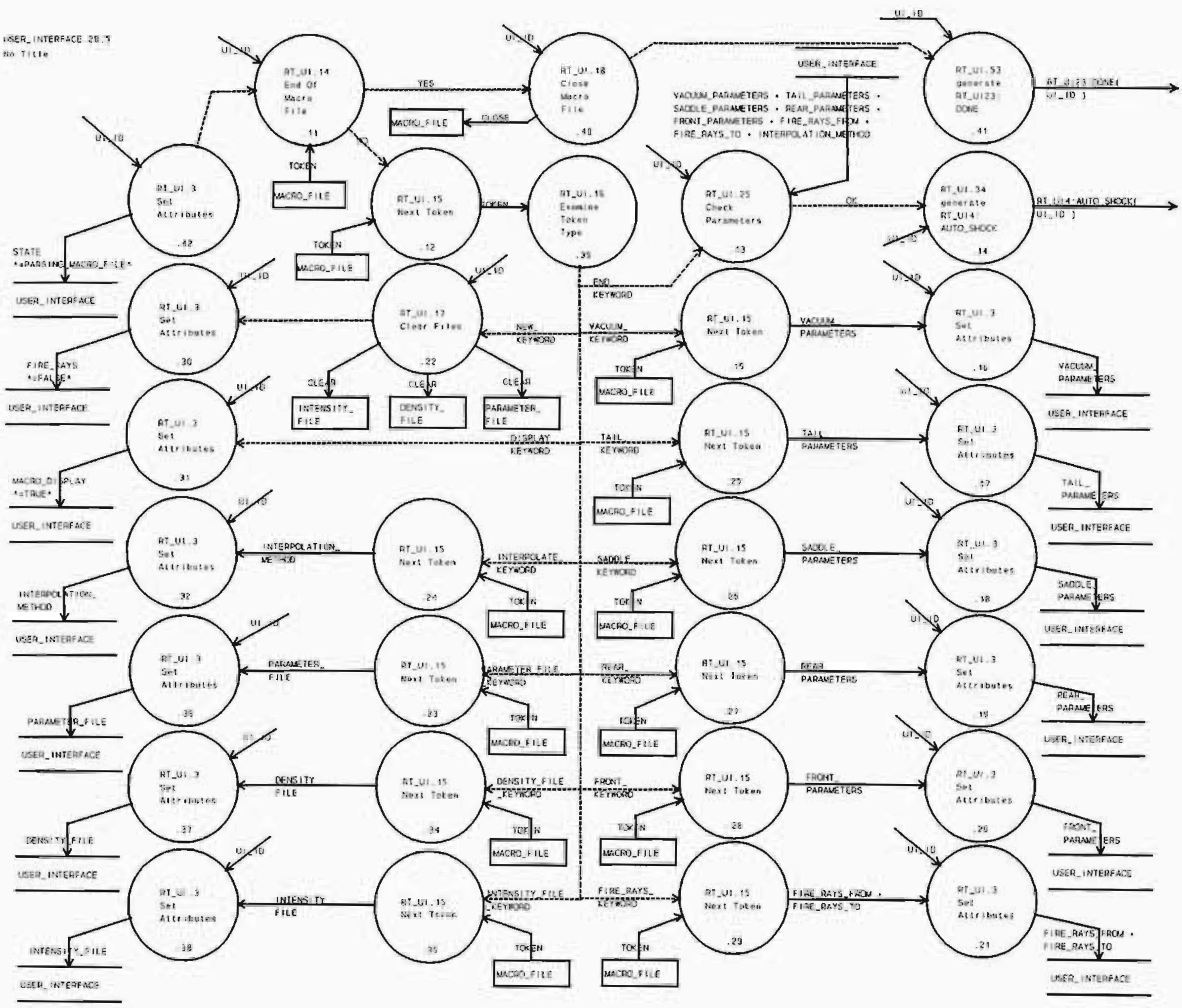


Figure D.48 Data Flow Diagram for Object User Interface in State Parsing Macro File

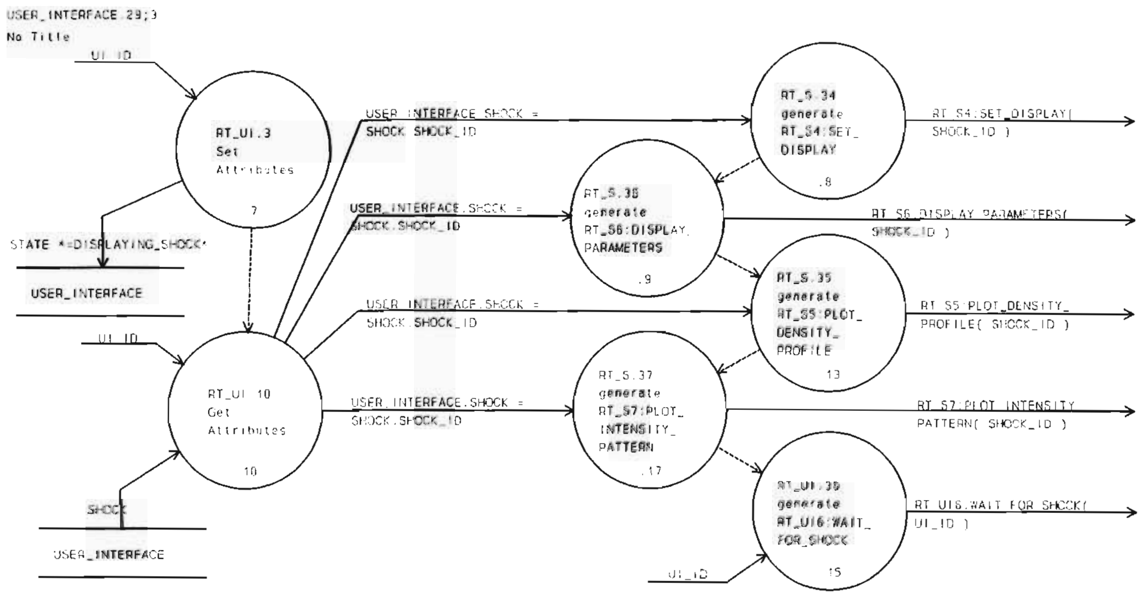


Figure D.49 Data Flow Diagram for Object User Interface in State Displaying Shock

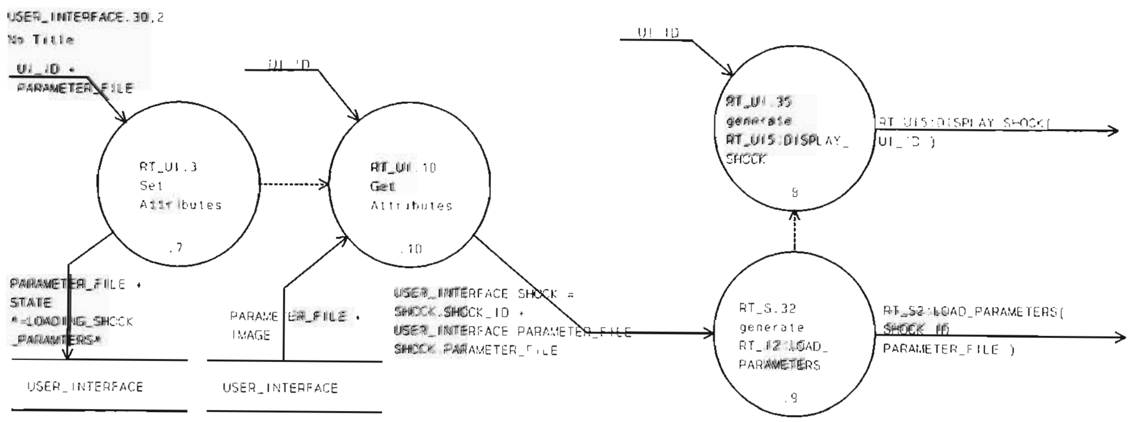


Figure D.50 Data Flow Diagram for Object User Interface in State Loading Shock Parameters

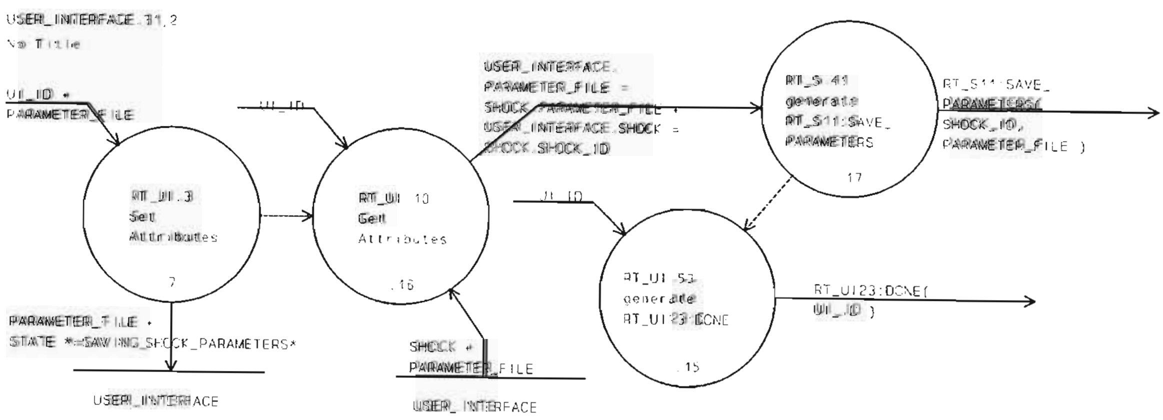


Figure D.51 Data Flow Diagram for Object User Interface in State Saving Shock Parameters

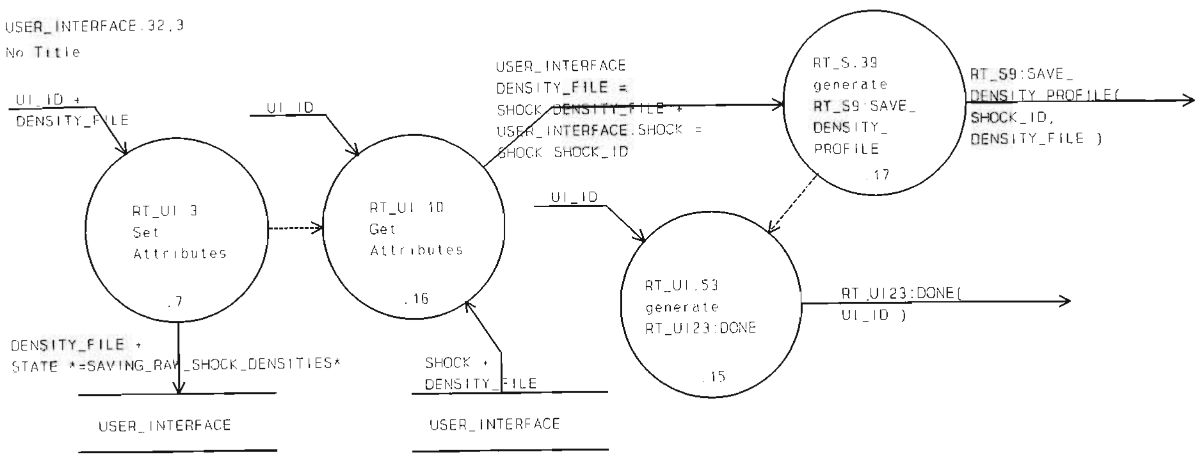


Figure D.52 Data Flow Diagram for Object User Interface in State Saving Raw Shock Densities

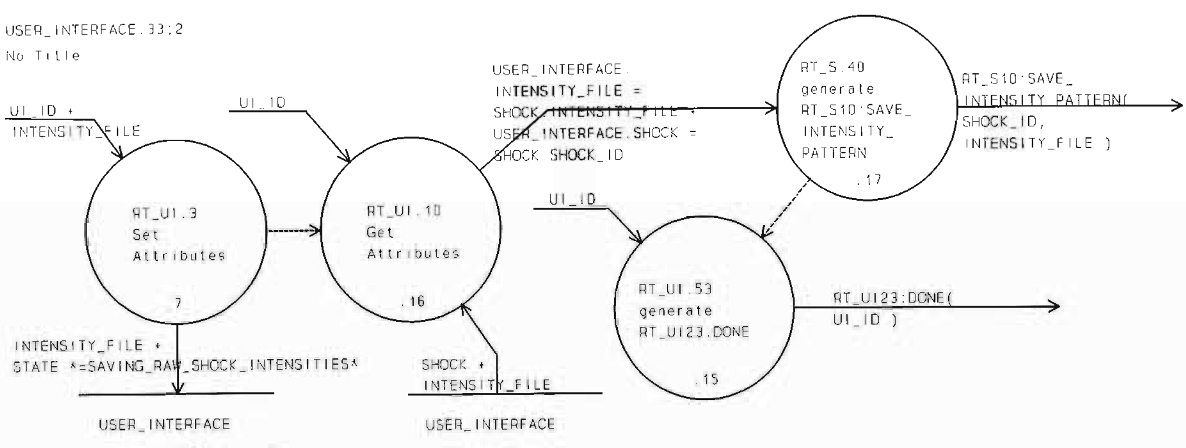


Figure D.53 Data Flow Diagram for Object User Interface in State Saving Raw Shock Intensities

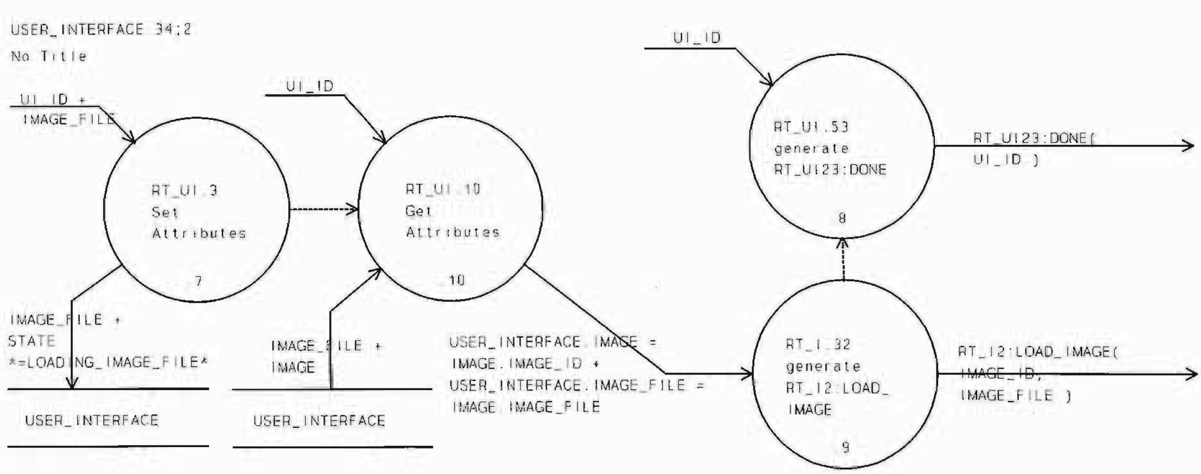


Figure D.54 Data Flow Diagram for Object User Interface in State Loading Image File

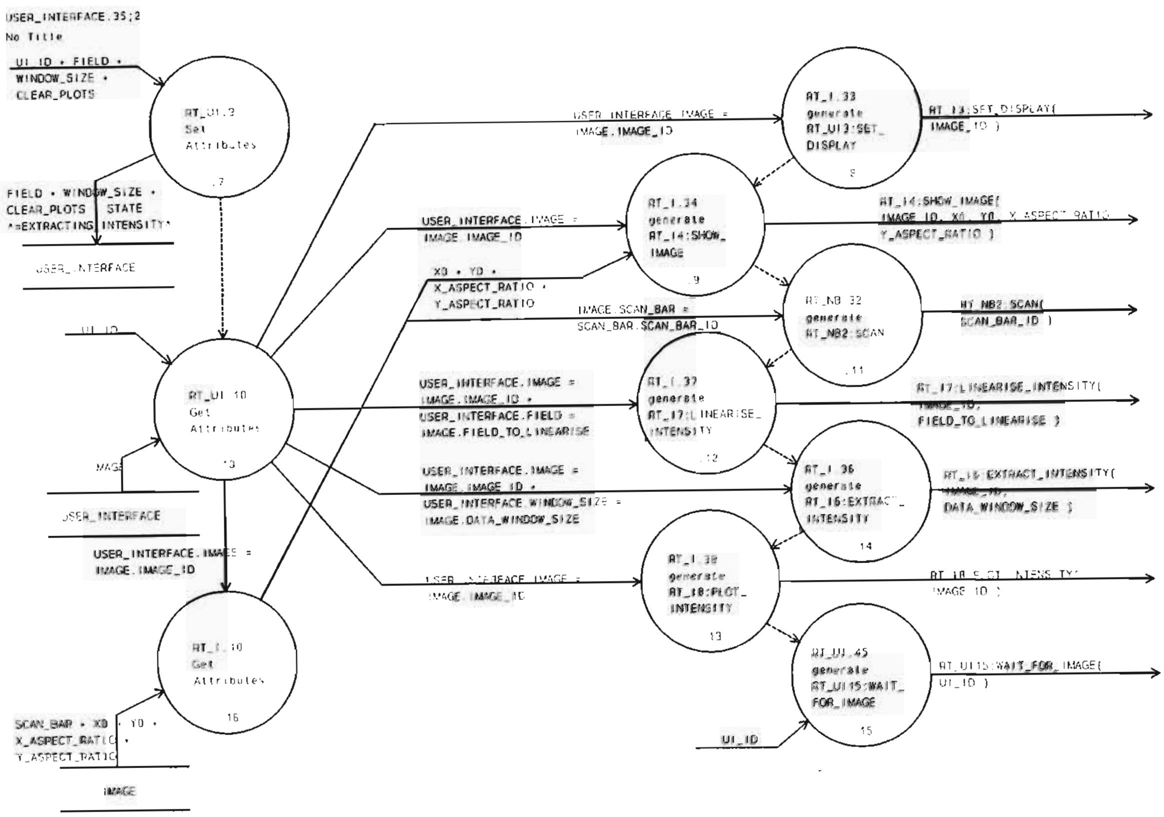


Figure D.55 Data Flow Diagram for Object User Interface in State Extracting Intensity

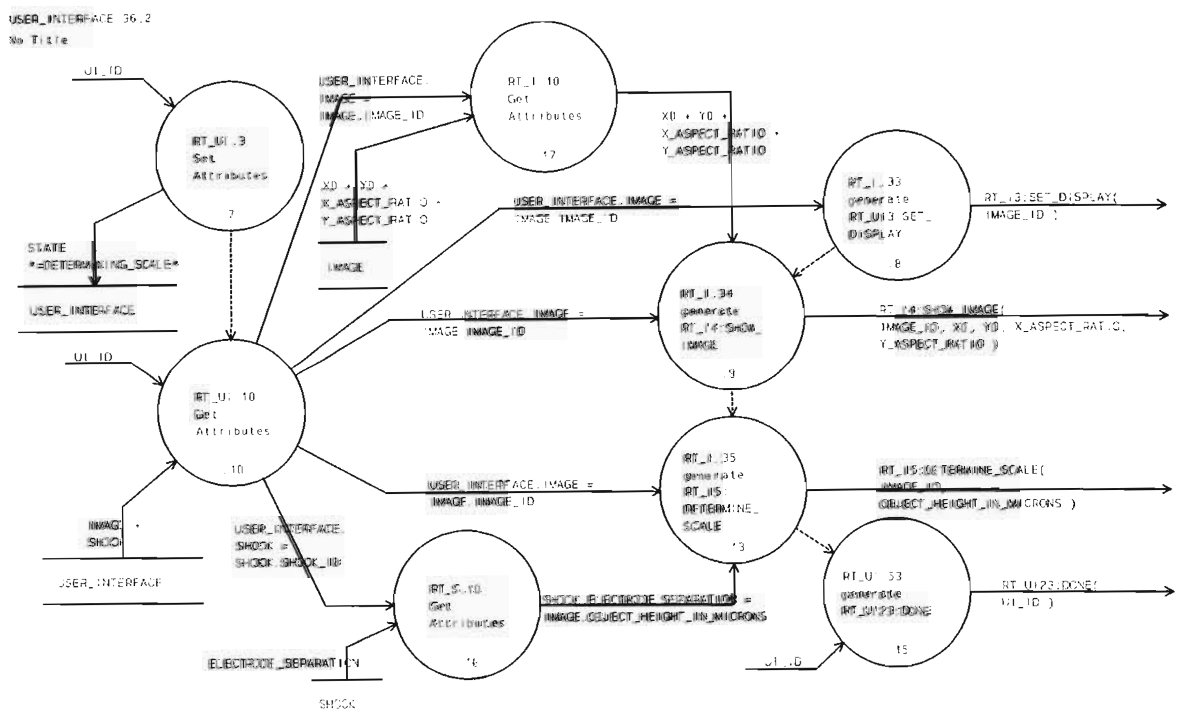


Figure D.56 Data Flow Diagram for Object User Interface in State Determining Image Scale

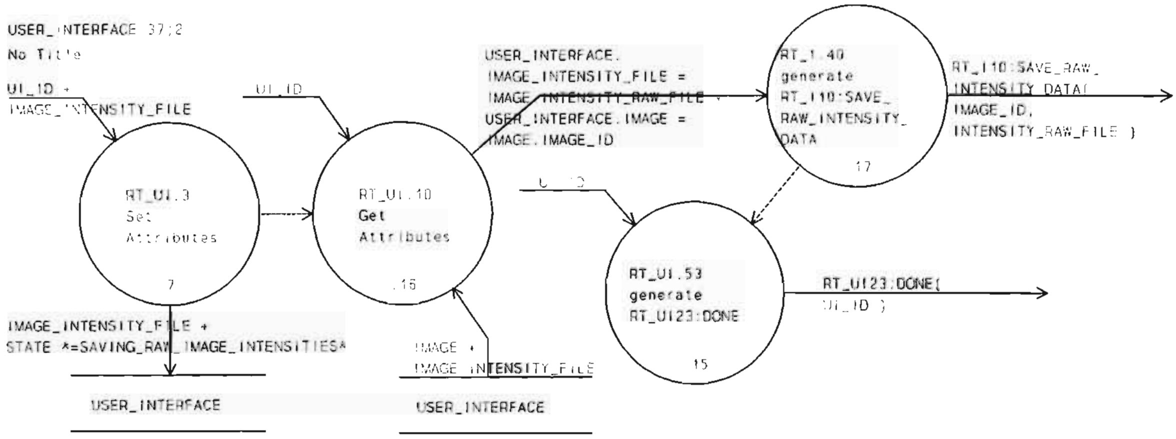


Figure D.57 Data Flow Diagram for Object User Interface in State Saving Raw Image Intensities

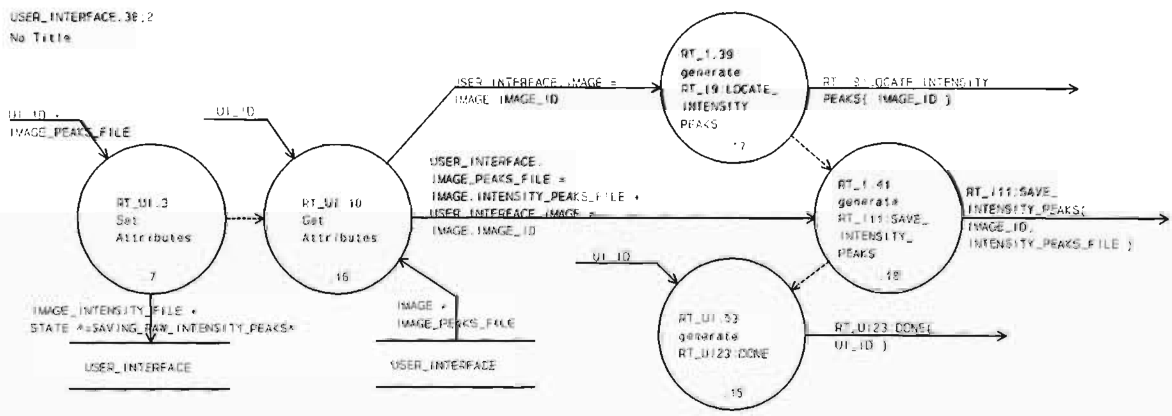


Figure D.58 Data Flow Diagram for Object User Interface in State Saving Raw Intensity Peaks

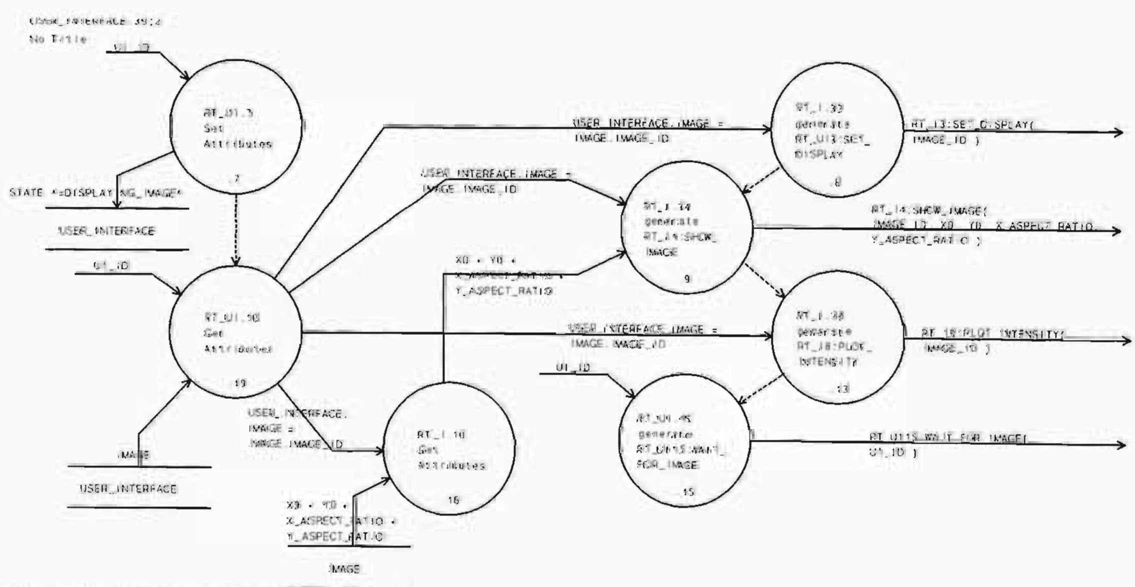


Figure D.59 Data Flow Diagram for Object User Interface in State Displaying Image

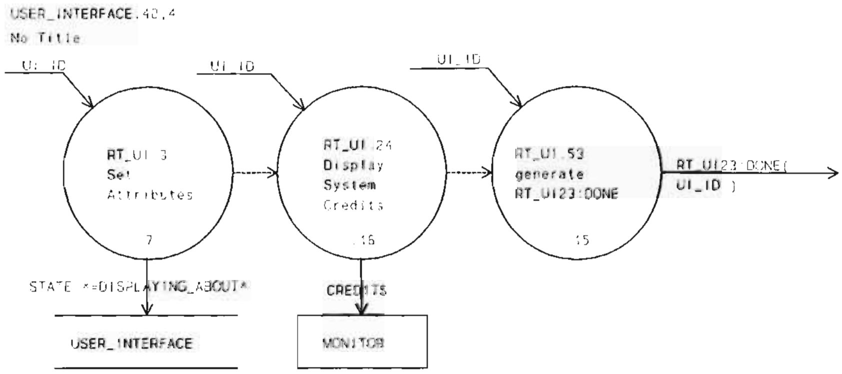


Figure D.60 Data Flow Diagram for Object User Interface in State Displaying About

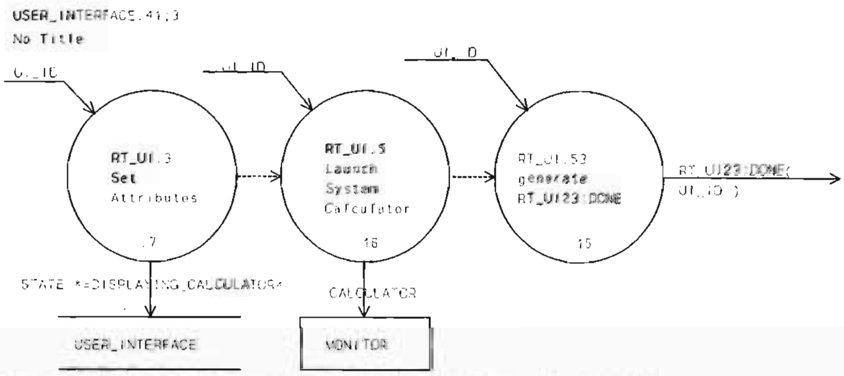


Figure D.61 Data Flow Diagram for Object User Interface in State Displaying Calculator

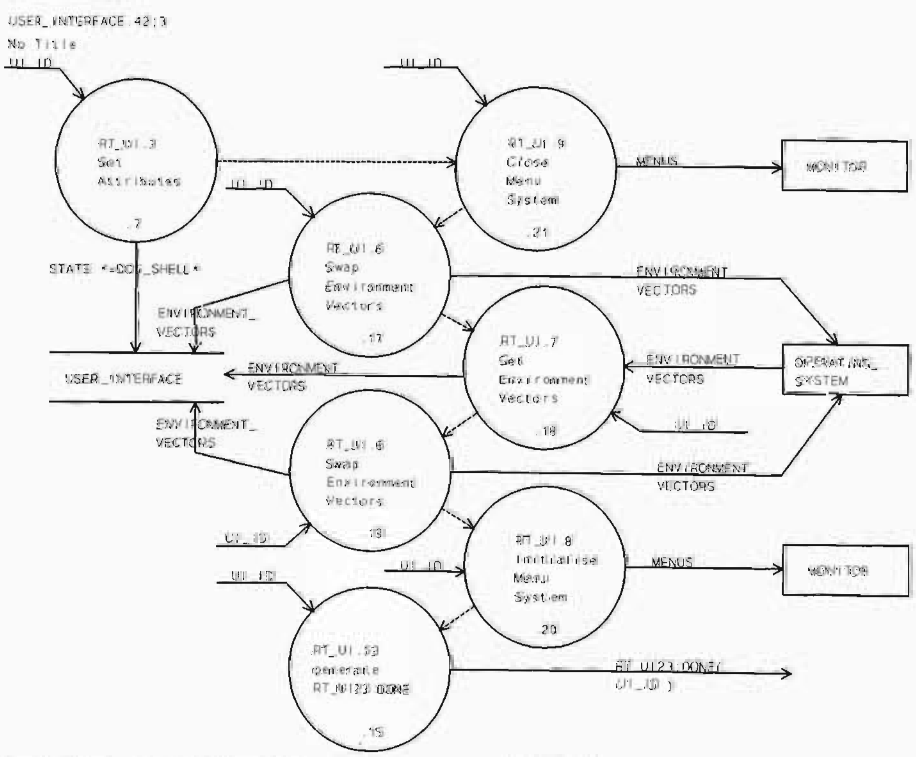


Figure D.62 Data Flow Diagram for Object User Interface in State DOS Shell

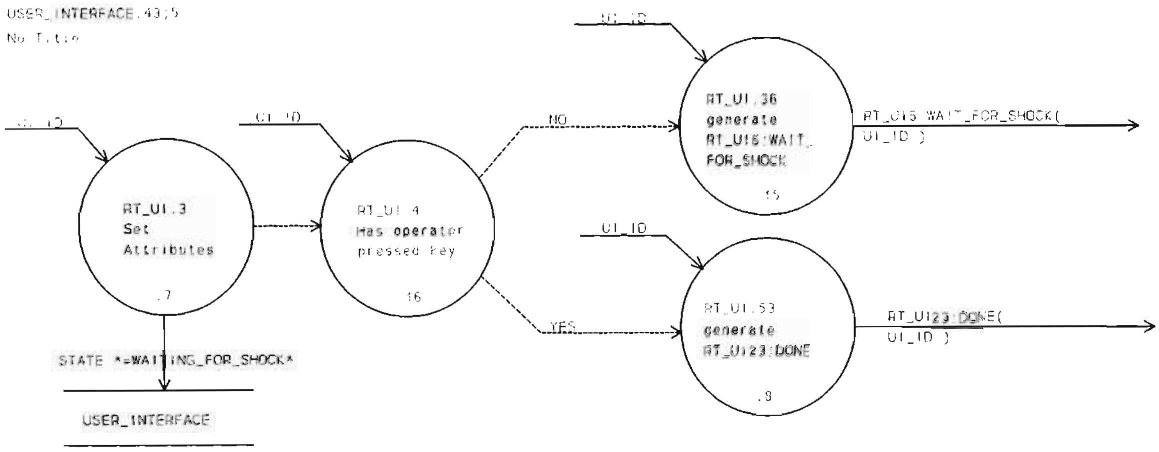


Figure D.63 Data Flow Diagram for Object User Interface in State Waiting For Shock

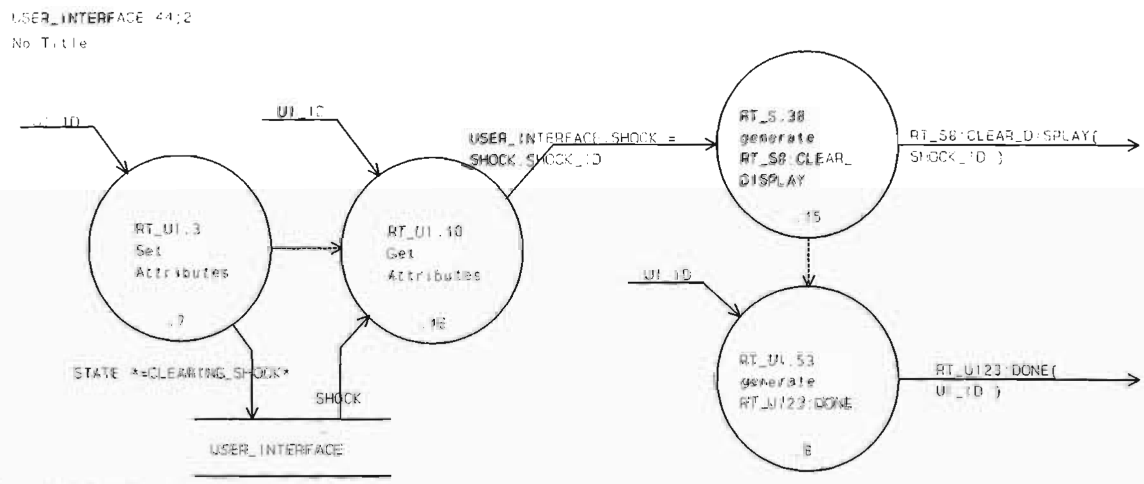


Figure D.64 Data Flow Diagram for Object User Interface in State Clearing Shock

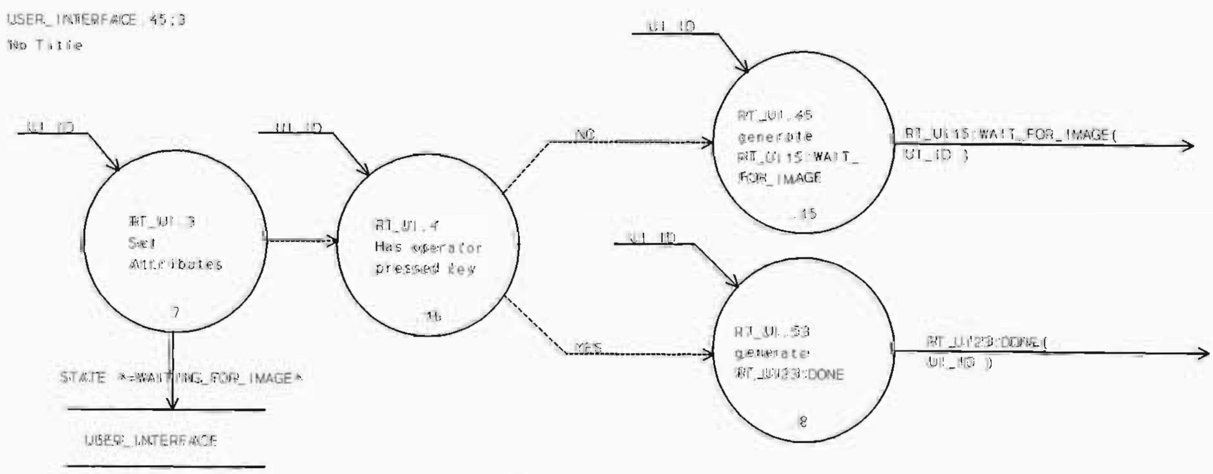


Figure D.65 Data Flow Diagram for Object User Interface in State Waiting For Image

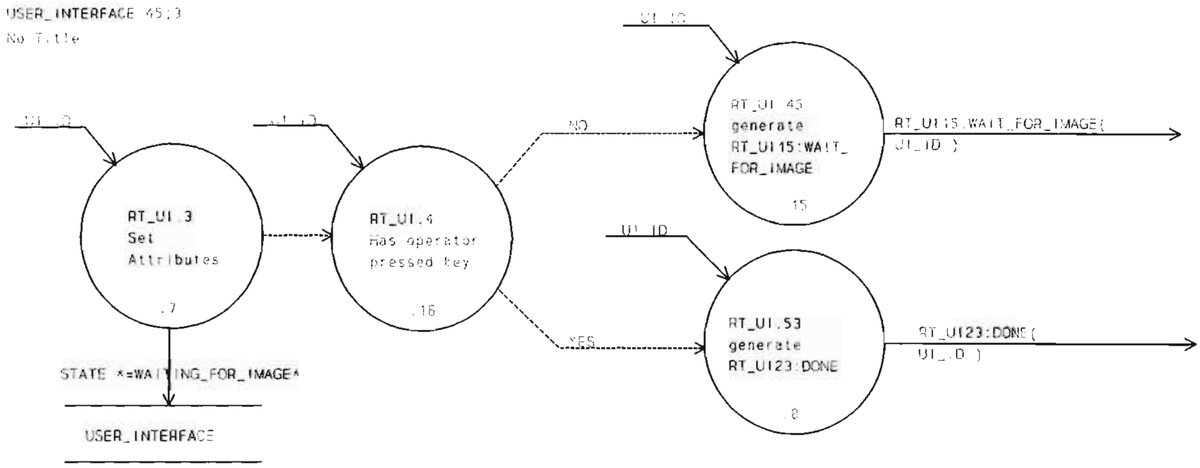


Figure D.65 Data Flow Diagram for Object User Interface in State Waiting For Image

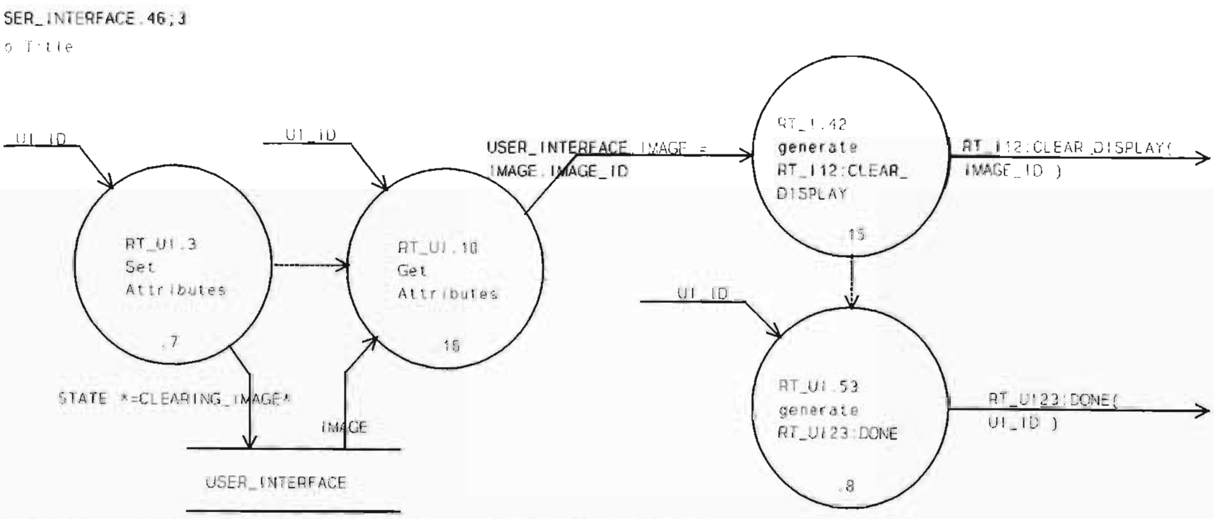


Figure D.66 Data Flow Diagram for Object User Interface in State Clearing Image

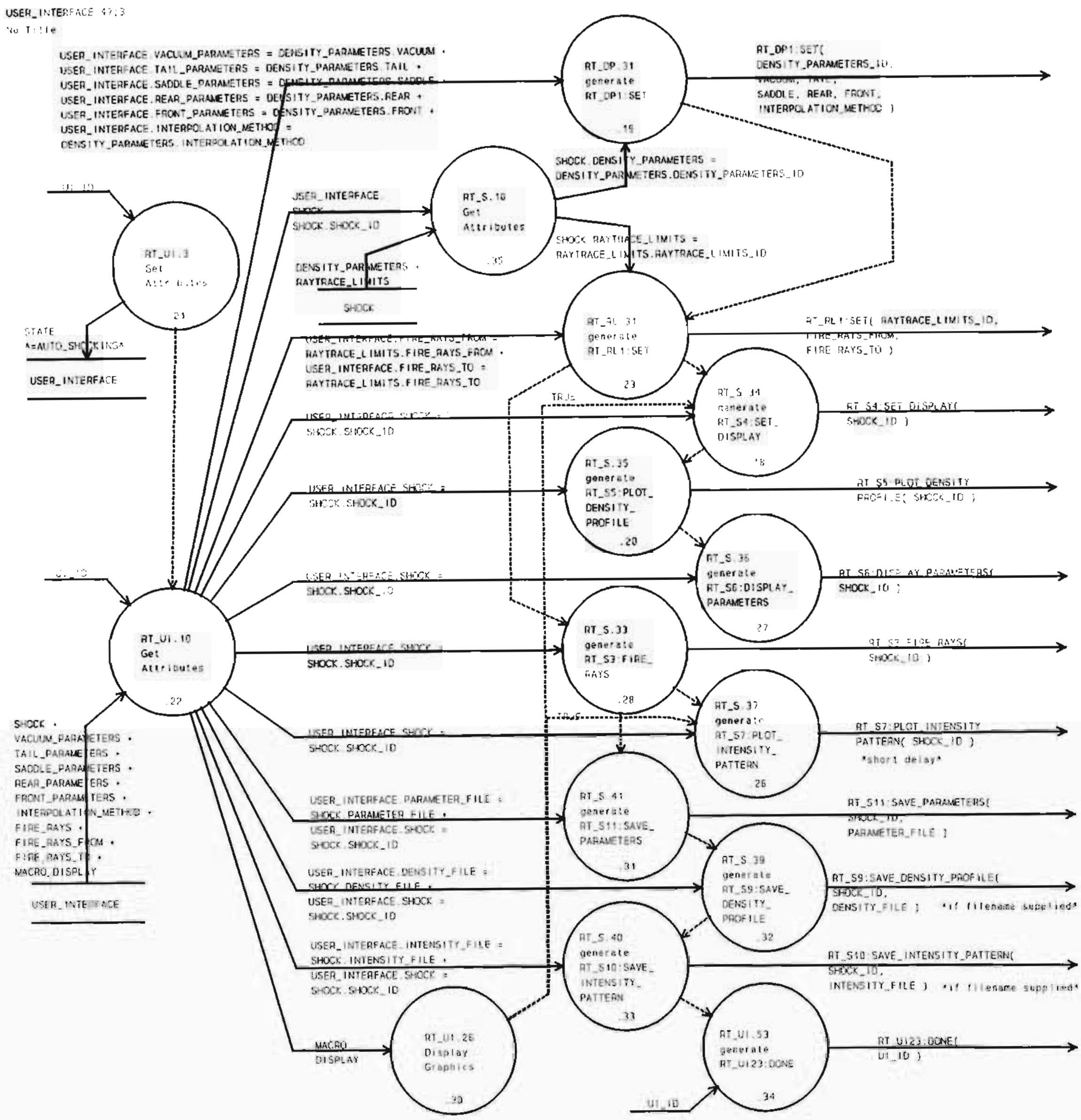


Figure D.67 Data Flow Diagram for Object User Interface in State Auto Shocking

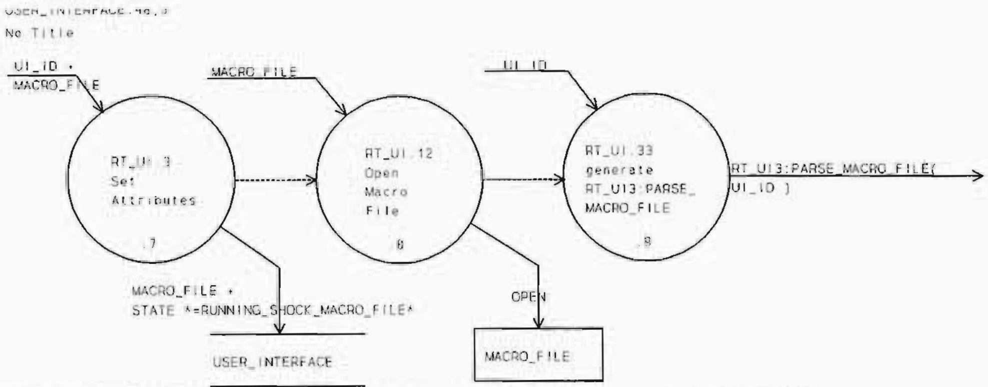


Figure D.68 Data Flow Diagram for Object User Interface in State Running Shock Macro File

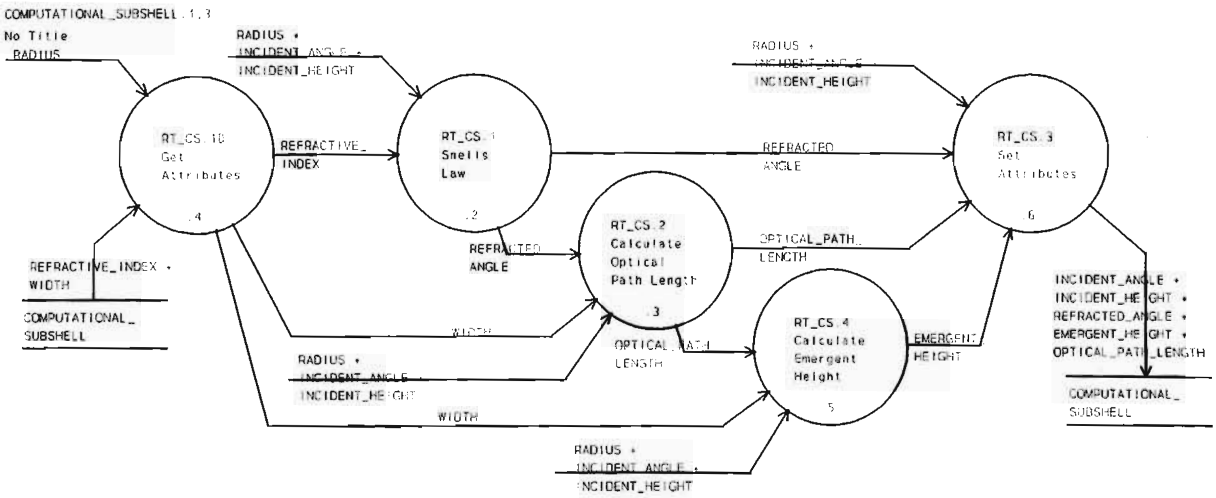


Figure D.69 Data Flow Diagram for Object Computational Subshell in State Temporal

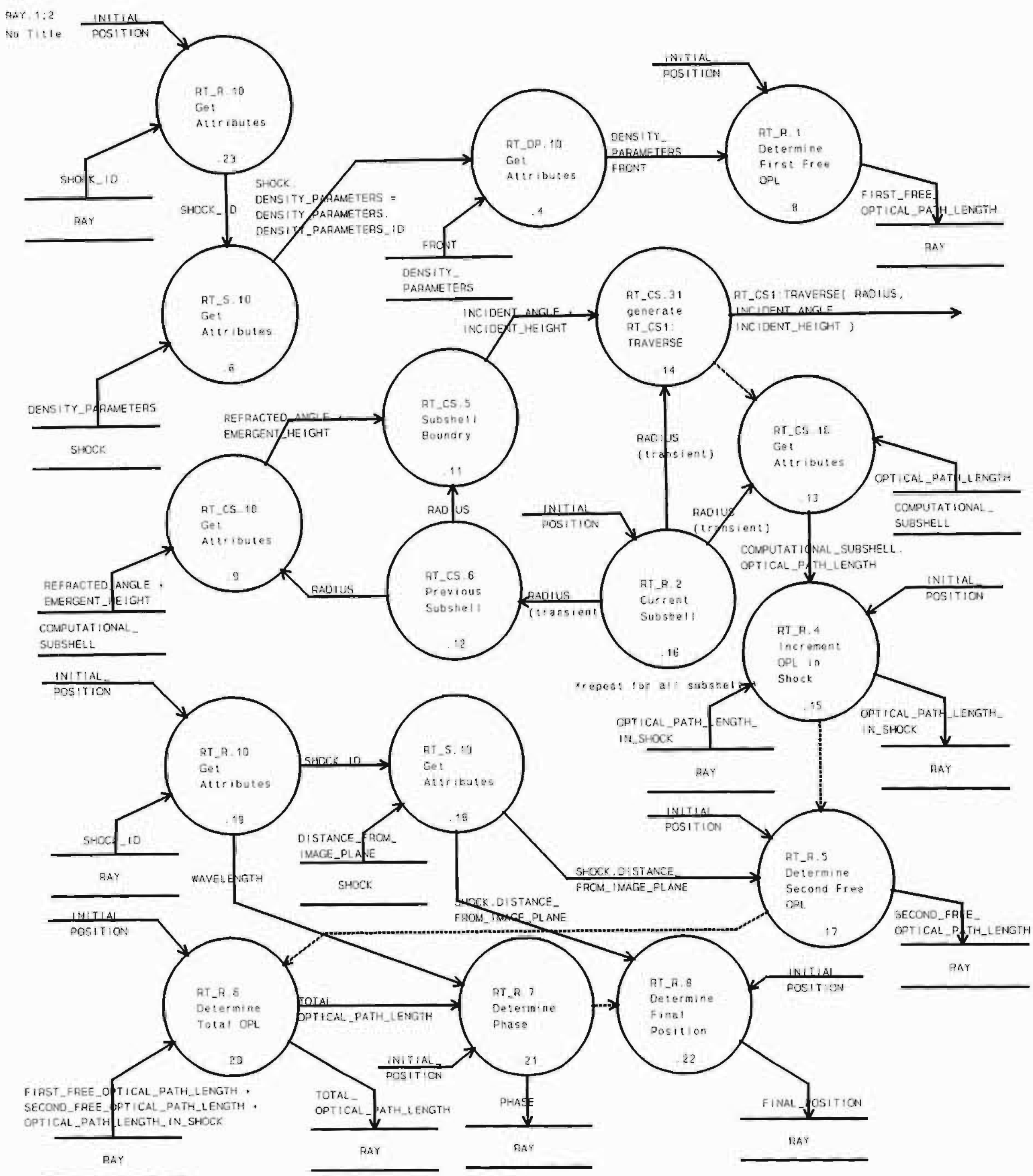


Figure D.70 Data Flow Diagram for Object Ray in State Temporal

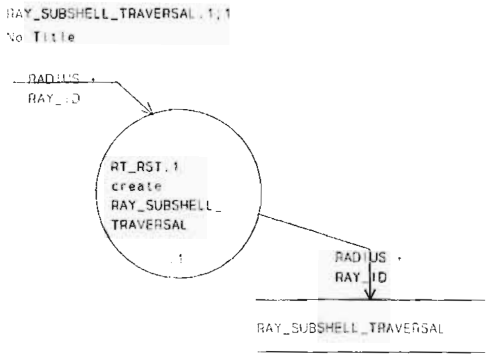


Figure D.71 Data Flow Diagram for Object Ray Subshell Traversal in State Temporal

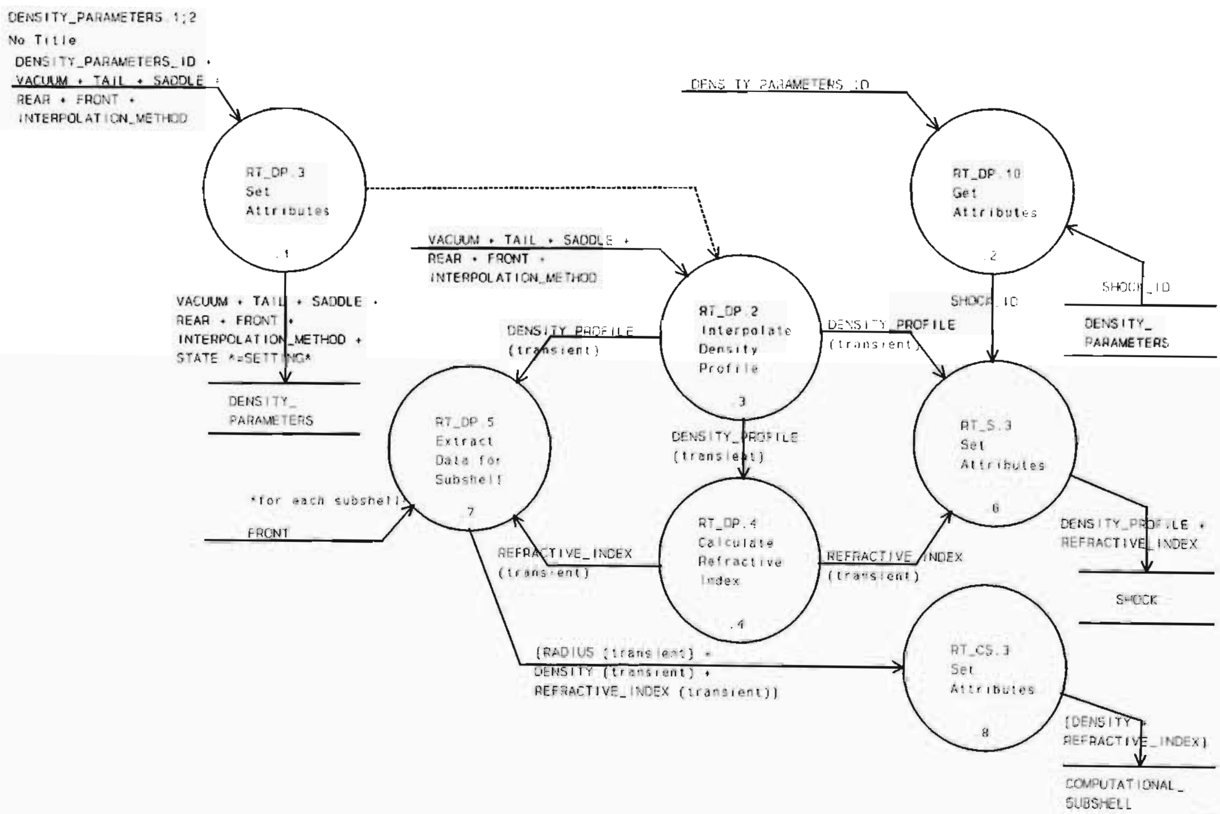


Figure D.72 Data Flow Diagram for Object Density Parameters in State Setting

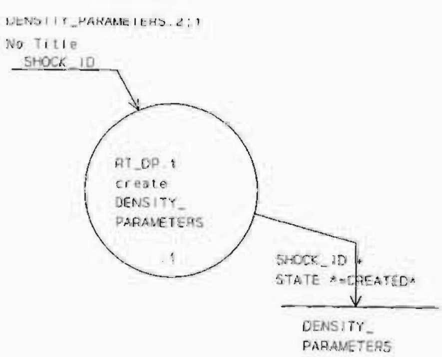


Figure D.73 Data Flow Diagram for Object Density Parameters in State Creating

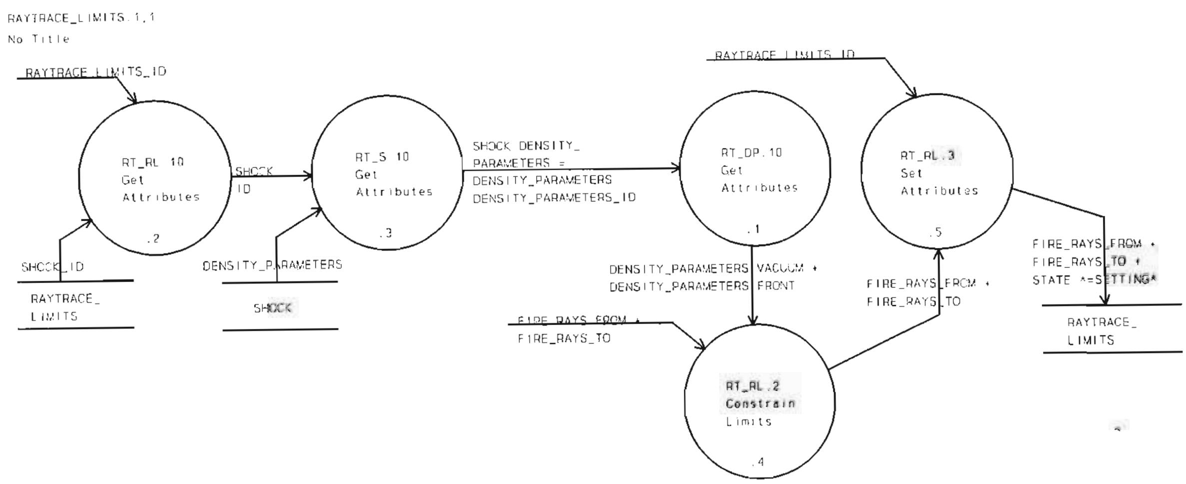


Figure D.74 Data Flow Diagram for Object Raytrace Limits in State Setting

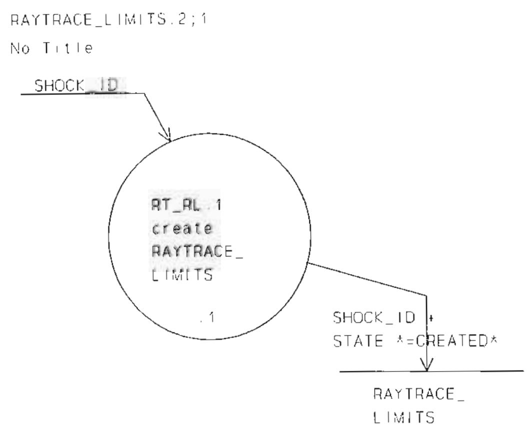


Figure D.75 Data Flow Diagram for Object Raytrace Limits in State Creating

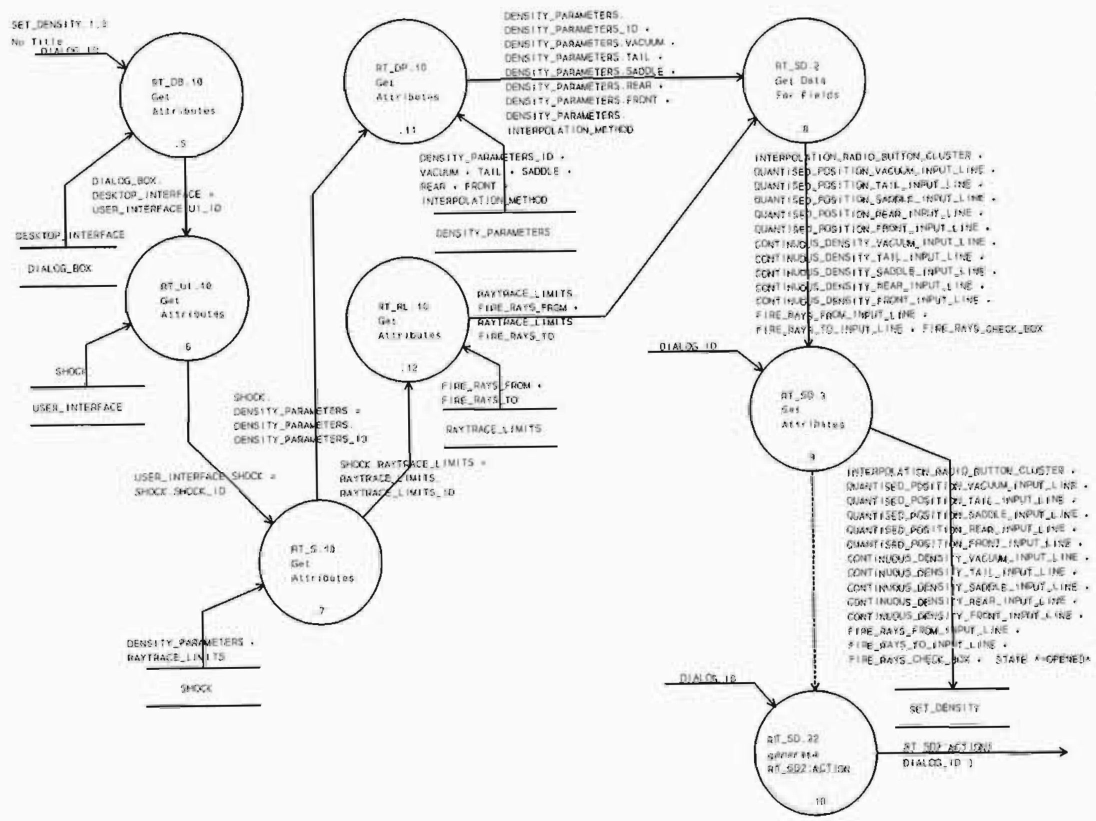


Figure D.76 Data Flow Diagram for Object Set Density in State Opened

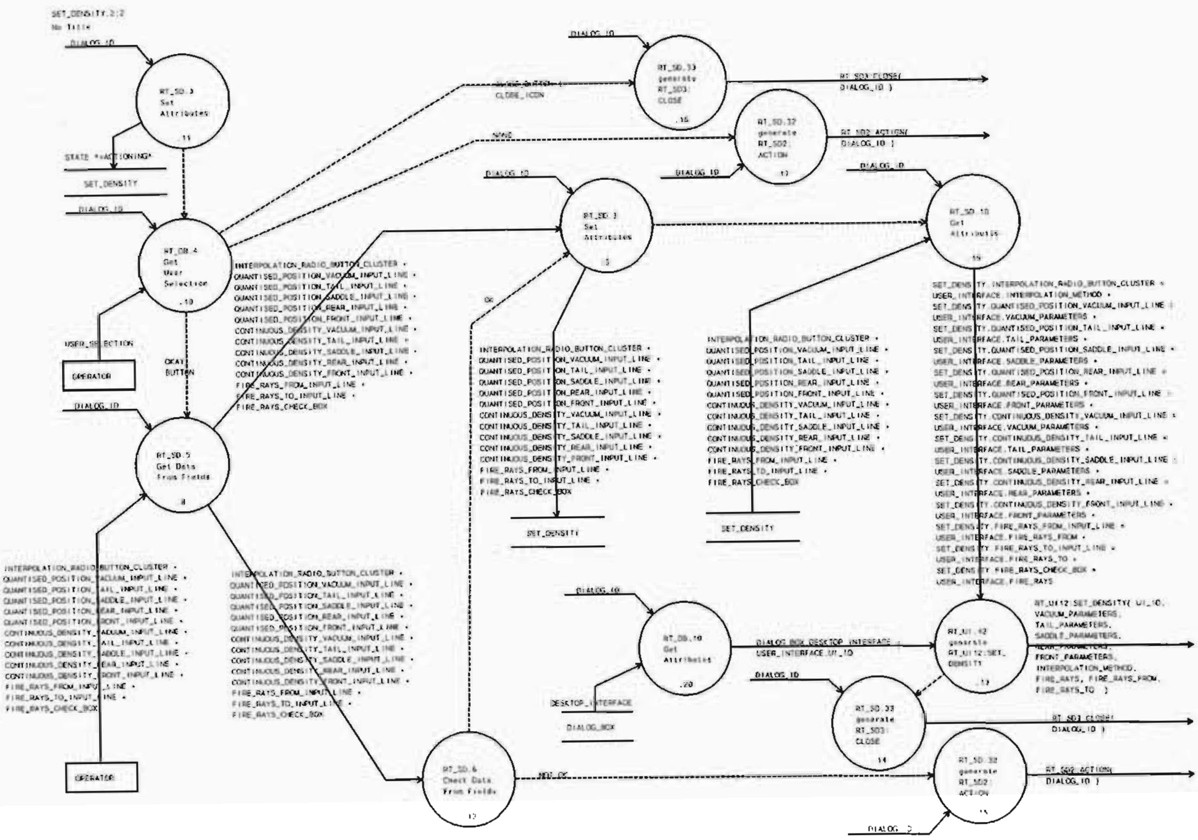


Figure D.77 Data Flow Diagram for Object Set Density in State Actioning

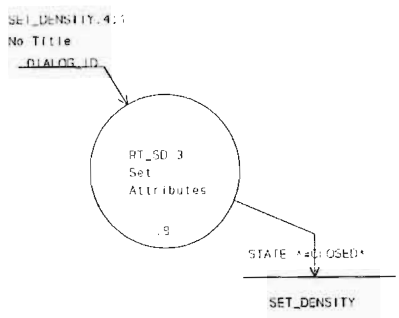


Figure D.78 Data Flow Diagram for Object Set Density in State Closed

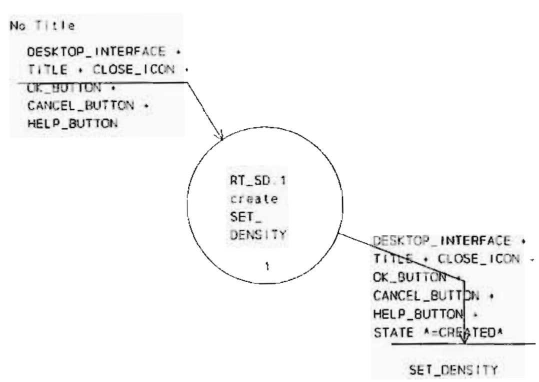


Figure D.79 Data Flow Diagram for Object Set Density in State Creating

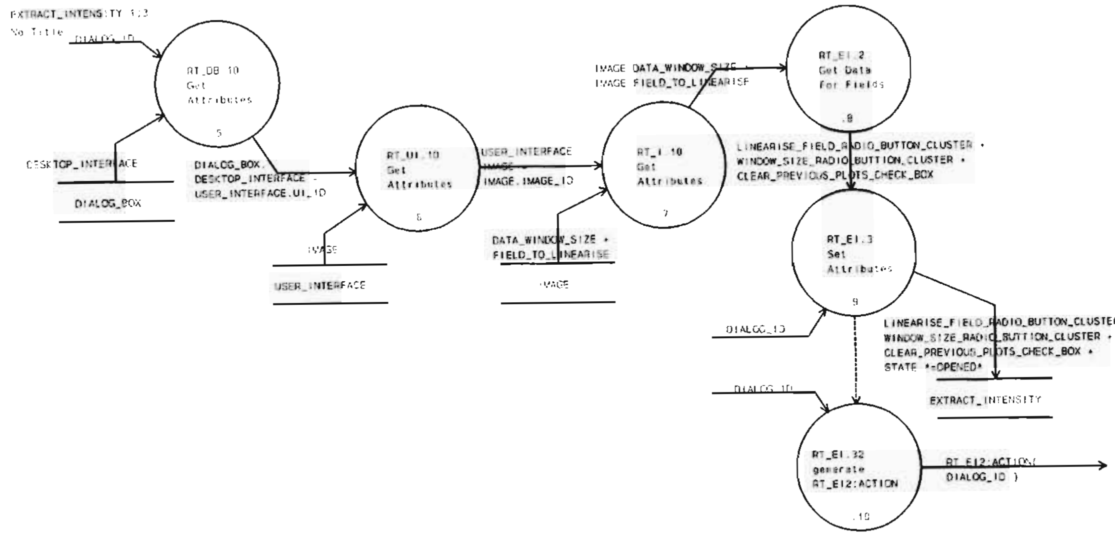


Figure D.80 Data Flow Diagram for Object Extract Intensity in State Opened

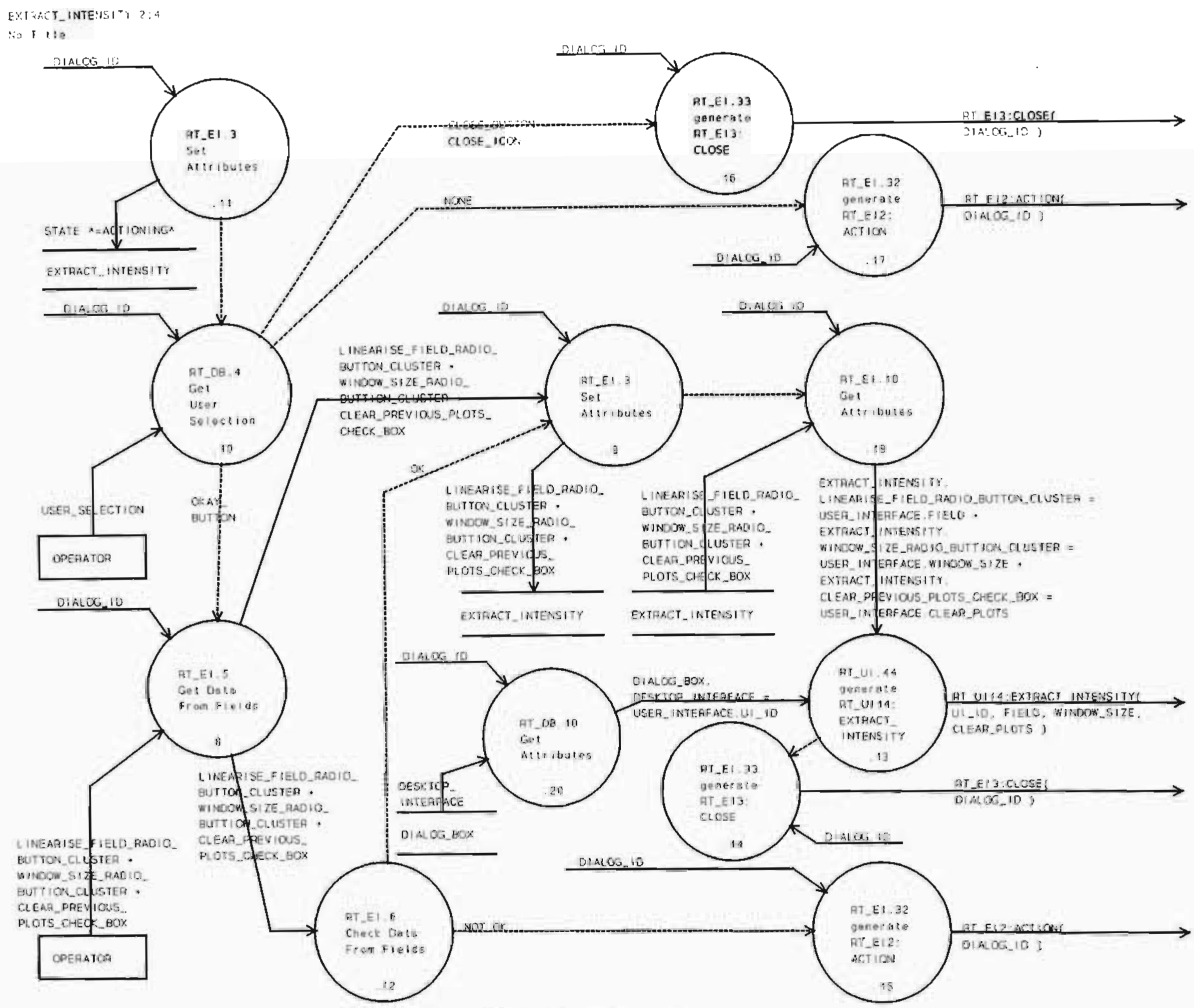


Figure D.81 Data Flow Diagram for Object Extract Intensity in State Actioning

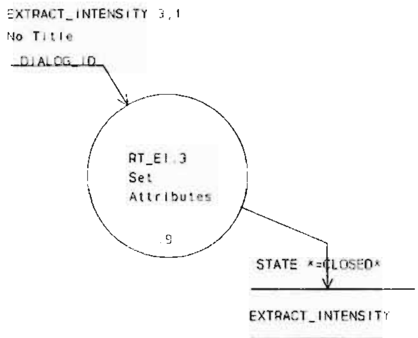


Figure D.82 Data Flow Diagram for Object Extract Intensity in State Closed

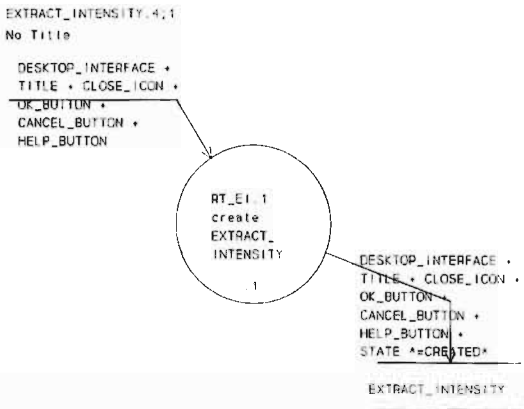


Figure D.83 Data Flow Diagram for Object Extract Intensity in State Creating

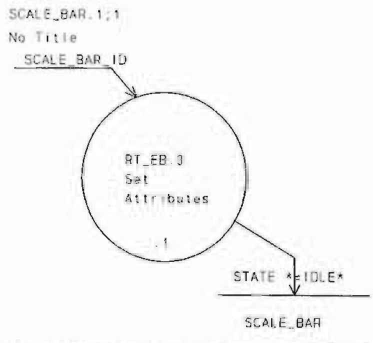


Figure D.84 Data Flow Diagram for Object Scale Bar in State Idle

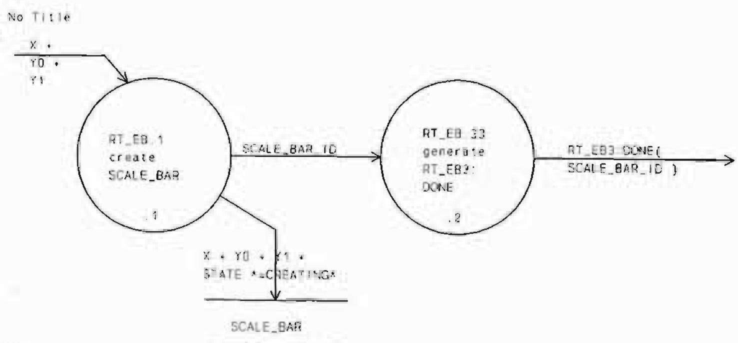


Figure D.85 Data Flow Diagram for Object Scale Bar in State Creating

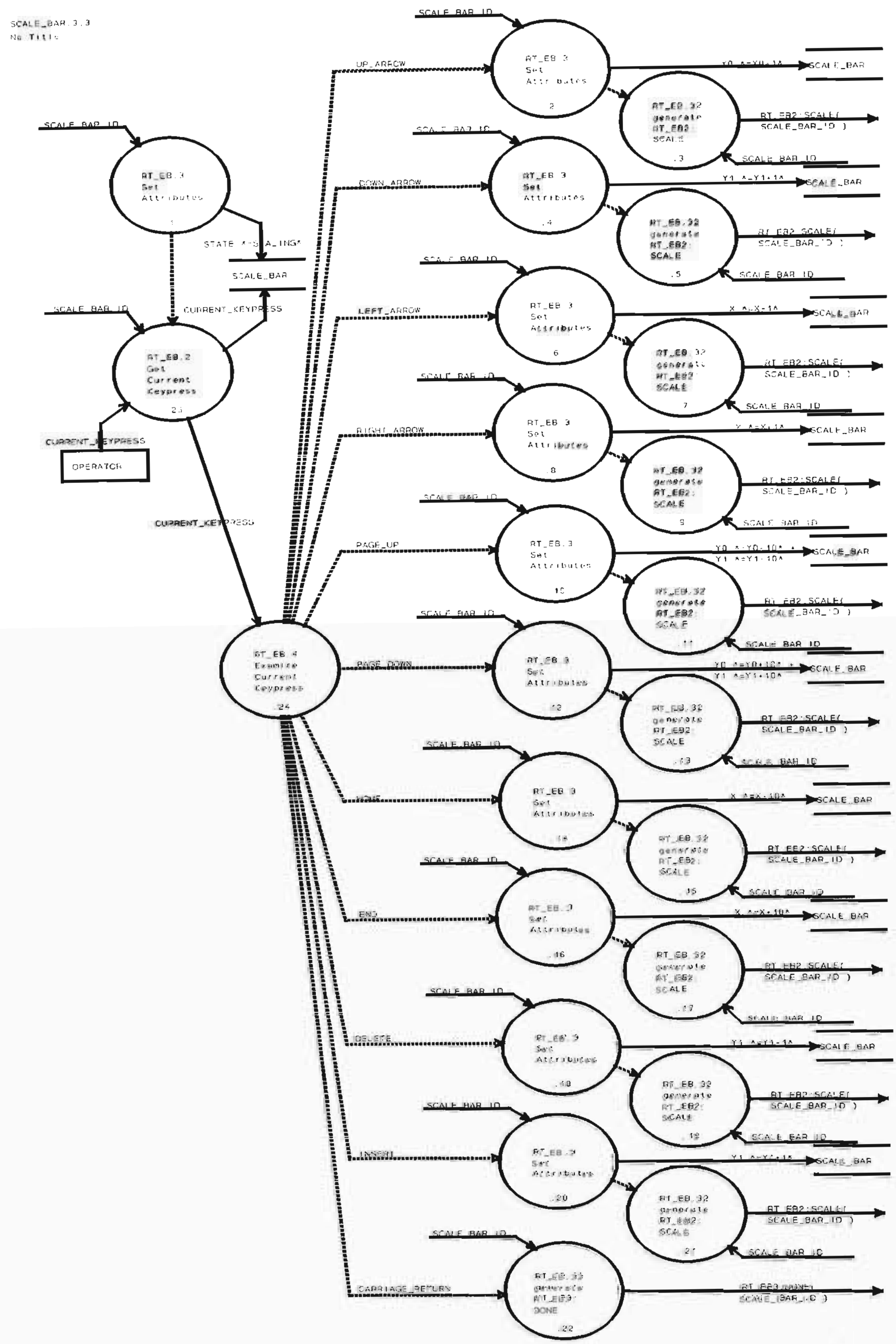


Figure D.86 Data Flow Diagram for Object Scale Bar in State Scaling

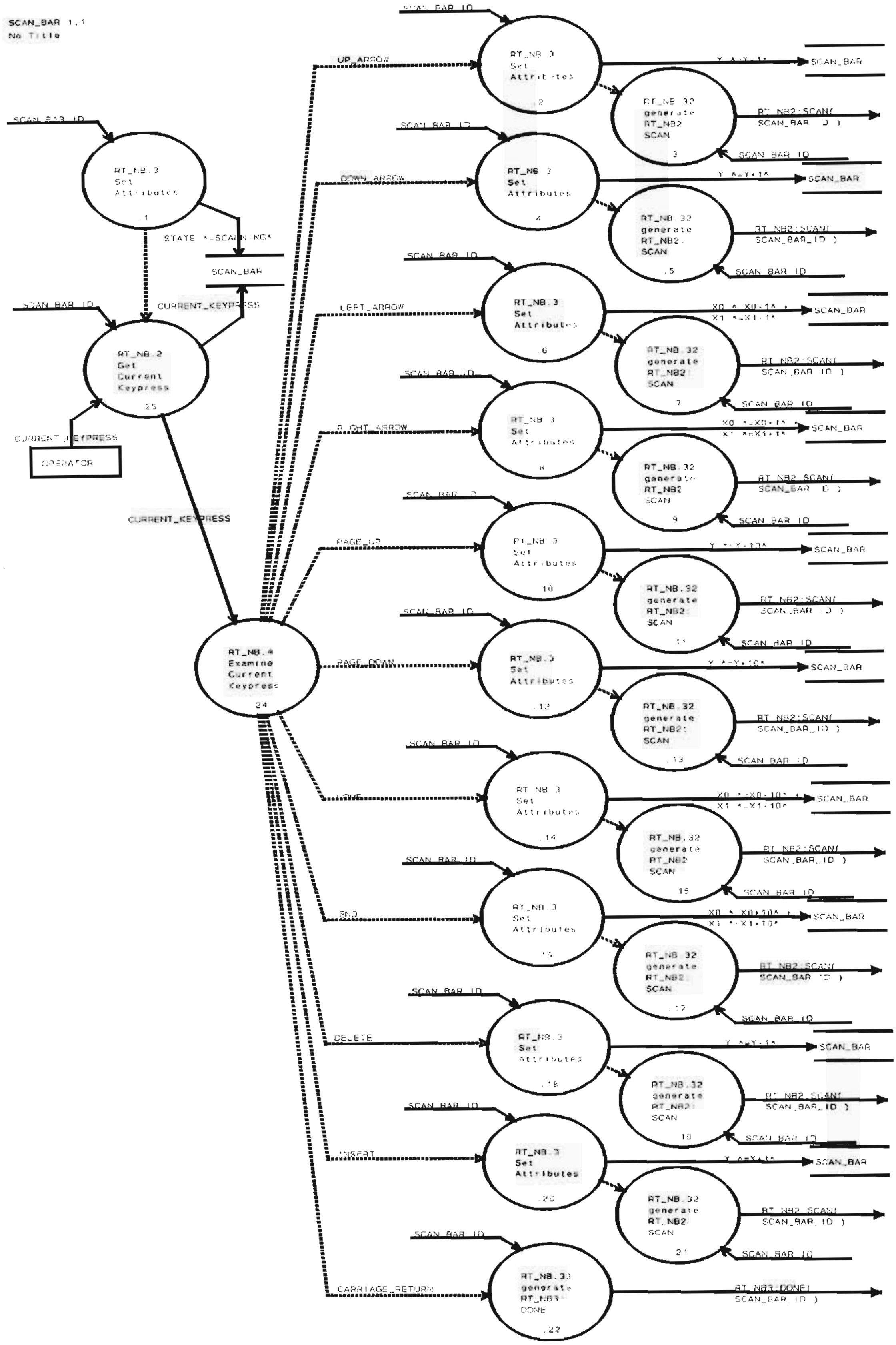


Figure D.87 Data Flow Diagram for Object Scan Bar in State Scanning

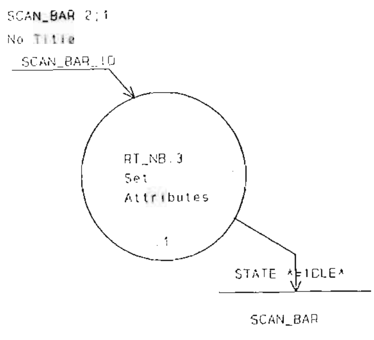


Figure D.88 Data Flow Diagram for Object Scan Bar in State Idle

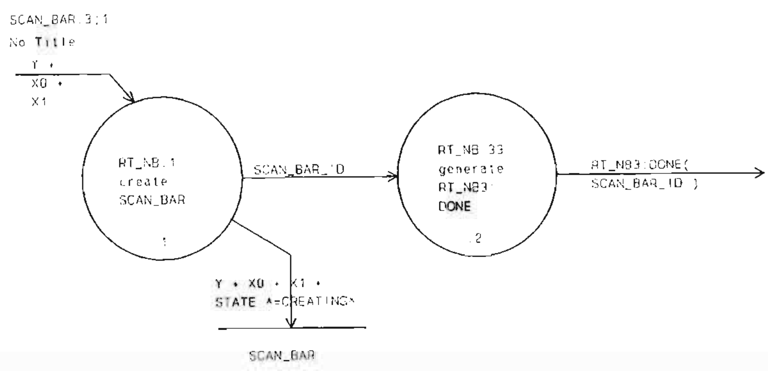


Figure D.89 Data Flow Diagram for Object Scan Bar in State Creating

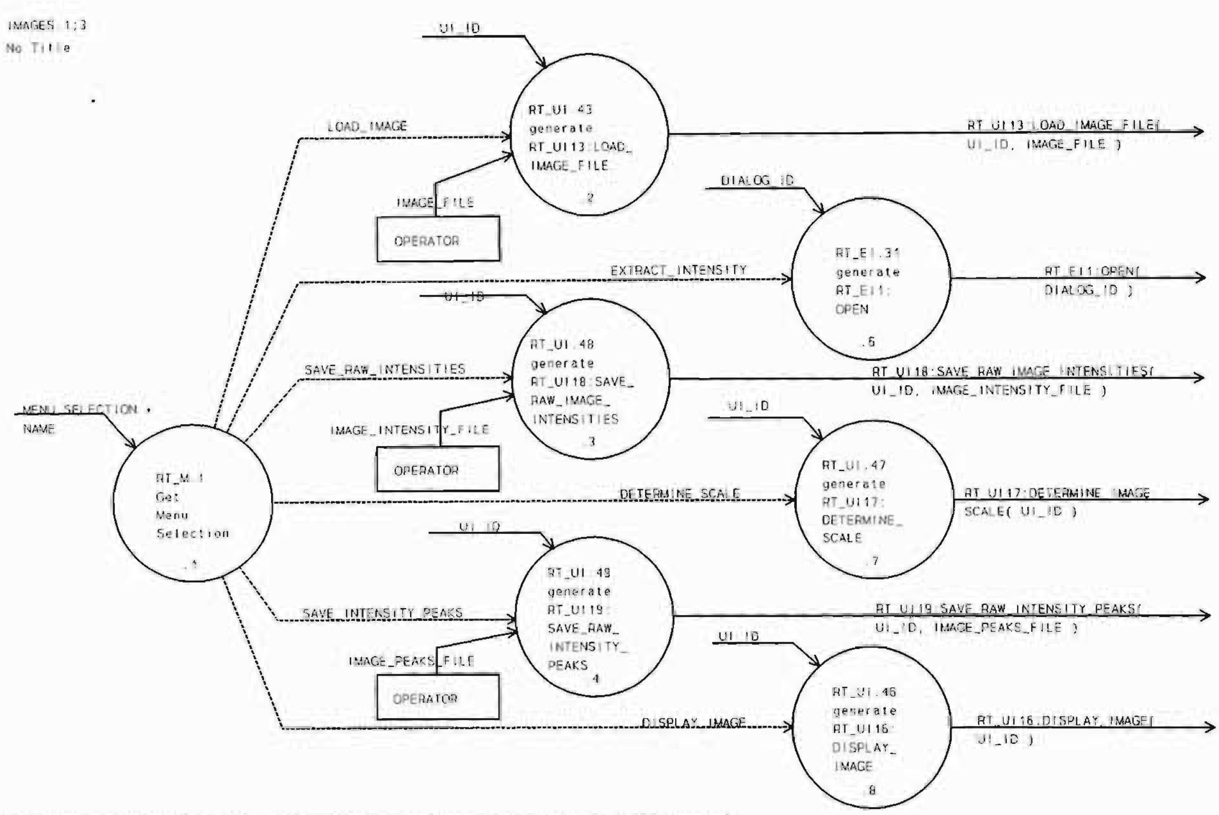


Figure D.90 Data Flow Diagram for Object Image Menu in State Temporal

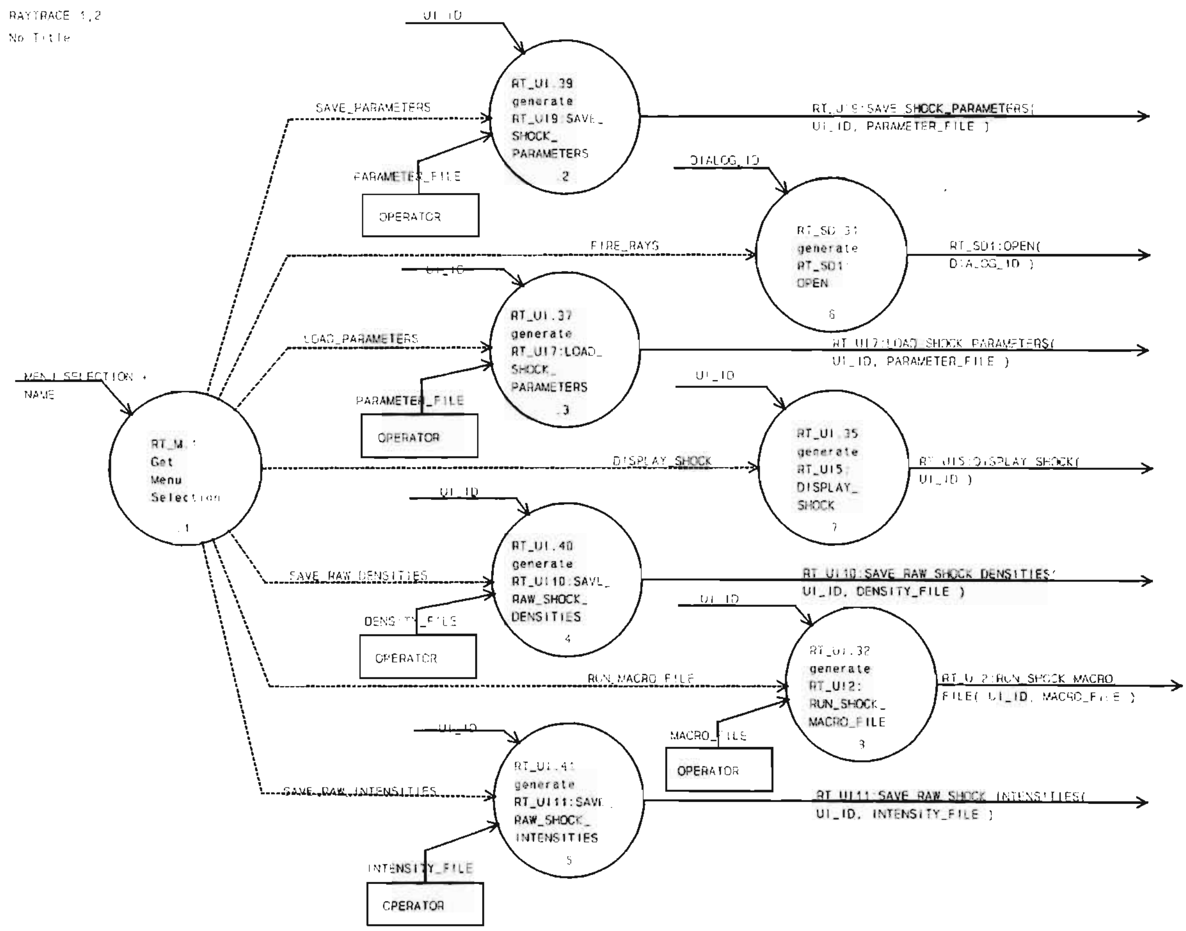


Figure D.91 Data Flow Diagram for Object Raytrace Menu in State Temporal

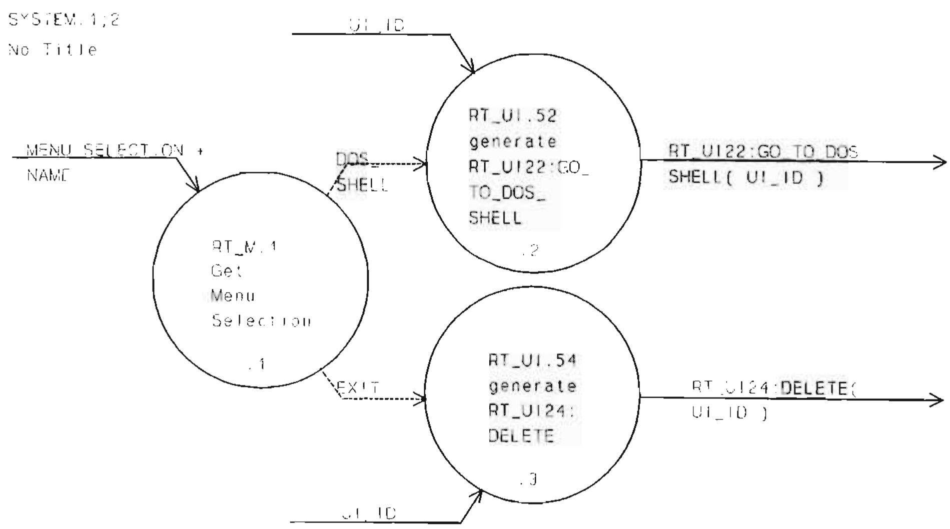


Figure D.92 Data Flow Diagram for Object System Menu in State Temporal

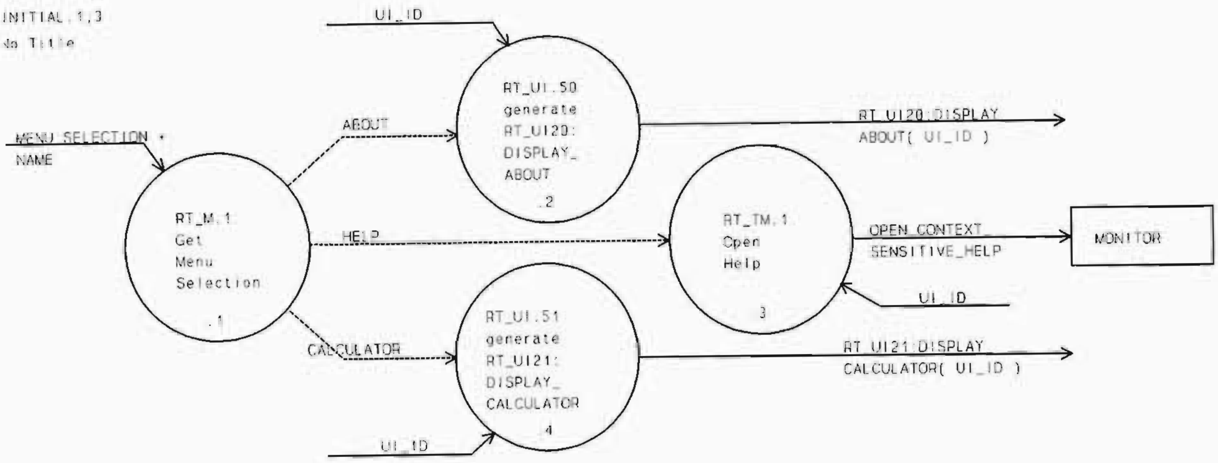


Figure D.93 Data Flow Diagram for Object Initial Menu in State Temporal

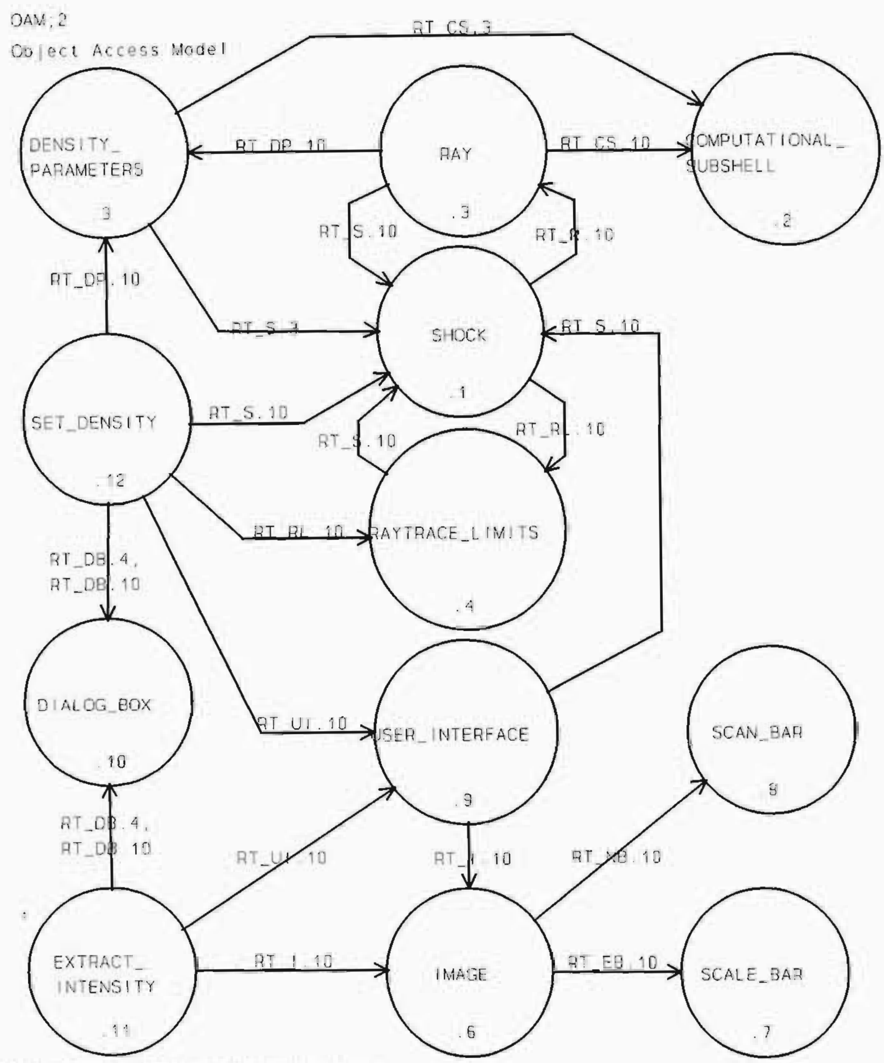


Figure D.94 Object Access Model for the entire domain

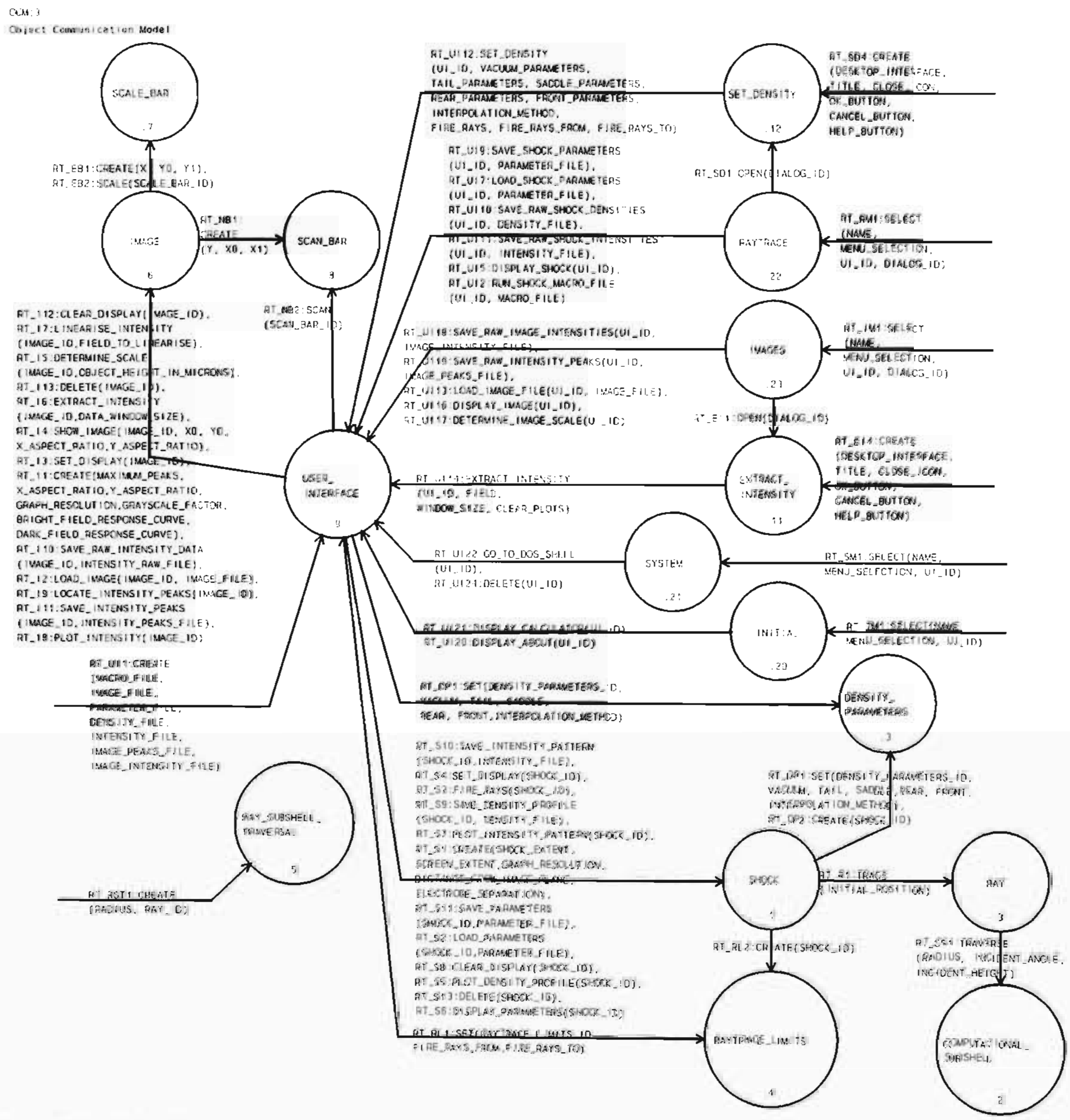


Figure D.95 Object Communication Model for the entire domain

D.14 State Process Table

Table D.3 State Process Table

ID	TYPE	NAME	USED IN STD	USED IN DFD
RT_CS.2	Transformation	Calculate Optical Path Length	COMPUTATIONAL_SUBSHELL	1
RT_CS.3	Accessor	Set Attributes	COMPUTATIONAL_SUBSHELL	1
RT_CS.3	Accessor	Set Attributes	DENSITY_PARAMETERS	1
RT_CS.4	Transformation	Calculate Emergent Height	COMPUTATIONAL_SUBSHELL	1
RT_CS.5	Transformation	Subshell Boundary	RAY	1
RT_CS.6	Transformation	Previous Subshell	RAY	1
RT_CS.10	Accessor	Get Attributes	COMPUTATIONAL_SUBSHELL	1
RT_CS.10	Accessor	Get Attributes	RAY	1
RT_CS.31	Event Generator	generate RT_CS1:TRAVERSE	RAY	1
RT_DB.4	Accessor	Get User Selection	EXTRACT_INTENSITY	2
RT_DB.4	Accessor	Get User Selection	SET_DENSITY	2
RT_DB.10	Accessor	Get Attributes	EXTRACT_INTENSITY	1, 2
RT_DB.10	Accessor	Get Attributes	SET_DENSITY	1, 2
RT_DP.1	Accessor	create DENSITY_PARAMETERS	DENSITY_PARAMETERS	2
RT_DP.2	Transformation	Interpolate Density Profile	DENSITY_PARAMETERS	1
RT_DP.3	Accessor	Set Attributes	DENSITY_PARAMETERS	1
RT_DP.4	Transformation	Calculate Refractive Index	DENSITY_PARAMETERS	1
RT_DP.5	Transformation	Extract Data for Subshell	DENSITY_PARAMETERS	1
RT_DP.10	Accessor	Get Attributes	DENSITY_PARAMETERS	1
RT_DP.10	Accessor	Get Attributes	RAY	1
RT_DP.10	Accessor	Get Attributes	SET_DENSITY	1
RT_DP.10	Transformation	Get Attributes	RAYTRACE_LIMITS	1
RT_DP.10	Transformation	Get Attributes	SHOCK	9, 5
RT_DP.31	Event Generator	generate RT_DP1:SET	SHOCK	10
RT_DP.31	Event Generator	generate RT_DP1:SET	USER_INTERFACE	27, 47
RT_DP.32	Event Generator	generate RT_DP2:CREATE	SHOCK	2
RT_EB.1	Accessor	create SCALE_BAR	SCALE_BAR	2

Table D.3 State Process Table

ID	TYPE	NAME	USED IN STD	USED IN DFD
RT_EB.2	Accessor	Get Current Keypress	SCALE_BAR	3
RT_EB.3	Accessor	Set Attributes	SCALE_BAR	3, 1
RT_EB.4	Test	Examine Current Keypress	SCALE_BAR	3
RT_EB.10	Accessor	Get Attributes	IMAGE	2
RT_EB.31	Event Generator	generate RT_EB1:CREATE	IMAGE	9
RT_EB.32	Event Generator	generate RT_EB2:SCALE	IMAGE	2
RT_EB.32	Event Generator	generate RT_EB2:SCALE	SCALE_BAR	3
RT_EB.33	Event Generator	generate RT_EB3:DONE	SCALE_BAR	2, 3
RT_EI.1	Accessor	create EXTRACT_INTENSITY	EXTRACT_INTENSITY	4
RT_EI.2	Transformation	Get Data For Fields	EXTRACT_INTENSITY	1
RT_EI.3	Accessor	Set Attributes	EXTRACT_INTENSITY	1, 3, 2
RT_EI.5	Accessor	Get Data From Fields	EXTRACT_INTENSITY	2
RT_EI.6	Test	Check Data From Fields	EXTRACT_INTENSITY	2
RT_EI.10	Accessor	Get Attributes	EXTRACT_INTENSITY	2
RT_EI.31	Event Generator	generate RT_EI1:OPEN	IMAGES	1
RT_EI.32	Event Generator	generate RT_EI2:ACTION	EXTRACT_INTENSITY	1, 2
RT_EI.33	Event Generator	generate RT_EI3:CLOSE	EXTRACT_INTENSITY	2
RT_I.1	Accessor	create IMAGE	IMAGE	9
RT_I.2	Accessor	Load Image Data	IMAGE	8
RT_I.3	Accessor	Set Attributes	IMAGE	12, 10, 4, 1, 8, 13, 5, 2, 6, 3, 11, 7
RT_I.4	Accessor	Initialise Graphics	IMAGE	6
RT_I.5	Accessor	Setup Graph Windows	IMAGE	6
RT_I.6	Accessor	Write Raw Intensity Data	IMAGE	7
RT_I.7	Accessor	Locate Intensity Peaks	IMAGE	11
RT_I.8	Accessor	Write Intensity Peaks	IMAGE	12
RT_I.9	Accessor	Close Graphics	IMAGE	10
RT_I.10	Accessor	Get Attributes	EXTRACT_INTENSITY	1

Table D.3 State Process Table

ID	TYPE	NAME	USED IN STD	USED IN DFD
RT_I.10	Accessor	Get Attributes	IMAGE	12, 4, 1, 5, 2, 9, 3, 11, 7
RT_I.10	Accessor	Get Attributes	USER_INTERFACE	36, 35, 39
RT_I.11	Accessor	delete IMAGE	IMAGE	13
RT_I.13	Transformation	Scale Image	IMAGE	2
RT_I.14	Transformation	Map Pixel Value	IMAGE	3
RT_I.15	Accessor	Plot Pixel	IMAGE	3
RT_I.16	Transformation	Get Pixel Position	IMAGE	3
RT_I.17	Test	Determine Field To Linearise	IMAGE	1
RT_I.18	Transformation	Deliver Pixels in Bright Row	IMAGE	1
RT_I.19	Transformation	Linearise Bright Pixel	IMAGE	1
RT_I.20	Transformation	Linearise Dark Pixel	IMAGE	1
RT_I.21	Transformation	Position and Size Data Window	IMAGE	4
RT_I.22	Transformation	Average and Reverse Columns in Data Window	IMAGE	4
RT_I.23	Accessor	Plot Intensity	IMAGE	5
RT_I.24	Transformation	Deliver Pixels in Dark Row	IMAGE	1
RT_I.31	Event Generator	generate RT_I1:CREATE	USER_INTERFACE	1
RT_I.32	Event Generator	generate RT_I2:LOAD_IMAGE	USER_INTERFACE	34
RT_I.33	Event Generator	generate RT_UI3:SET_DISPLAY	USER_INTERFACE	36, 35, 39
RT_I.34	Event Generator	generate RT_I4:SHOW_IMAGE	USER_INTERFACE	36, 35, 39
RT_I.35	Event Generator	generate RT_I5: DETERMINE_SCALE	USER_INTERFACE	36
RT_I.36	Event Generator	generate RT_I6:EXTRACT_INTENSITY	USER_INTERFACE	35
RT_I.37	Event Generator	generate RT_I7:LINEARISE_INTENSITY	USER_INTERFACE	35
RT_I.38	Event Generator	generate RT_I8:PLOT_INTENSITY	USER_INTERFACE	35, 39

Table D.3 State Process Table

ID	TYPE	NAME	USED IN STD	USED IN DFD
RT_I.39	Event Generator	generate RT_I9:LOCATE_ INTENSITY_PEAKS	USER_INTERFACE	38
RT_I.40	Event Generator	generate RT_I10:SAVE_RAW_ INTENSITY_DATA	USER_INTERFACE	37
RT_I.41	Event Generator	generate RT_I11:SAVE_INTENSITY_ PEAKS	USER_INTERFACE	38
RT_I.42	Event Generator	generate RT_I12:CLEAR_DISPLAY	USER_INTERFACE	46
RT_I.43	Event Generator	generate RT_I13:DELETE	USER_INTERFACE	23
RT_M.1	Test	Get Menu Selection	IMAGES	1
RT_M.1	Test	Get Menu Selection	INITIAL	1
RT_M.1	Test	Get Menu Selection	RAYTRACE	1
RT_M.1	Test	Get Menu Selection	SYSTEM	1
RT_NB.1	Accessor	create SCAN_BAR	SCAN_BAR	3
RT_NB.2	Accessor	Get Current Keypress	SCAN_BAR	1
RT_NB.3	Accessor	Set Attributes	SCAN_BAR	1
RT_NB.4	Test	Examine Current Keypress	SCAN_BAR	1
RT_NB.10	Accessor	Get Attributes	IMAGE	4, 11, 7
RT_NB.31	Event Generator	generate RT_NB1:CREATE	IMAGE	9
RT_NB.32	Event Generator	generate RT_NB2: SCAN	SCAN_BAR	1
RT_NB.32	Event Generator	generate RT_NB2:SCAN	USER_INTERFACE	35
RT_NB.33	Event Generator	generate RT_NB3: DONE	SCAN_BAR	3, 1
RT_R.1	Accessor	Determine First Free OPL	RAY	1
RT_R.2	Transformation	Current Subshell	RAY	1
RT_R.4	Accessor	Increment OPL in Shock	RAY	1
RT_R.5	Accessor	Determine Second Free OPL	RAY	1
RT_R.6	Accessor	Determine Total OPL	RAY	1
RT_R.7	Accessor	Determine Phase	RAY	1
RT_R.8	Accessor	Determine Final Position	RAY	1
RT_R.10	Accessor	Get Attributes	RAY	1

Table D.3 State Process Table				
ID	TYPE	NAME	USED IN STD	USED IN DFD
RT_R.10	Accessor	Get Attributes	SHOCK	6
RT_R.31	Event Generator	generate RT_R1:TRACE	SHOCK	6
RT_RL.1	Accessor	create RAYTRACE_ LIMITS	RAYTRACE_LIMITS	2
RT_RL.2	Transformation	Constrain Limits	RAYTRACE_LIMITS	1
RT_RL.3	Accessor	Set Attributes	RAYTRACE_LIMITS	1
RT_RL.10	Accessor	Get Attributes	RAYTRACE_LIMITS	1
RT_RL.10	Accessor	Get Attributes	SET_DENSITY	1
RT_RL.10	Accessor	Get Attributes	SHOCK	6
RT_RL.31	Event Generator	generate RT_RL1:SET	USER_INTERFACE	27, 47
RT_RL.32	Event Generator	generate RT_RL2:CREATE	SHOCK	2
RT_RST.1	Accessor	create RAY_SUBSHELL_ TRAVERSAL	RAY_SUBSHELL_TRAVERSAL	1
RT_S.1	Accessor	create SHOCK	SHOCK	2
RT_S.2	Accessor	Initialise Density Attributes	SHOCK	2
RT_S.3	Accessor	Set Attributes	DENSITY_PARAMETERS	1
RT_S.3	Accessor	Set Attributes	SHOCK	9, 14, 12, 8, 1, 15, 10, 13, 11, 7, 5, 6
RT_S.4	Accessor	Initialise Intensity Attributes	SHOCK	2
RT_S.5	Accessor	Read Parameter File	SHOCK	10
RT_S.6	Accessor	Interpolate Intensity Parameters	SHOCK	10
RT_S.7	Accessor	Initialise Graphics	SHOCK	11
RT_S.8	Accessor	Increment Electric Field	SHOCK	6
RT_S.9	Accessor	Determine Light Intensity	SHOCK	6
RT_S.10	Accessor	Get Attributes	RAY	1
RT_S.10	Accessor	Get Attributes	RAYTRACE_LIMITS	1
RT_S.10	Accessor	Get Attributes	SET_DENSITY	1
RT_S.10	Accessor	Get Attributes	SHOCK	9, 14, 12, 8, 10, 11, 7, 5, 6
RT_S.10	Accessor	Get Attributes	USER_INTERFACE	36, 27, 47

Table D.3 State Process Table				
ID	TYPE	NAME	USED IN STD	USED IN DFD
RT_S.11	Accessor	Smooth and Limit Light Intensity	SHOCK	6
RT_S.12	Accessor	Plot Density Profile	SHOCK	12
RT_S.13	Accessor	Plot Intensity Pattern	SHOCK	14
RT_S.14	Accessor	Display Density Parameters	SHOCK	5
RT_S.15	Accessor	Save Density Profile	SHOCK	8
RT_S.16	Accessor	Save Intensity Pattern	SHOCK	7
RT_S.17	Transformation	Extract Intensity Peaks	SHOCK	9
RT_S.18	Accessor	Save Density Parameters	SHOCK	9
RT_S.19	Accessor	Save Intensity Parameters	SHOCK	9
RT_S.20	Accessor	Close Graphics	SHOCK	13
RT_S.21	Accessor	delete SHOCK	SHOCK	15
RT_S.31	Event Generator	generate RT_S1:CREATE	USER_INTERFACE	1
RT_S.32	Event Generator	generate RT_I2:LOAD_ PARAMETERS	USER_INTERFACE	30
RT_S.33	Event Generator	generate RT_S3:FIRE_RAYS	USER_INTERFACE	27, 47
RT_S.34	Event Generator	generate RT_S4:SET_DISPLAY	USER_INTERFACE	27, 29, 47
RT_S.35	Event Generator	generate RT_S5:PLOT_ DENSITY_PROFILE	USER_INTERFACE	27, 29, 47
RT_S.36	Event Generator	generate RT_S6:DISPLAY_ PARAMETERS	USER_INTERFACE	27, 29, 47
RT_S.37	Event Generator	generate RT_S7:PLOT_ INTENSITY_PATTERN	USER_INTERFACE	27, 29, 47
RT_S.38	Event Generator	generate RT_S8:CLEAR_DISPLAY	USER_INTERFACE	44
RT_S.39	Event Generator	generate RT_S9:SAVE_ DENSITY_PROFILE	USER_INTERFACE	47, 32
RT_S.40	Event Generator	generate RT_S10:SAVE_ INTENSITY_PATTERN	USER_INTERFACE	47, 33
RT_S.41	Event Generator	generate RT_S11:SAVE_ PARAMETERS	USER_INTERFACE	31, 47
RT_S.42	Event Generator	generate RT_S12:DONE	SHOCK	9, 8, 13, 7

Table D.3 State Process Table				
ID	TYPE	NAME	USED IN STD	USED IN DFD
RT_S.43	Event Generator	generate RT_S13:DELETE	USER_INTERFACE	23
RT_SD.1	Accessor	create SET_ DENSITY	SET_DENSITY	5
RT_SD.2	Transformation	Get Data For Fields	SET_DENSITY	1
RT_SD.3	Accessor	Set Attributes	SET_DENSITY	1, 2, 4
RT_SD.5	Accessor	Get Data From Fields	SET_DENSITY	2
RT_SD.6	Test	Check Data From Fields	SET_DENSITY	2
RT_SD.10	Accessor	Get Attributes	SET_DENSITY	2
RT_SD.31	Event Generator	generate RT_SD1:OPEN	RAYTRACE	1
RT_SD.32	Event Generator	generate RT_SD2:ACTION	SET_DENSITY	1, 2
RT_SD.33	Event Generator	generate RT_SD3:CLOSE	SET_DENSITY	2
RT_TM.1	Accessor	Open Help	INITIAL	1
RT_UI.1	Accessor	create USER_INTERFACE	USER_INTERFACE	1
RT_UI.2	Accessor	Update Indicators	USER_INTERFACE	4
RT_UI.3	Accessor	Set Attributes	USER_INTERFACE	31, 36, 23, 48, 28, 44, 30, 35, 39, 41, 27, 29, 47, 34, 38, 42, 33, 45, 40, 32, 4, 37, 46, 43
RT_UI.4	Test	Has operator pressed key	USER_INTERFACE	45, 43
RT_UI.5	Accessor	Launch System Calculator	USER_INTERFACE	41
RT_UI.6	Accessor	Swap Environment Vectors	USER_INTERFACE	42
RT_UI.7	Accessor	Get Environment Vectors	USER_INTERFACE	42
RT_UI.8	Accessor	Initialise Menu System	USER_INTERFACE	42
RT_UI.9	Accessor	Close Menu System	USER_INTERFACE	42
RT_UI.10	Accessor	Get Attributes	EXTRACT_INTENSITY	1
RT_UI.10	Accessor	Get Attributes	SET_DENSITY	1
RT_UI.10	Accessor	Get Attributes	USER_INTERFACE	31, 36, 23, 44, 30, 35, 39, 27, 29, 47, 34, 38, 33, 32, 4, 37, 46
RT_UI.11	Test	Is Fire Rays Selected	USER_INTERFACE	27
RT_UI.12	Accessor	Open Macro File	USER_INTERFACE	48

Table D.3 State Process Table				
ID	TYPE	NAME	USED IN STD	USED IN DFD
RT_UI.14	Accessor	End Of Macro File	USER_INTERFACE	28
RT_UI.15	Accessor	Next Token	USER_INTERFACE	28
RT_UI.16	Test	Examine Token Type	USER_INTERFACE	28
RT_UI.17	Accessor	Clear Files	USER_INTERFACE	28
RT_UI.18	Accessor	Close Macro File	USER_INTERFACE	28
RT_UI.19	Accessor	delete USER_INTERFACE	USER_INTERFACE	23
RT_UI.24	Accessor	Display System Credits	USER_INTERFACE	40
RT_UI.25	Accessor	Check Parameters	USER_INTERFACE	28
RT_UI.26	Test	Display Graphics	USER_INTERFACE	47
RT_UI.32	Event Generator	generate RT_UI2:RUN_ SHOCK_MACRO_FILE	RAYTRACE	1
RT_UI.33	Event Generator	generate RT_UI3:PARSE_ MACRO_FILE	USER_INTERFACE	48
RT_UI.34	Event Generator	generate RT_UI4:AUTO_SHOCK	USER_INTERFACE	28
RT_UI.35	Event Generator	generate RT_UI5:DISPLAY_SHOCK	RAYTRACE	1
RT_UI.35	Event Generator	generate RT_UI5:DISPLAY_SHOCK	USER_INTERFACE	30
RT_UI.36	Event Generator	generate RT_UI6:WAIT_FOR_SHOCK	USER_INTERFACE	27, 29, 43
RT_UI.37	Event Generator	generate RT_UI7:LOAD_ SHOCK_ PARAMETERS	RAYTRACE	1
RT_UI.39	Event Generator	generate RT_UI9:SAVE_ SHOCK_ PARAMETERS	RAYTRACE	1
RT_UI.40	Event Generator	generate RT_UI10:SAVE_ RAW_SHOCK_DENSITIES	RAYTRACE	1
RT_UI.41	Event Generator	generate RT_UI11:SAVE_RAW_ SHOCK_INTENSITIES	RAYTRACE	1
RT_UI.42	Event Generator	generate RT_UI12:SET_DENSITY	SET_DENSITY	2
RT_UI.43	Event Generator	generate RT_UI13:LOAD_ IMAGE_FILE	IMAGES	1
RT_UI.44	Event Generator	generate RT_UI14: EXTRACT_INTENSITY	EXTRACT_INTENSITY	2

Table D.3 State Process Table				
ID	TYPE	NAME	USED IN STD	USED IN DFD
RT_UI.45	Event Generator	generate RT_UI15: WAIT_FOR_IMAGE	USER_INTERFACE	35, 39, 45
RT_UI.46	Event Generator	generate RT_UI16: DISPLAY_IMAGE	IMAGES	1
RT_UI.47	Event Generator	generate RT_UI17: DETERMINE_ SCALE	IMAGES	1
RT_UI.48	Event Generator	generate RT_UI18:SAVE_ RAW_IMAGE_INTENSITIES	IMAGES	1
RT_UI.49	Event Generator	generate RT_UI19:SAVE_ RAW_INTENSITY_PEAKS	IMAGES	1
RT_UI.50	Event Generator	generate RT_UI20: DISPLAY_ABOUT	INITIAL	1
RT_UI.51	Event Generator	generate RT_UI21: DISPLAY_CALCULATOR	INITIAL	1
RT_UI.52	Event Generator	generate RT_UI22: GO_TO_DOS_SHELL	SYSTEM	1
RT_UI.53	Event Generator	generate RT_UI23:DONE	USER_INTERFACE	1, 31, 36, 28, 44, 41, 47, 34, 38, 42, 33, 45, 40, 32, 37, 46, 43
RT_UI.54	Event Generator	generate RT_UI24:DELETE	SYSTEM	1

D.15 Process Specifications

RT_CS.1 SnellsLaw

Process Type: Transformation

Interface: RADIUS : data_in
INCIDENT_ANGLE : data_in
INCIDENT_HEIGHT : data_in
REFRACTIVE_INDEX : data_in
REFRACTED_ANGLE : data_out

Description: Applies Snell's Law to compute the refracted angle of a ray of light incident on the boundary between two optical media according to equation (2.4).

RT_CS.2 CalculateOpticalPathLength

Process Type: Transformation

Interface: REFRACTED_ANGLE : data_in
WIDTH : data_in
RADIUS : data_in
INCIDENT_ANGLE : data_in
INCIDENT_HEIGHT : data_in
OPTICAL_PATH_LENGTH : data_out

Description: Calculates the optical path length travelled by the ray within a subshell according to equation (2.11).

RT_CS.3 SetAttributes

Process Type: Accessor

Interface: RADIUS : data_in
DENSITY : data_in
REFRACTIVE_INDEX : data_in
SHOCK_ID : data_in
WIDTH : data_in
INCIDENT_ANGLE : data_in
REFRACTED_ANGLE : data_in
INCIDENT_HEIGHT : data_in
EMERGENT_HEIGHT : data_in
OPTICAL_PATH_LENGTH : data_in
DENSITY : write_access
REFRACTIVE_INDEX : write_access
SHOCK_ID : write_access
WIDTH : write_access
INCIDENT_ANGLE : write_access
REFRACTED_ANGLE : write_access
INCIDENT_HEIGHT : write_access
EMERGENT_HEIGHT : write_access
OPTICAL_PATH_LENGTH : write_access

Description: Write accessor.

RT_CS.4 CalculateEmergentHeight

Process Type: Transformation

Interface: OPTICAL_PATH_LENGTH : data_in
WIDTH : data_in
RADIUS : data_in
INCIDENT_ANGLE : data_in
INCIDENT_HEIGHT : data_in
EMERGENT_HEIGHT : data_out

Description: Calculates the height at which an incident ray emerges from a subshell after refraction using equation (2.12). This emergent height forms the incident height into the next subshell.

RT_CS.5 SubshellBoundry
Process Type: Transformation
Interface: REFRACTED_ANGLE : data_in
EMERGENT_HEIGHT : data_in
RADIUS : data_in
INCIDENT_ANGLE : data_out
INCIDENT_HEIGHT : data_out
Description: Calculates the incident angle and height into the next subshell from the emergent height and refracted angle of the previous subshell again using equation (2.12).

RT_CS.6 PreviousSubshell
Process Type: Transformation
Interface: RADIUS : data_in
RADIUS : data_out
Description: Returns the identity of the previous subshell.

RT_CS.10 GetAttributes
Process Type: Accessor
Interface: RADIUS : data_in
OPTICAL_PATH_LENGTH : data_out
SHOCK_ID : data_out
WIDTH : data_out
DENSITY : data_out
REFRACTIVE_INDEX : data_out
INCIDENT_ANGLE : data_out
REFRACTED_ANGLE : data_out
INCIDENT_HEIGHT : data_out
EMERGENT_HEIGHT : data_out
OPTICAL_PATH_LENGTH : read_access
SHOCK_ID : read_access
WIDTH : read_access
DENSITY : read_access
REFRACTIVE_INDEX : read_access
INCIDENT_ANGLE : read_access
REFRACTED_ANGLE : read_access
INCIDENT_HEIGHT : read_access
EMERGENT_HEIGHT : read_access
Description: Read accessor.

RT_CS.31 generateRT_CS1:TRAVERSE
Process Type: Event Generator
Interface: INCIDENT_ANGLE : data_in
INCIDENT_HEIGHT : data_in
RADIUS : data_in
RT_CS1:TRAVERSE : generated_event
Description: Event generator.

RT_DB.4 GetUserSelection
Process Type: Accessor
Interface: DIALOG_ID : data_in
USER_SELECTION : data_in
CLOSE_BUTTON¦CLOSE_ICON : conditional_control_out
NONE : conditional_control_out
OKAY_BUTTON : conditional_control_out
Description: Returns the user selection of editor options.

RT_DB.3 SetAttributes

Process Type: Accessor

Interface: DIALOG_ID : data_in
DESKTOP_INTERFACE : data_in
TITLE : data_in
CLOSE_ICON : data_in
OK_BUTTON : data_in
CANCEL_BUTTON : data_in
HELP_BUTTON : data_in
STATE : data_in
DESKTOP_INTERFACE : write_access
TITLE : write_access
CLOSE_ICON : write_access
OK_BUTTON : write_access
CANCEL_BUTTON : write_access
HELP_BUTTON : write_access
STATE : write_access

Description: Write accessor.

RT_DB.10 GetAttributes

Process Type: Accessor

Interface: DIALOG_ID : data_in
DESKTOP_INTERFACE : data_out
TITLE : data_out
CLOSE_ICON : data_out
OK_BUTTON : data_out
CANCEL_BUTTON : data_out
HELP_BUTTON : data_out
STATE : data_out
DESKTOP_INTERFACE : read_access
TITLE : read_access
CLOSE_ICON : read_access
OK_BUTTON : read_access
CANCEL_BUTTON : read_access
HELP_BUTTON : read_access
STATE : read_access

Description: Read accessor.

RT_DP.1 createDENSITY_PARAMETERS

Process Type: Accessor

Interface: SHOCK_ID : data_in
SHOCK_ID : write_access
STATE : write_access

Description: Create accessor.

RT_DP.2 InterpolateDensityProfile

Process Type: Transformation

Interface: VACUUM : data_in
TAIL : data_in
SADDLE : data_in
REAR : data_in
FRONT : data_in
INTERPOLATION_METHOD : data_in
DENSITY_PROFILE : data_out

Description: Interpolates the point parameters representing the density profile, using the interpolation method, and produces a smooth density profile.

RT_DP.3 SetAttributes
Process Type: Accessor
Interface: DENSITY_PARAMETERS_ID : data_in
SHOCK_ID : data_in
VACUUM : data_in
TAIL : data_in
SADDLE : data_in
REAR : data_in
FRONT : data_in
INTERPOLATION_METHOD : data_in
STATE : data_in
SHOCK_ID : write_access
VACUUM : write_access
TAIL : write_access
SADDLE : write_access
REAR : write_access
FRONT : write_access
INTERPOLATION_METHOD : write_access
STATE : write_access
Description: Write accessor.

RT_DP.4 CalculateRefractiveIndex
Process Type: Transformation
Interface: DENSITY_PROFILE : data_in
REFRACTIVE_INDEX : data_out
Description: The refractive index of a gas depends empirically on its density. Once a density profile is selected for the shock, we can calculate the shocks refractive index at any point using equation (2.3).

RT_DP.5 ExtractDataforSubshell
Process Type: Transformation
Interface: DENSITY_PROFILE : data_in
REFRACTIVE_INDEX : data_in
FRONT : data_in
RADIUS : data_out
DENSITY : data_out
REFRACTIVE_INDEX : data_out
Description: The density and refractive index attributes for any subshell may be extracted from those of the whole shock.

RT_DP.10 GetAttributes
Process Type: Accessor
Interface: DENSITY_PARAMETERS.DENSITY_PARAMETERS_ID : data_in
VACUUM : data_out
TAIL : data_out
SADDLE : data_out
REAR : data_out
FRONT : data_out
INTERPOLATION_METHOD : data_out
SHOCK_ID : data_out
STATE : data_out
VACUUM : read_access
TAIL : read_access
SADDLE : read_access

REAR : read_access
FRONT : read_access
INTERPOLATION_METHOD : read_access
SHOCK_ID : read_access
STATE : read_access
Description: Read accessor.

RT_DP.31 generateRT_DP1:SET
Process Type: Event Generator
Interface: DENSITY_PARAMETERS.VACUUM : data_in
DENSITY_PARAMETERS.TAIL : data_in
DENSITY_PARAMETERS.SADDLE : data_in
DENSITY_PARAMETERS.REAR : data_in
DENSITY_PARAMETERS.FRONT : data_in
DENSITY_PARAMETERS.INTERPOLATION_METHOD : data_in
DENSITY_PARAMETERS.DENSITY_PARAMETERS_ID : data_in
RT_DP1:SET : generated_event
Description: Event generator.

RT_DP.32 generateRT_DP2:CREATE
Process Type: Event Generator
Interface: SHOCK_ID : data_in
RT_DP2:CREATE : generated_event
Description: Event generator.

RT_EB.1 createSCALE_BAR
Process Type: Accessor
Interface: X : data_in
Y0 : data_in
Y1 : data_in
SCALE_BAR_ID : data_out
X : write_access
Y0 : write_access
Y1 : write_access
STATE : write_access
Description: Create accessor.

RT_EB.2 GetCurrentKeypress
Process Type: Accessor
Interface: SCALE_BAR_ID : data_in
CURRENT_KEYPRESS : data_in
CURRENT_KEYPRESS : data_out
SCALE_BAR.CURRENT_KEYPRESS : write_access
Description: Returns the key pressed by the operator.

RT_EB.3 SetAttributes
Process Type: Accessor
Interface: SCALE_BAR_ID : data_in
X : data_in
Y0 : data_in
Y1 : data_in
CURRENT_KEYPRESS : data_in
STATE : data_in
X : write_access
Y0 : write_access

	Y1 : write_access
	CURRENT_KEYPRESS : write_access
	STATE : write_access
Description:	Write accessor.

RT_EB.4 ExamineCurrentKeypress

Process Type:	Test
Interface:	CURRENT_KEYPRESS : data_in
	UP_ARROW : conditional_control_out
	DOWN_ARROW : conditional_control_out
	LEFT_ARROW : conditional_control_out
	RIGHT_ARROW : conditional_control_out
	PAGE_UP : conditional_control_out
	PAGE_DOWN : conditional_control_out
	HOME : conditional_control_out
	END : conditional_control_out
	DELETE : conditional_control_out
	INSERT : conditional_control_out
	CARRIAGE_RETURN : conditional_control_out
Description:	Determines the type of keypress.

RT_EB.10 GetAttributes

Process Type:	Accessor
Interface:	SCALE_BAR_ID : data_in
	Y0 : data_out
	Y1 : data_out
	CURRENT_KEYPRESS : data_out
	STATE : data_out
	Y0 : read_access
	Y1 : read_access
	CURRENT_KEYPRESS : read_access
	STATE : read_access
Description:	Read accessor.

RT_EB.31 generateRT_EB1:CREATE

Process Type:	Event Generator
Interface:	SCALE_BAR.X : data_in
	SCALE_BAR.Y0 : data_in
	SCALE_BAR.Y1 : data_in
Description:	Event generator.

RT_EB.32 generateRT_EB2:SCALE

Process Type:	Event Generator
Interface:	SCALE_BAR_ID : data_in
	RT_EB2:SCALE : generated_event
Description:	Event generator.

RT_EB.33 generateRT_EB3:DONE

Process Type:	Event Generator
Interface:	SCALE_BAR_ID : data_in
	RT_EB3:DONE : generated_event
Description:	Event generator.

RT_EI.1 createEXTRACT_INTENSITY

Process Type:	Accessor

Interface: DESKTOP_INTERFACE : data_in
TITLE : data_in
CLOSE_ICON : data_in
OK_BUTTON : data_in
CANCEL_BUTTON : data_in
HELP_BUTTON : data_in
DESKTOP_INTERFACE : write_access
TITLE : write_access
CLOSE_ICON : write_access
OK_BUTTON : write_access
CANCEL_BUTTON : write_access
HELP_BUTTON : write_access
STATE : write_access

Description: Create accessor.

RT_EI.2 GetDataForFields

Process Type: Transformation

Interface: IMAGE.DATA_WINDOW_SIZE : data_in
IMAGE.FIELD_TO_LINEARISE : data_in
LINEARISE_FIELD_RADIO_BUTTON_CLUSTER : data_out
WINDOW_SIZE_RADIO_BUTTION_CLUSTER : data_out
CLEAR_PREVIOUS_PLOTS_CHECK_BOX : data_out

Description: Reads data from the database to set and display the latest values in the dialog box fields.

RT_EI.3 SetAttributes

Process Type: Accessor

Interface: DIALOG_ID : data_in
LINEARISE_FIELD_RADIO_BUTTON_CLUSTER : data_in
WINDOW_SIZE_RADIO_BUTTION_CLUSTER : data_in
CLEAR_PREVIOUS_PLOTS_CHECK_BOX : data_in
STATE : data_in
LINEARISE_FIELD_RADIO_BUTTON_CLUSTER : write_access
WINDOW_SIZE_RADIO_BUTTION_CLUSTER : write_access
CLEAR_PREVIOUS_PLOTS_CHECK_BOX : write_access
STATE : write_access

Description: Write accessor.

RT_EI.5 GetDataFromFields

Process Type: Accessor

Interface: DIALOG_ID : data_in
LINEARISE_FIELD_RADIO_BUTTON_CLUSTER : data_in
WINDOW_SIZE_RADIO_BUTTION_CLUSTER : data_in
CLEAR_PREVIOUS_PLOTS_CHECK_BOX : data_in
LINEARISE_FIELD_RADIO_BUTTON_CLUSTER : data_out
WINDOW_SIZE_RADIO_BUTTION_CLUSTER : data_out
CLEAR_PREVIOUS_PLOTS_CHECK_BOX : data_out

Description: Reads the latest, operator modified, data from the dialog box fields.

RT_EI.6 CheckDataFromFields

Process Type: Test

Interface: LINEARISE_FIELD_RADIO_BUTTON_CLUSTER : data_in
WINDOW_SIZE_RADIO_BUTTION_CLUSTER : data_in
CLEAR_PREVIOUS_PLOTS_CHECK_BOX : data_in
OK : conditional_control_out
NOT_OK : conditional_control_out

Description: Checks the latest, operator modified, data from the dialog box fields.

RT_EI.10 GetAttributes

Process Type: Accessor

Interface: DIALOG_ID : data_in
EXTRACT_INTENSITY.LINEARISE_FIELD_RADIO_BUTTON_CLUSTER : data_out
EXTRACT_INTENSITY.WINDOW_SIZE_RADIO_BUTTION_CLUSTER : data_out
EXTRACT_INTENSITY.CLEAR_PREVIOUS_PLOTS_CHECK_BOX : data_out
STATE : data_out
LINEARISE_FIELD_RADIO_BUTTON_CLUSTER : read_access
WINDOW_SIZE_RADIO_BUTTION_CLUSTER : read_access
CLEAR_PREVIOUS_PLOTS_CHECK_BOX : read_access
STATE : read_access

Description: Read accessor.

RT_EI.31 generateRT_EI1:OPEN

Process Type: Event Generator

Interface: DIALOG_ID : data_in
RT_EI1:OPEN : generated_event

Description: Event generator.

RT_EI.32 generateRT_EI2:ACTION

Process Type: Event Generator

Interface: DIALOG_ID : data_in
RT_EI2:ACTION : generated_event

Description: Event generator.

RT_EI.33 generateRT_EI3:CLOSE

Process Type: Event Generator

Interface: DIALOG_ID : data_in
RT_EI3:CLOSE : generated_event

Description: Event generator.

RT_I.1 createIMAGE

Process Type: Accessor

Interface: MAXIMUM_PEAKS : data_in
X_ASPECT_RATIO : data_in
Y_ASPECT_RATIO : data_in
GRAPH_RESOLUTION : data_in
GRAYSCALE_FACTOR : data_in
BRIGHT_FIELD_RESPONSE_CURVE : data_in
DARK_FIELD_RESPONSE_CURVE : data_in
IMAGE_ID : data_out
MAXIMUM_PEAKS : write_access
X_ASPECT_RATIO : write_access
Y_ASPECT_RATIO : write_access
GRAPH_RESOLUTION : write_access
GRAYSCALE_FACTOR : write_access
BRIGHT_FIELD_RESPONSE_CURVE : write_access
DARK_FIELD_RESPONSE_CURVE : write_access
NUMBER_OF_PEAKS : write_access
PEAKS : write_access
X0 : write_access
Y0 : write_access
DATA_WINDOW_SIZE : write_access

MICRONS_PER_PIXEL : write_access
STATE : write_access

Description: Create accessor.

RT_I.2 LoadImageData

Process Type: Accessor

Interface:
IMAGE_ID : data_in
IMAGE_FILE : data_in
IMAGE_DATA : data_in
IMAGE.IMAGE_FILE : write_access
IMAGE.IMAGE_DATA : write_access

Description: Loads, but does not display, the image file.

RT_I.3 SetAttributes

Process Type: Accessor

Interface:
IMAGE_ID : data_in
SCALE_BAR : data_in
SCAN_BAR : data_in
X0 : data_in
Y0 : data_in
IMAGE_SIZE : data_in
IMAGE_DATA : data_in
DATA_WINDOW_SIZE : data_in
DATA_WINDOW : data_in
INTENSITY : data_in
PEAKS : data_in
NUMBER_OF_PEAKS : data_in
MAXIMUM_PEAKS : data_in
IMAGE_FILE : data_in
INTENSITY_RAW_FILE : data_in
INTENSITY_PEAKS_FILE : data_in
OBJECT_HEIGHT_IN_MICRONS : data_in
MICRONS_PER_PIXEL : data_in
FIELD_TO_LINEARISE : data_in
X_ASPECT_RATIO : data_in
Y_ASPECT_RATIO : data_in
GRAPH_RESOLUTION : data_in
IMAGE_GRAPH_WINDOW : data_in
INTENSITY_GRAPH_WINDOW : data_in
GRAYSCALE_FACTOR : data_in
BRIGHT_FIELD_RESPONSE_CURVE : data_in
DARK_FIELD_RESPONSE_CURVE : data_in
STATE : data_in
SCALE_BAR : write access
SCAN_BAR : write access
X0 : write access
Y0 : write access
IMAGE_SIZE : write access
IMAGE_DATA : write access
DATA_WINDOW_SIZE : write access
DATA_WINDOW : write access
INTENSITY : write access
PEAKS : write access
NUMBER_OF_PEAKS : write access
MAXIMUM_PEAKS : write access

IMAGE_FILE : write access
INTENSITY_RAW_FILE : write access
INTENSITY_PEAKS_FILE : write access
OBJECT_HEIGHT_IN_MICRONS : write access
MICRONS_PER_PIXEL : write access
FIELD_TO_LINEARISE : write access
X_ASPECT_RATIO : write access
Y_ASPECT_RATIO : write access
GRAPH_RESOLUTION : write access
IMAGE_GRAPH_WINDOW : write access
INTENSITY_GRAPH_WINDOW : write access
GRAYSCALE_FACTOR : write access
BRIGHT_FIELD_RESPONSE_CURVE : write access
DARK_FIELD_RESPONSE_CURVE : write access
STATE : write_access

Description: Write accessor.

RT_I.4 InitialiseGraphics

Process Type: Accessor

Interface: IMAGE_ID : data_in
GRAYSCALE_FACTOR : data_out
IMAGE.GRAYSCALE_FACTOR : read_access

Description: Initialises the graphics card and returns the number of grayscales it is capable of displaying.

RT_I.5 SetupGraphWindows

Process Type: Accessor

Interface: IMAGE_ID : data_in
IMAGE_GRAPH_WINDOW : write_access
INTENSITY_GRAPH_WINDOW : write_access

Description: Sets up a graphics window (titles, scaling, aspect ratios etc.) to display the image.

RT_I.6 WriteRawIntensityData

Process Type: Accessor

Interface: INTENSITY : data_in
SCAN_BAR.X0 : data_in
SCAN_BAR.X1 : data_in
INTENSITY_RAW_FILE : data_in
INTENSITY : data_out

Description: Saves raw intensity data, extracted from a data window around the scan bar, to a file.

RT_I.7 LocateIntensityPeaks

Process Type: Accessor

Interface: MAXIMUM_PEAKS : data_in
MICRONS_PER_PIXEL : data_in
INTENSITY : data_in
PEAKS : data_in
SCAN_BAR.X0 : data_in
SCAN_BAR.X1 : data_in
PEAKS : write_access
NUMBER_OF_PEAKS : write_access

Description: Locates peaks in the intensity data extracted from a data window around the scan bar.

RT_I.8 WriteIntensityPeaks

Process Type: Accessor

Interface: NUMBER_OF_PEAKS : data_in

PEAKS : data_in
INTENSITY_PEAKS_FILE : data_in
INTENSITY_PEAKS_FILE : data_in
PEAKS : data_out

Description: Writes the located intensity peaks to file.

RT_I.9 CloseGraphics

Process Type: Accessor

Interface: IMAGE_ID : data_in
CLOSE : data_out

Description: Closes the system graphics and returns to the text-based menu system.

RT_I.10 GetAttributes

Process Type: Accessor

Interface: IMAGE_ID : data_in
SCALE_BAR : data_out
SCAN_BAR : data_out
X0 : data_out
Y0 : data_out
IMAGE_SIZE : data_out
IMAGE_DATA : data_out
DATA_WINDOW_SIZE : data_out
DATA_WINDOW : data_out
INTENSITY : data_out
PEAKS : data_out
NUMBER_OF_PEAKS : data_out
MAXIMUM_PEAKS : data_out
IMAGE_FILE : data_out
INTENSITY_RAW_FILE : data_out
INTENSITY_PEAKS_FILE : data_out
OBJECT_HEIGHT_IN_MICRONS : data_out
MICRONS_PER_PIXEL : data_out
FIELD_TO_LINEARISE : data_out
X_ASPECT_RATIO : data_out
Y_ASPECT_RATIO : data_out
GRAPH_RESOLUTION : data_out
IMAGE_GRAPH_WINDOW : data_out
INTENSITY_GRAPH_WINDOW : data_out
GRAYSCALE_FACTOR : data_out
BRIGHT_FIELD_RESPONSE_CURVE : data_out
DARK_FIELD_RESPONSE_CURVE : data_out
STATE : data_out
SCALE_BAR : read_access
SCAN_BAR : read_access
X0 : read_access
Y0 : read_access
IMAGE_SIZE : read_access
IMAGE_DATA : read_access
DATA_WINDOW_SIZE : read_access
DATA_WINDOW : read_access
INTENSITY : read_access
PEAKS : read_access
NUMBER_OF_PEAKS : read_access
MAXIMUM_PEAKS : read_access
IMAGE_FILE : read_access

INTENSITY_RAW_FILE : read_access
INTENSITY_PEAKS_FILE : read_access
OBJECT_HEIGHT_IN_MICRONS : read_access
MICRONS_PER_PIXEL : read_access
FIELD_TO_LINEARISE : read_access
X_ASPECT_RATIO : read_access
Y_ASPECT_RATIO : read_access
GRAPH_RESOLUTION : read_access
IMAGE_GRAPH_WINDOW : read_access
INTENSITY_GRAPH_WINDOW : read_access
GRAYSCALE_FACTOR : read_access
BRIGHT_FIELD_RESPONSE_CURVE : read_access
DARK_FIELD_RESPONSE_CURVE : read_access
STATE : read_access

Description: Read accessor.

RT_I.11 deleteIMAGE

Process Type: Accessor
Interface: IMAGE_ID : data_in
Description: Delete accessor, that deletes a particular instance of the object and recovers its memory.

RT_I.13 ScaleImage

Process Type: Transformation
Interface: Y_ASPECT_RATIO : data_in
SCALE_BAR.Y0 : data_in
SCALE_BAR.Y1 : data_in
OBJECT_HEIGHT_IN_MICRONS : data_in
MICRONS_PER_PIXEL : data_out
Description: Scales the image by superimposing the scale bar over an object of known height.

RT_I.14 MapPixelValue

Process Type: Transformation
Interface: PIXEL_X_POSITION : data_in
PIXEL_Y_POSITION : data_in
GRAYSCALE_FACTOR : data_in
IMAGE_DATA : data_in
PIXEL_VALUE : data_out
Description: Maps the image pixel data onto pixel values that the system graphics can display.

RT_I.15 PlotPixel

Process Type: Accessor
Interface: PIXEL_X_POSITION : data_in
PIXEL_Y_POSITION : data_in
PIXEL_VALUE : data_in
IMAGE_GRAPH_WINDOW : data_in
PLOT : data_out
Description: Plots the pixel at a position in the image graph window.

RT_I.16 GetPixelPosition

Process Type: Transformation
Interface: X0 : data_in
Y0 : data_in
X_ASPECT_RATIO : data_in
Y_ASPECT_RATIO : data_in
PIXEL_X_POSITION : data_out

PIXEL_Y_POSITION : data_out

Description: Determines where in the image graph window, a particular pixel will be plotted. Only every X Aspect Ratio'ed and Y Aspect Ratio'ed pixel in the image data gets plotted in the image graph window.

RT_I.17 DetermineFieldToLinearise

Process Type: Test

Interface: FIELD_TO_LINEARISE : data_in
BRIGHT : conditional_control_out
DARK : conditional_control_out

Description: Determines which field was selected for linearisation.

RT_I.18 DeliverPixelsinBrightRows

Process Type: Transformation

Interface: DATA_WINDOW : data_in
PIXEL : data_out

Description: Delivers the bright field pixels from the rows within the data window.

RT_I.19 LineariseBrightPixel

Process Type: Transformation

Interface: BRIGHT_FIELD_RESPONSE_CURVE : data_in
PIXEL : data_in
DATA_WINDOW : data_out

Description: Linearises the bright field pixels using the bright field response curve.

RT_I.20 LineariseDarkPixel

Process Type: Transformation

Interface: DARK_FIELD_RESPONSE_CURVE : data_in
PIXEL : data_in
DATA_WINDOW : data_out

Description: Linearises the dark field pixels using the dark field response curve.

RT_I.21 PositionandSizeDataWindow

Process Type: Transformation

Interface: DATA_WINDOW : data_in
SCAN_BAR.Y : data_in
SCAN_BAR.X0 : data_in
SCAN_BAR.X1 : data_in
IMAGE_ID : data_in
DATA_WINDOW_SIZE : data_in
DATA_WINDOW : data_out

Description: Positions the data window around the scan bar and scales it to the selected window size.

RT_I.22 AverageandReverseColumnsinDataWindow

Process Type: Transformation

Interface: DATA_WINDOW : data_in
INTENSITY : data_out

Description: Averages the intensities (in each column) of the different rows in the data window and reverses the order of the columns since the rightmost column will be at the centre (position 0) of the radial shock.

RT_I.23 PlotIntensity

Process Type: Accessor

Interface: INTENSITY : data_in
INTENSITY_GRAPH_WINDOW : data_in
GRAPH_RESOLUTION : data_in
PLOT : data_out

Description: Plots the extracted intensity in the intensity graph window.

RT_I.24 DeliverPixelsinDarkRows

Process Type: Transformation
Interface: DATA_WINDOW : data_in
PIXEL : data_out
Description: Delivers the dark field pixels from the rows within the data window.

RT_I.31 generateRT_I1:CREATE

Process Type: Event Generator
Interface: MAXIMUM_PEAKS : data_in
X_ASPECT_RATIO : data_in
Y_ASPECT_RATIO : data_in
GRAPH_RESOLUTION : data_in
GRAYSCALE_FACTOR : data_in
BRIGHT_FIELD_RESPONSE_CURVE : data_in
DARK_FIELD_RESPONSE_CURVE : data_in
RT_I1:CREATE : generated_event
Description: Event generator.

RT_I.32 generateRT_I2:LOAD_IMAGE

Process Type: Event Generator
Interface: IMAGE.IMAGE_ID : data_in
IMAGE.IMAGE_FILE : data_in
RT_I2:LOAD_IMAGE : generated_event
Description: Event generator.

RT_I.33 generateRT_UI3:SET_DISPLAY

Process Type: Event Generator
Interface: IMAGE.IMAGE_ID : data_in
RT_UI3:SET_DISPLAY : generated_event
Description: Event generator.

RT_I.34 generateRT_I4:SHOW_IMAGE

Process Type: Event Generator
Interface: IMAGE.IMAGE_ID : data_in
X0 : data_in
Y0 : data_in
X_ASPECT_RATIO : data_in
Y_ASPECT_RATIO : data_in
RT_I4:SHOW_IMAGE : generated_event
Description: Event generator.

RT_I.35 generateRT_I5:DETERMINE_SCALE

Process Type: Event Generator
Interface: IMAGE.IMAGE_ID : data_in
IMAGE.OBJECT_HEIGHT_IN_MICRONS : data_in
RT_I5:DETERMINE_SCALE : generated_event
Description: Event generator.

RT_I.36 generateRT_I6:EXTRACT_INTENSITY

Process Type: Event Generator
Interface: IMAGE.IMAGE_ID : data_in
IMAGE.DATA_WINDOW_SIZE : data_in
RT_I6:EXTRACT_INTENSITY : generated_event

Description: Event generator.

RT_I.37 generateRT_I7:LINEARISE_INTENSITY

Process Type: Event Generator
Interface: IMAGE.IMAGE_ID : data_in
IMAGE.FIELD_TO_LINEARISE : data_in
RT_I7:LINEARISE_INTENSITY : generated_event
Description: Event generator.

RT_I.38 generateRT_I8:PLOT_INTENSITY

Process Type: Event Generator
Interface: IMAGE.IMAGE_ID : data_in
RT_I8:PLOT_INTENSITY : generated_event
Description: Event generator.

RT_I.39 generateRT_I9:LOCATE_INTENSITY_PEAKS

Process Type: Event Generator
Interface: IMAGE.IMAGE_ID : data_in
RT_I9:LOCATE_INTENSITY_PEAKS : generated_event
Description: Event generator.

RT_I.40 generateRT_I10:SAVE_RAW_INTENSITY_DATA

Process Type: Event Generator
Interface: IMAGE.INTENSITY_RAW_FILE : data_in
IMAGE.IMAGE_ID : data_in
RT_I10:SAVE_RAW_INTENSITY_DATA : generated_event
Description: Event generator.

RT_I.41 generateRT_I11:SAVE_INTENSITY_PEAKS

Process Type: Event Generator
Interface: IMAGE.INTENSITY_PEAKS_FILE : data_in
IMAGE.IMAGE_ID : data_in
RT_I11:SAVE_INTENSITY_PEAKS : generated_event
Description: Event generator.

RT_I.42 generateRT_I12:CLEAR_DISPLAY

Process Type: Event Generator
Interface: IMAGE.IMAGE_ID : data_in
RT_I12:CLEAR_DISPLAY : generated_event
Description: Event generator.

RT_I.43 generateRT_I13:DELETE

Process Type: Event Generator
Interface: IMAGE.IMAGE_ID : data_in
RT_I13:DELETE : generated_event
Description: Event generator.

RT_M.1 GetMenuSelection

Process Type: Test
Interface: MENU_SELECTION : data_in
NAME : data_in
ABOUT : conditional_control_out
CALCULATOR : conditional_control_out
HELP : conditional_control_out
DOS_SHELL : conditional_control_out

EXIT : conditional_control_out
FIRE_RAYS : conditional_control_out
SAVE_PARAMETERS : conditional_control_out
LOAD_PARAMETERS : conditional_control_out
SAVE_RAW_DENSITIES : conditional_control_out
SAVE_RAW_INTENSITIES : conditional_control_out
DISPLAY_SHOCK : conditional_control_out
RUN_MACRO_FILE : conditional_control_out
LOAD_IMAGE : conditional_control_out
DETERMINE_SCALE : conditional_control_out
EXTRACT_INTENSITY : conditional_control_out
SAVE_RAW_INTENSITIES : conditional_control_out
SAVE_INTENSITY_PEAKS : conditional_control_out
DISPLAY_IMAGE : conditional_control_out

Description: Returns any of the menu selections from all the top level menus.

RT_M.3 SetAttributes

Process Type: Accessor
NAME : data_in
MENU_BAR : data_in
HOTKEY_SEQUENCE : data_in
HIGHLIGHT_KEY : data_in
COMMAND : data_in
UI_ID : data_in
MENU_SELECTION : data_in
MENU_BAR : write_access
HOTKEY_SEQUENCE : write_access
HIGHLIGHT_KEY : write_access
COMMAND : write_access
UI_ID : write_access
MENU_SELECTION : write_access

Description: Write accessor.

RT_M.10 GetAttributes

Process Type: Accessor
NAME : data_in
MENU_BAR : data_out
HOTKEY_SEQUENCE : data_out
HIGHLIGHT_KEY : data_out
COMMAND : data_out
UI_ID : data_out
MENU_SELECTION : data_out
MENU_BAR : read_access
HOTKEY_SEQUENCE : read_access
HIGHLIGHT_KEY : read_access
COMMAND : read_access
UI_ID : read_access
MENU_SELECTION : read_access

Description: Read accessor.

RT_NB.1 createSCAN_BAR

Process Type: Accessor
Interface: Y : data_in
X0 : data_in
X1 : data_in

SCAN_BAR_ID : data_out
Y : write_access
X0 : write_access
X1 : write_access
STATE : write_access

Description: Create accessor.

RT_NB.2 GetCurrentKeypress

Process Type: Accessor

Interface: SCAN_BAR_ID : data_in
CURRENT_KEYPRESS : data_in
CURRENT_KEYPRESS : data_out
SCAN_BAR.CURRENT_KEYPRESS : write_access

Description: Returns the operator keypress.

RT_NB.3 SetAttributes

Process Type: Accessor

Interface: SCAN_BAR_ID : data_in
X0 : data_in
X1 : data_in
Y : data_in
CURRENT_KEYPRESS : data_in
STATE : data_in
X0 : write_access
X1 : write_access
Y : write_access
CURRENT_KEYPRESS : write_access
STATE : write_access

Description: Write accessor.

RT_NB.4 ExamineCurrentKeypress

Process Type: Test

Interface: CURRENT_KEYPRESS : data_in
UP_ARROW : conditional_control_out
DOWN_ARROW : conditional_control_out
LEFT_ARROW : conditional_control_out
RIGHT_ARROW : conditional_control_out
PAGE_UP : conditional_control_out
PAGE_DOWN : conditional_control_out
HOME : conditional_control_out
END : conditional_control_out
DELETE : conditional_control_out
INSERT : conditional_control_out
CARRIAGE_RETURN : conditional_control_out

Description: Determines which key the operator pressed.

RT_NB.10 GetAttributes

Process Type: Accessor

Interface: SCAN_BAR_ID : data_in
X0 : data_out
X1 : data_out
Y : data_out
CURRENT_KEYPRESS : data_out
STATE : data_out
X0 : read_access

X1 : read_access
Y : read_access
CURRENT_KEYPRESS : read_access
STATE : read_access
Description: Read accessor.

RT_NB.31 generateRT_NB1:CREATE
Process Type: Event Generator
Interface: SCAN_BAR.X0 : data_in
SCAN_BAR.X1 : data_in
SCAN_BAR.Y : data_in
RT_NB1:CREATE : generated_event
Description: Event generator.

RT_NB.32 generateRT_NB2:SCAN
Process Type: Event Generator
Interface: SCAN_BAR.SCAN_BAR_ID : data_in
RT_NB2:SCAN : generated_event
Description: Event generator.

RT_NB.33 generateRT_NB3:DONE
Process Type: Event Generator
Interface: SCAN_BAR_ID : data_in
RT_NB3:DONE : generated_event
Description: Event generator.

RT_R.1 DetermineFirstFreeOPL
Process Type: Accessor
Interface: DENSITY_PARAMETERS.FRONT : data_in
INITIAL_POSITION : data_in
FIRST_FREE_OPTICAL_PATH_LENGTH : write_access
Description: Determines the first free optical path length travelled by the ray before entering the shock according to equation (2.14).

RT_R.2 CurrentSubshell
Process Type: Transformation
Interface: INITIAL_POSITION : data_in
RADIUS : data_out
Description: Determines the current subshell being traversed from the ray position.

RT_R.3 SetAttributes
Process Type: Accessor
INITIAL_POSITION : data_in
SHOCK_ID : data_in
LIMITS_ID : data_in
FINAL_POSITION : data_in
FIRST_FREE_OPTICAL_PATH_LENGTH : data_in
OPTICAL_PATH_LENGTH_IN_SHOCK : data_in
SECOND_FREE_OPTICAL_PATH_LENGTH : data_in
TOTAL_OPTICAL_PATH_LENGTH : data_in
WAVELENGTH : data_in
PHASE : data_in
SHOCK_ID : write_access
LIMITS_ID : write_access
FINAL_POSITION : write_access

FIRST_FREE_OPTICAL_PATH_LENGTH : write_access
OPTICAL_PATH_LENGTH_IN_SHOCK : write_access
SECOND_FREE_OPTICAL_PATH_LENGTH : write_access
TOTAL_OPTICAL_PATH_LENGTH : write_access
WAVELENGTH : write_access
PHASE : write_access

Description: Write accessor.

RT_R.4 IncrementOPLinShock

Process Type: Accessor

Interface: COMPUTATIONAL_SUBSHELL.OPTICAL_PATH_LENGTH : data_in
INITIAL_POSITION : data_in
OPTICAL_PATH_LENGTH_IN_SHOCK : read_access
OPTICAL_PATH_LENGTH_IN_SHOCK : write_access

Description: Increments the optical path length within the entire shock as the ray traverses each subshell in it, following equation (2.13).

RT_R.5 DetermineSecondFreeOPL

Process Type: Accessor

Interface: SHOCK.DISTANCE_FROM_IMAGE_PLANE : data_in
INITIAL_POSITION : data_in
SECOND_FREE_OPTICAL_PATH_LENGTH : write_access

Description: Determines the second free optical path length travelled by the ray on exiting the shock according to equation (2.15).

RT_R.6 DetermineTotalOPL

Process Type: Accessor

Interface: INITIAL_POSITION : data_in
TOTAL_OPTICAL_PATH_LENGTH : data_out
FIRST_FREE_OPTICAL_PATH_LENGTH : read_access
SECOND_FREE_OPTICAL_PATH_LENGTH : read_access
OPTICAL_PATH_LENGTH_IN_SHOCK : read_access
TOTAL_OPTICAL_PATH_LENGTH : write_access

Description: Determines the total optical path length travelled by the ray according to equation (2.16).

RT_R.7 DeterminePhase

Process Type: Accessor

Interface: WAVELENGTH : data_in
TOTAL_OPTICAL_PATH_LENGTH : data_in
INITIAL_POSITION : data_in
PHASE : write_access

Description: Determines the phase of the ray at the image plane using equation (2.17).

RT_R.8 DetermineFinalPosition

Process Type: Accessor

Interface: SHOCK.DISTANCE_FROM_IMAGE_PLANE : data_in
INITIAL_POSITION : data_in
FINAL_POSITION : write_access

Description: Determines the position (height) of the ray on the image plane following the derivation of equation (2.15).

RT_R.10 GetAttributes

Process Type: Accessor

Interface: INITIAL_POSITION : data_in
SHOCK_ID : data_out
LIMITS_ID : data_out

	FINAL_POSITION : data_out
	FIRST_FREE_OPTICAL_PATH_LENGTH : data_out
	OPTICAL_PATH_LENGTH_IN_SHOCK : data_out
	SECOND_FREE_OPTICAL_PATH_LENGTH : data_out
	TOTAL_OPTICAL_PATH_LENGTH : data_out
	WAVELENGTH : data_out
	PHASE : data_out
	SHOCK_ID : read_access
	LIMITS_ID : read_access
	FINAL_POSITION : read_access
	FIRST_FREE_OPTICAL_PATH_LENGTH : read_access
	OPTICAL_PATH_LENGTH_IN_SHOCK : read_access
	SECOND_FREE_OPTICAL_PATH_LENGTH : read_access
	TOTAL_OPTICAL_PATH_LENGTH : read_access
	WAVELENGTH : read_access
	PHASE : read_access
Description:	Read accessor.

RT_R.31 generateRT_R1:TRACE

Process Type:	Event Generator
Interface:	RAY.INITIAL_POSITION : data_in
	RT_R1:TRACE : generated_event
Description:	Event generator.

RT_RL.1 createRAYTRACE_LIMITS

Process Type:	Accessor
Interface:	SHOCK_ID : data_in
	SHOCK_ID : write_access
	STATE : write_access
Description:	Create accessor.

RT_RL.2 ConstrainLimits

Process Type:	Transformation
Interface:	DENSITY_PARAMETERS.VACUUM : data_in
	DENSITY_PARAMETERS.FRONT : data_in
	FIRE_RAYS_FROM : data_in
	FIRE_RAYS_TO : data_in
	FIRE_RAYS_FROM : data_out
	FIRE_RAYS_TO : data_out
Description:	Ensures that the fired rays will lie within the shock.

RT_RL.3 SetAttributes

Process Type:	Accessor
Interface:	RAYTRACE_LIMITS_ID : data_in
	SHOCK_ID : data_in
	FIRE_RAYS_FROM : data_in
	FIRE_RAYS_TO : data_in
	CURRENT_RAY : data_in
	STATE : data_in
	SHOCK_ID : write_access
	FIRE_RAYS_FROM : write_access
	FIRE_RAYS_TO : write_access
	CURRENT_RAY : write_access
	STATE : write_access
Description:	Write accessor.

RT_RL.10 GetAttributes
Process Type: Accessor
Interface: RAYTRACE_LIMITS_ID : data_in
SHOCK_ID : data_out
CURRENT_RAY : data_out
FIRE_RAYS_FROM : data_out
FIRE_RAYS_TO : data_out
STATE : data_out
SHOCK_ID : read_access
FIRE_RAYS_FROM : read_access
FIRE_RAYS_TO : read_access
CURRENT_RAY : read_access
STATE : read_access
Description: Read accessor.

RT_RL.31 generateRT_RL1:SET
Process Type: Event Generator
Interface: RAYTRACE_LIMITS.FIRE_RAYS_FROM : data_in
RAYTRACE_LIMITS.FIRE_RAYS_TO : data_in
RAYTRACE_LIMITS.RAYTRACE_LIMITS_ID : data_in
RT_RL1:SET : generated_event
Description: Event generator.

RT_RL.32 generateRT_RL2:CREATE
Process Type: Event Generator
Interface: SHOCK_ID : data_in
RT_RL2:CREATE : generated_event
Description: Event generator.

RT_RST.1 createRAY_SUBSHELL_TRAVERSAL
Process Type: Accessor
Interface: RADIUS : data_in
RAY_ID : data_in
RADIUS : write_access
RAY_ID : write_access
Description: Create accessor.

RT_RST.3 SetAttributes
Process Type: Accessor
RADIUS : data_in
RAY_ID : data_in
RADIUS : write_access
RAY_ID : write_access
Description: Write accessor.

RT_RST.10 GetAttributes
Process Type: Accessor
RADIUS : data_in
RAY_ID : data_out
RADIUS : read_access
RAY_ID : read_access
Description: Read accessor.

RT_S.1 createSHOCK
Process Type: Accessor

Interface: SHOCK_EXTENT : data_in
SCREEN_EXTENT : data_in
GRAPH_RESOLUTION : data_in
DISTANCE_FROM_IMAGE_PLANE : data_in
ELECTRODE_SEPARATION : data_in
SHOCK_ID : data_out
SHOCK_EXTENT : write_access
SCREEN_EXTENT : write_access
GRAPH_RESOLUTION : write_access
DISTANCE_FROM_IMAGE_PLANE : write_access
ELECTRODE_SEPARATION : write_access
EXTREMA : write_access
STATE : write_access

Description: Create accessor.

RT_S.2 InitialiseDensityAttributes

Process Type: Accessor

Interface: SHOCK_ID : data_in
SHOCK_EXTENT : data_in
DENSITY_PROFILE : write_access
REFRACTIVE_INDEX : write_access

Description: Sets the air density prior to shock formation to atmospheric (1) and the refractive index to 0.

RT_S.3 SetAttributes

Process Type: Accessor

Interface: SHOCK_ID : data_in
DENSITY_PARAMETERS : data_in
INTENSITY_PARAMETERS : data_in
RAYTRACE_LIMITS : data_in
INTENSITY_PATTERN : data_in
DENSITY_PROFILE : data_in
REFRACTIVE_INDEX : data_in
COS_COMPONENT_OF_ELECTRIC_FIELD : data_in
SIN_COMPONENT_OF_ELECTRIC_FIELD : data_in
ELECTRIC_FIELD : data_in
EXTREMA : data_in
PARAMETER_FILE : data_in
DENSITY_FILE : data_in
INTENSITY_FILE : data_in
DENSITY_GRAPH_WINDOW : data_in
INTENSITY_GRAPH_WINDOW : data_in
PARAMETER_WINDOW : data_in
GRAPH_RESOLUTION : data_in
SHOCK_EXTENT : data_in
SCREEN_EXTENT : data_in
DISTANCE_FROM_IMAGE_PLANE : data_in
ELECTRODE_SEPARATION : data_in
STATE : data_in
DENSITY_PARAMETERS : write_access
INTENSITY_PARAMETERS : write_access
RAYTRACE_LIMITS : write_access
INTENSITY_PATTERN : write_access
DENSITY_PROFILE : write_access
REFRACTIVE_INDEX : write_access
COS_COMPONENT_OF_ELECTRIC_FIELD : write_access

SIN_COMPONENT_OF_ELECTRIC_FIELD : write_access
ELECTRIC_FIELD : write_access
EXTREMA : write_access
PARAMETER_FILE : write_access
DENSITY_FILE : write_access
INTENSITY_FILE : write_access
DENSITY_GRAPH_WINDOW : write_access
INTENSITY_GRAPH_WINDOW : write_access
PARAMETER_WINDOW : write_access
GRAPH_RESOLUTION : write_access
SHOCK_EXTENT : write_access
SCREEN_EXTENT : write_access
DISTANCE_FROM_IMAGE_PLANE : write_access
ELECTRODE_SEPARATION : write_access
STATE : write_access

Description: Write accessor.

RT_S.4 InitialiseIntensityAttributes

Process Type: Accessor

Interface:
SHOCK_ID : data_in
SCREEN_EXTENT : data_in
INTENSITY_PATTERN : write_access
INTENSITY_PARAMETERS : write_access
COS_COMPONENT_OF_ELECTRIC_FIELD : write_access
SIN_COMPONENT_OF_ELECTRIC_FIELD : write_access

Description: Initialises the intensity pattern prior to raytracing to 0, sets the sin component to 0 and the cos component to 1.

RT_S.5 ReadParameterFile

Process Type: Accessor

Interface:
PARAMETER_FILE : data_in
DENSITY_PARAMETERS : data_in
EXTREMA : data_in
INTENSITY_PARAMETERS : data_in
DENSITY_PARAMETERS : data_out
EXTREMA : data_out
INTENSITY_PARAMETERS : data_out

Description: Reads in a parameter file storing density and intensity parameters and sets the parameters to those values.

RT_S.6 InterpolateIntensityParameters

Process Type: Accessor

Interface:
INTENSITY_PARAMETERS : data_in
SHOCK_ID : data_in
INTENSITY_PATTERN : write_access

Description: Interpolates the intensity parameters (peaks) using the cubic spline technique to produce an intensity pattern of micron resolution.

RT_S.7 InitialiseGraphics

Process Type: Accessor

Interface:
PARAMETER_WINDOW : data_in
DENSITY_GRAPH_WINDOW : data_in
INTENSITY_GRAPH_WINDOW : data_in
INITIALISE : data_out

Description: Initialises the graphics system.

RT_S.8 IncrementElectricField

Process Type: Accessor

Interface: RAY.PHASE : data_in
SHOCK_ID : data_in
SIN_COMPONENT_OF_ELECTRIC_FIELD : read_access
COS_COMPONENT_OF_ELECTRIC_FIELD : read_access
SIN_COMPONENT_OF_ELECTRIC_FIELD : write_access
COS_COMPONENT_OF_ELECTRIC_FIELD : write_access

Description: Calculates, using the phase of the ray, the sin and cos components of the electric field that it produces at the image plane and adds these values to those already calculated for that point on the image plane. In this way the resultant electric field is constructed.

RT_S.9 DetermineLightIntensity

Process Type: Accessor

Interface: SHOCK_ID : data_in
SIN_COMPONENT_OF_ELECTRIC_FIELD : read_access
COS_COMPONENT_OF_ELECTRIC_FIELD : read_access
INTENSITY_PATTERN : write_access

Description: Calculates the light intensity at every point on the image plane (the intensity pattern) from the sin and cos components of the electric field using equation (2.18).

RT_S.10 GetAttributes

Process Type: Accessor

Interface: SHOCK_ID : data_in
DENSITY_PARAMETERS : data_out
INTENSITY_PARAMETERS : data_out
RAYTRACE_LIMITS : data_out
INTENSITY_PATTERN : data_out
DENSITY_PROFILE : data_out
REFRACTIVE_INDEX : data_out
COS_COMPONENT_OF_ELECTRIC_FIELD : data_out
SIN_COMPONENT_OF_ELECTRIC_FIELD : data_out
ELECTRIC_FIELD : data_out
EXTREMA : data_out
PARAMETER_FILE : data_out
DENSITY_FILE : data_out
INTENSITY_FILE : data_out
DENSITY_GRAPH_WINDOW : data_out
INTENSITY_GRAPH_WINDOW : data_out
PARAMETER_WINDOW : data_out
GRAPH_RESOLUTION : data_out
SHOCK_EXTENT : data_out
SCREEN_EXTENT : data_out
DISTANCE_FROM_IMAGE_PLANE : data_out
ELECTRODE_SEPARATION : data_out
STATE : data_out
DENSITY_PARAMETERS : read_access
INTENSITY_PARAMETERS : read_access
RAYTRACE_LIMITS : read_access
INTENSITY_PATTERN : read_access
DENSITY_PROFILE : read_access
REFRACTIVE_INDEX : read_access
COS_COMPONENT_OF_ELECTRIC_FIELD : read_access
SIN_COMPONENT_OF_ELECTRIC_FIELD : read_access
ELECTRIC_FIELD : read_access

EXTREMA : read_access
PARAMETER_FILE : read_access
DENSITY_FILE : read_access
INTENSITY_FILE : read_access
DENSITY_GRAPH_WINDOW : read_access
INTENSITY_GRAPH_WINDOW : read_access
PARAMETER_WINDOW : read_access
GRAPH_RESOLUTION : read_access
SHOCK_EXTENT : read_access
SCREEN_EXTENT : read_access
DISTANCE_FROM_IMAGE_PLANE : read_access
ELECTRODE_SEPARATION : read_access
STATE : read_access

Description: Read accessor.

RT_S.11 SmoothandLimitLightIntensity

Process Type: Accessor

Interface: SHOCK_ID : data_in
INTENSITY_PATTERN : read_access
INTENSITY_PATTERN : write_access

Description: Smooths the intensity pattern over 50 micron bins and restricts the intensities to a range of 0..10.

RT_S.12 PlotDensityProfile

Process Type: Accessor

Interface: DENSITY_GRAPH_WINDOW : data_in
DENSITY_PROFILE : data_in
GRAPH_RESOLUTION : data_in
PLOT : data_out

Description: Plots every graph resolution'th point of the density profile in the density graph window.

RT_S.13 PlotIntensityPattern

Process Type: Accessor

Interface: INTENSITY_GRAPH_WINDOW : data_in
INTENSITY_PATTERN : data_in
GRAPH_RESOLUTION : data_in
PLOT : data_out

Description: Plots every graph resolution'th point of the intensity pattern in the intensity graph window.

RT_S.14 DisplayDensityParameters

Process Type: Accessor

Interface: PARAMETER_WINDOW : data_in
VACUUM : data_in
TAIL : data_in
SADDLE : data_in
REAR : data_in
FRONT : data_in
DISPLAY : data_out

Description: Displays the density parameters in the parameter window.

RT_S.15 SaveDensityProfile

Process Type: Accessor

Interface: DENSITY_PROFILE : data_in
DENSITY_FILE : data_in
DENSITY_PROFILE : data_out

Description: Saves the density profile to the density file in ASCII format.

RT_S.16 SaveIntensityPattern

Process Type: Accessor
Interface: INTENSITY_PATTERN : data_in
INTENSITY_FILE : data_in
INTENSITY_PATTERN : data_out
Description: Saves the intensity pattern to the intensity file in ASCII format.

RT_S.17 ExtractIntensityPeaks

Process Type: Transformation
Interface: INTENSITY_PATTERN : data_in
INTENSITY_PARAMETERS : data_out
EXTREMA : data_out
Description: Extracts the extrema (peaks and troughs) from the intensity pattern.

RT_S.18 SaveDensityParameters

Process Type: Accessor
Interface: VACUUM : data_in
TAIL : data_in
SADDLE : data_in
REAR : data_in
FRONT : data_in
INTERPOLATION_METHOD : data_in
PARAMETER_FILE : data_in
VACUUM : data_out
TAIL : data_out
SADDLE : data_out
REAR : data_out
FRONT : data_out
INTERPOLATION_METHOD : data_out
Description: Saves the density parameters to the parameter file in ASCII format.

RT_S.19 SaveIntensityParameters

Process Type: Accessor
Interface: INTENSITY_PARAMETERS : data_in
EXTREMA : data_in
PARAMETER_FILE : data_in
EXTREMA : data_out
INTENSITY_PARAMETERS : data_out
Description: Saves the intensity parameters (extrema) to the parameter file in ASCII format.

RT_S.20 CloseGraphics

Process Type: Accessor
Interface: SHOCK_ID : data_in
CLOSE : data_out
Description: Shuts down the graphics system and returns to the text-based menu system.

RT_S.21 deleteSHOCK

Process Type: Accessor
Interface: SHOCK_ID : data_in
Description: Delete accessor that recovers the memory consumed by the object.

RT_S.31 generateRT_S1:CREATE

Process Type: Event Generator

Interface: SHOCK_EXTENT : data_in
SCREEN_EXTENT : data_in
GRAPH_RESOLUTION : data_in
DISTANCE_FROM_IMAGE_PLANE : data_in
ELECTRODE_SEPARATION : data_in
RT_S1:CREATE : generated_event
Description: Event generator.

RT_S.32 generateRT_I2:LOAD_PARAMETERS
Process Type: Event Generator
Interface: SHOCK.SHOCK_ID : data_in
SHOCK.PARAMETER_FILE : data_in
RT_I2:LOAD_PARAMETERS : generated_event
Description: Event generator.

RT_S.33 generateRT_S3:FIRE_RAYS
Process Type: Event Generator
Interface: SHOCK.SHOCK_ID : data_in
RT_S3:FIRE_RAYS : generated_event
Description: Event generator.

RT_S.34 generateRT_S4:SET_DISPLAY
Process Type: Event Generator
Interface: SHOCK.SHOCK_ID : data_in
RT_S4:SET_DISPLAY : generated_event
Description: Event generator.

RT_S.35 generateRT_S5:PLOT_DENSITY_PROFILE
Process Type: Event Generator
Interface: SHOCK.SHOCK_ID : data_in
RT_S5:PLOT_DENSITY_PROFILE : generated_event
Description: Event generator.

RT_S.36 generateRT_S6:DISPLAY_PARAMETERS
Process Type: Event Generator
Interface: SHOCK.SHOCK_ID : data_in
RT_S6:DISPLAY_PARAMETERS : generated_event
Description: Event generator.

RT_S.37 generateRT_S7:PLOT_INTENSITY_PATTERN
Process Type: Event Generator
Interface: SHOCK.SHOCK_ID : data_in
RT_S7:PLOT_INTENSITY_PATTERN : generated_event
Description: Event generator.

RT_S.38 generateRT_S8:CLEAR_DISPLAY
Process Type: Event Generator
Interface: SHOCK.SHOCK_ID : data_in
RT_S8:CLEAR_DISPLAY : generated_event
Description: Event generator.

RT_S.39 generateRT_S9:SAVE_DENSITY_PROFILE
Process Type: Event Generator
Interface: SHOCK.DENSITY_FILE : data_in
SHOCK.SHOCK_ID : data_in

	RT_S9:SAVE_DENSITY_PROFILE : generated_event
Description:	Event generator.

RT_S.40 generateRT_S10:SAVE_INTENSITY_PATTERN

Process Type:	Event Generator
Interface:	SHOCK.INTENSITY_FILE : data_in
	SHOCK.SHOCK_ID : data_in
	RT_S10:SAVE_INTENSITY_PATTERN : generated_event
Description:	Event generator.

RT_S.41 generateRT_S11:SAVE_PARAMETERS

Process Type:	Event Generator
Interface:	SHOCK.PARAMETER_FILE : data_in
	SHOCK.SHOCK_ID : data_in
	RT_S11:SAVE_PARAMETERS : generated_event
Description:	Event generator.

RT_S.42 generateRT_S12:DONE

Process Type:	Event Generator
Interface:	SHOCK_ID : data_in
	RT_S12:DONE : generated_event
Description:	Event generator.

RT_S.43 generateRT_S13:DELETE

Process Type:	Event Generator
Interface:	SHOCK.SHOCK_ID : data_in
	RT_S13:DELETE : generated_event
Description:	Event generator.

RT_SD.1 createSET_DENSITY

Process Type:	Accessor
Interface:	DESKTOP_INTERFACE : data_in
	TITLE : data_in
	CLOSE_ICON : data_in
	OK_BUTTON : data_in
	CANCEL_BUTTON : data_in
	HELP_BUTTON : data_in
	DESKTOP_INTERFACE : write_access
	TITLE : write_access
	CLOSE_ICON : write_access
	OK_BUTTON : write_access
	CANCEL_BUTTON : write_access
	HELP_BUTTON : write_access
	STATE : write_access
Description:	Create accessor.

RT_SD.2 GetDataForFields

Process Type:	Transformation
Interface:	DENSITY_PARAMETERS.DENSITY_PARAMETERS_ID : data_in
	DENSITY_PARAMETERS.VACUUM : data_in
	DENSITY_PARAMETERS.TAIL : data_in
	DENSITY_PARAMETERS.SADDLE : data_in
	DENSITY_PARAMETERS.REAR : data_in
	DENSITY_PARAMETERS.FRONT : data_in
	DENSITY_PARAMETERS.INTERPOLATION_METHOD : data_in

RAYTRACE_LIMITS.FIRE_RAYS_FROM : data_in
RAYTRACE_LIMITS.FIRE_RAYS_TO : data_in
INTERPOLATION_RADIO_BUTTON_CLUSTER : data_out
QUANTISED_POSITION_VACUUM_INPUT_LINE : data_out
QUANTISED_POSITION_TAIL_INPUT_LINE : data_out
QUANTISED_POSITION_SADDLE_INPUT_LINE : data_out
QUANTISED_POSITION_REAR_INPUT_LINE : data_out
QUANTISED_POSITION_FRONT_INPUT_LINE : data_out
CONTINUOUS_DENSITY_VACUUM_INPUT_LINE : data_out
CONTINUOUS_DENSITY_TAIL_INPUT_LINE : data_out
CONTINUOUS_DENSITY_SADDLE_INPUT_LINE : data_out
CONTINUOUS_DENSITY_REAR_INPUT_LINE : data_out
CONTINUOUS_DENSITY_FRONT_INPUT_LINE : data_out
FIRE_RAYS_FROM_INPUT_LINE : data_out
FIRE_RAYS_TO_INPUT_LINE : data_out
FIRE_RAYS_CHECK_BOX : data_out

Description: Retrieves the latest data and displays it in the dialog box fields.

RT_SD.3 SetAttributes

Process Type: Accessor

Interface: DIALOG_ID : data_in
INTERPOLATION_RADIO_BUTTON_CLUSTER : data_in
QUANTISED_POSITION_VACUUM_INPUT_LINE : data_in
QUANTISED_POSITION_TAIL_INPUT_LINE : data_in
QUANTISED_POSITION_SADDLE_INPUT_LINE : data_in
QUANTISED_POSITION_REAR_INPUT_LINE : data_in
QUANTISED_POSITION_FRONT_INPUT_LINE : data_in
CONTINUOUS_DENSITY_VACUUM_INPUT_LINE : data_in
CONTINUOUS_DENSITY_TAIL_INPUT_LINE : data_in
CONTINUOUS_DENSITY_SADDLE_INPUT_LINE : data_in
CONTINUOUS_DENSITY_REAR_INPUT_LINE : data_in
CONTINUOUS_DENSITY_FRONT_INPUT_LINE : data_in
FIRE_RAYS_FROM_INPUT_LINE : data_in
FIRE_RAYS_TO_INPUT_LINE : data_in
FIRE_RAYS_CHECK_BOX : data_in
STATE : data_in
INTERPOLATION_RADIO_BUTTON_CLUSTER : write_access
QUANTISED_POSITION_VACUUM_INPUT_LINE : write_access
QUANTISED_POSITION_TAIL_INPUT_LINE : write_access
QUANTISED_POSITION_SADDLE_INPUT_LINE : write_access
QUANTISED_POSITION_REAR_INPUT_LINE : write_access
QUANTISED_POSITION_FRONT_INPUT_LINE : write_access
CONTINUOUS_DENSITY_VACUUM_INPUT_LINE : write_access
CONTINUOUS_DENSITY_TAIL_INPUT_LINE : write_access
CONTINUOUS_DENSITY_SADDLE_INPUT_LINE : write_access
CONTINUOUS_DENSITY_REAR_INPUT_LINE : write_access
CONTINUOUS_DENSITY_FRONT_INPUT_LINE : write_access
FIRE_RAYS_FROM_INPUT_LINE : write_access
FIRE_RAYS_TO_INPUT_LINE : write_access
FIRE_RAYS_CHECK_BOX : write_access
STATE : write_access

Description: Write accessor.

RT_SD.5 GetDataFromFields

Process Type: Accessor

Interface: DIALOG_ID : data_in
INTERPOLATION_RADIO_BUTTON_CLUSTER : data_in
QUANTISED_POSITION_VACUUM_INPUT_LINE : data_in
QUANTISED_POSITION_TAIL_INPUT_LINE : data_in
QUANTISED_POSITION_SADDLE_INPUT_LINE : data_in
QUANTISED_POSITION_REAR_INPUT_LINE : data_in
QUANTISED_POSITION_FRONT_INPUT_LINE : data_in
CONTINUOUS_DENSITY_VACUUM_INPUT_LINE : data_in
CONTINUOUS_DENSITY_TAIL_INPUT_LINE : data_in
CONTINUOUS_DENSITY_SADDLE_INPUT_LINE : data_in
CONTINUOUS_DENSITY_REAR_INPUT_LINE : data_in
CONTINUOUS_DENSITY_FRONT_INPUT_LINE : data_in
FIRE_RAYS_FROM_INPUT_LINE : data_in
FIRE_RAYS_TO_INPUT_LINE : data_in
FIRE_RAYS_CHECK_BOX : data_in
INTERPOLATION_RADIO_BUTTON_CLUSTER : data_out
QUANTISED_POSITION_VACUUM_INPUT_LINE : data_out
QUANTISED_POSITION_TAIL_INPUT_LINE : data_out
QUANTISED_POSITION_SADDLE_INPUT_LINE : data_out
QUANTISED_POSITION_REAR_INPUT_LINE : data_out
QUANTISED_POSITION_FRONT_INPUT_LINE : data_out
CONTINUOUS_DENSITY_VACUUM_INPUT_LINE : data_out
CONTINUOUS_DENSITY_TAIL_INPUT_LINE : data_out
CONTINUOUS_DENSITY_SADDLE_INPUT_LINE : data_out
CONTINUOUS_DENSITY_REAR_INPUT_LINE : data_out
CONTINUOUS_DENSITY_FRONT_INPUT_LINE : data_out
FIRE_RAYS_FROM_INPUT_LINE : data_out
FIRE_RAYS_TO_INPUT_LINE : data_out
FIRE_RAYS_CHECK_BOX : data_out

Description: Retrieves the latest (user-modified) data from the dialog fields.

RT_SD.6 CheckDataFromFields

Process Type: Test

Interface: INTERPOLATION_RADIO_BUTTON_CLUSTER : data_in
QUANTISED_POSITION_VACUUM_INPUT_LINE : data_in
QUANTISED_POSITION_TAIL_INPUT_LINE : data_in
QUANTISED_POSITION_SADDLE_INPUT_LINE : data_in
QUANTISED_POSITION_REAR_INPUT_LINE : data_in
QUANTISED_POSITION_FRONT_INPUT_LINE : data_in
CONTINUOUS_DENSITY_VACUUM_INPUT_LINE : data_in
CONTINUOUS_DENSITY_TAIL_INPUT_LINE : data_in
CONTINUOUS_DENSITY_SADDLE_INPUT_LINE : data_in
CONTINUOUS_DENSITY_REAR_INPUT_LINE : data_in
CONTINUOUS_DENSITY_FRONT_INPUT_LINE : data_in
FIRE_RAYS_FROM_INPUT_LINE : data_in
FIRE_RAYS_TO_INPUT_LINE : data_in
FIRE_RAYS_CHECK_BOX : data_in
OK : conditional_control_out
NOT_OK : conditional_control_out

Description: Checks if the data retrieved from the dialog box fields is within bounds and monotonically increasing in position.

RT_SD.10 GetAttributes

Process Type: Accessor

Interface: DIALOG_ID : data_in
SET_DENSITY.INTERPOLATION_RADIO_BUTTON_CLUSTER : data_out

SET_DENSITY.QUANTISED_POSITION_VACUUM_INPUT_LINE : data_out
SET_DENSITY.QUANTISED_POSITION_TAIL_INPUT_LINE : data_out
SET_DENSITY.QUANTISED_POSITION_SADDLE_INPUT_LINE : data_out
SET_DENSITY.QUANTISED_POSITION_REAR_INPUT_LINE : data_out
SET_DENSITY.QUANTISED_POSITION_FRONT_INPUT_LINE : data_out
SET_DENSITY.CONTINUOUS_DENSITY_VACUUM_INPUT_LINE : data_out
SET_DENSITY.CONTINUOUS_DENSITY_TAIL_INPUT_LINE : data_out
SET_DENSITY.CONTINUOUS_DENSITY_SADDLE_INPUT_LINE : data_out
SET_DENSITY.CONTINUOUS_DENSITY_REAR_INPUT_LINE : data_out
SET_DENSITY.CONTINUOUS_DENSITY_FRONT_INPUT_LINE : data_out
SET_DENSITY.FIRE_RAYS_FROM_INPUT_LINE : data_out
SET_DENSITY.FIRE_RAYS_TO_INPUT_LINE : data_out
SET_DENSITY.FIRE_RAYS_CHECK_BOX : data_out
STATE : data_out
INTERPOLATION_RADIO_BUTTON_CLUSTER : read_access
QUANTISED_POSITION_VACUUM_INPUT_LINE : read_access
QUANTISED_POSITION_TAIL_INPUT_LINE : read_access
QUANTISED_POSITION_SADDLE_INPUT_LINE : read_access
QUANTISED_POSITION_REAR_INPUT_LINE : read_access
QUANTISED_POSITION_FRONT_INPUT_LINE : read_access
CONTINUOUS_DENSITY_VACUUM_INPUT_LINE : read_access
CONTINUOUS_DENSITY_TAIL_INPUT_LINE : read_access
CONTINUOUS_DENSITY_SADDLE_INPUT_LINE : read_access
CONTINUOUS_DENSITY_REAR_INPUT_LINE : read_access
CONTINUOUS_DENSITY_FRONT_INPUT_LINE : read_access
FIRE_RAYS_FROM_INPUT_LINE : read_access
FIRE_RAYS_TO_INPUT_LINE : read_access
FIRE_RAYS_CHECK_BOX : read_access
STATE : read_access

Description: Read Accessor.

RT_SD.31 generateRT_SD1:OPEN

Process Type: Event Generator
Interface: DIALOG_ID : data_in
RT_SD1:OPEN : generated_event
Description: Event generator.

RT_SD.32 generateRT_SD2:ACTION

Process Type: Event Generator
Interface: DIALOG_ID : data_in
RT_SD2:ACTION : generated_event
Description: Event generator.

RT_SD.33 generateRT_SD3:CLOSE

Process Type: Event Generator
Interface: DIALOG_ID : data_in
RT_SD3:CLOSE : generated_event
Description: Event generator.

RT_TM.1 OpenHelp

Process Type: Accessor
Interface: UI_ID : data_in
OPEN_CONTEXT_SENSITIVE_HELP : data_out
Description: Opens an appropriate context sensitive help dialog.

RT_UI.1 createUSER_INTERFACE

Process Type: Accessor

Interface:
MACRO_FILE : data_in
IMAGE_FILE : data_in
PARAMETER_FILE : data_in
DENSITY_FILE : data_in
INTENSITY_FILE : data_in
IMAGE_PEAKS_FILE : data_in
IMAGE_INTENSITY_FILE : data_in
SHOCK_EXTENT : data_out
SCREEN_EXTENT : data_out
GRAPH_RESOLUTION : data_out
DISTANCE_FROM_IMAGE_PLANE : data_out
ELECTRODE_SEPARATION : data_out
MAXIMUM_PEAKS : data_out
X_ASPECT_RATIO : data_out
Y_ASPECT_RATIO : data_out
GRAYSCALE_FACTOR : data_out
BRIGHT_FIELD_RESPONSE_CURVE : data_out
DARK_FIELD_RESPONSE_CURVE : data_out
UI_ID : data_out
MACRO_FILE : write_access
IMAGE_FILE : write_access
PARAMETER_FILE : write_access
DENSITY_FILE : write_access
INTENSITY_FILE : write_access
IMAGE_PEAKS_FILE : write_access
IMAGE_INTENSITY_FILE : write_access
HEAP_SIZE_INDICATOR : write_access
CLOCK_DISPLAY : write_access
STATE : write_access

Description: Create accessor.

RT_UI.2 UpdateIndicators

Process Type: Accessor

Interface:
CLOCK_DISPLAY : data_in
HEAP_SIZE_INDICATOR : data_in
UI_ID : data_in
CLOCK_DISPLAY : write_access
HEAP_SIZE_INDICATOR : write_access

Description: Updates the clock and heap indicators on the menu bar and status line.

RT_UI.3 SetAttributes

Process Type: Accessor

Interface:
UI_ID : data_in
SHOCK : data_in
IMAGE : data_in
MENU_BAR : data_in
STATUS_LINE : data_in
CLOCK_DISPLAY : data_in
HEAP_SIZE_INDICATOR : data_in
VACUUM_PARAMETERS : data_in
TAIL_PARAMETERS : data_in
SADDLE_PARAMETERS : data_in
REAR_PARAMETERS : data_in

FRONT_PARAMETERS : data_in
INTERPOLATION_METHOD : data_in
FIRE_RAYS : data_in
FIRE_RAYS_FROM : data_in
FIRE_RAYS_TO : data_in
FIELD : data_in
TOKEN : data_in
WINDOW_SIZE : data_in
CLEAR_PLOTS : data_in
MACRO_DISPLAY : data_in
MACRO_FILE : data_in
IMAGE_FILE : data_in
PARAMETER_FILE : data_in
DENSITY_FILE : data_in
INTENSITY_FILE : data_in
IMAGE_PEAKS_FILE : data_in
IMAGE_INTENSITY_FILE : data_in
ENVIRONMENT_VECTORS : data_in
STATE : data_in
SHOCK : write_access
IMAGE : write_access
MENU_BAR : write_access
STATUS_LINE : write_access
CLOCK_DISPLAY : write_access
HEAP_SIZE_INDICATOR : write_access
VACUUM_PARAMETERS : write_access
TAIL_PARAMETERS : write_access
SADDLE_PARAMETERS : write_access
REAR_PARAMETERS : write_access
FRONT_PARAMETERS : write_access
INTERPOLATION_METHOD : write_access
FIRE_RAYS : write_access
FIRE_RAYS_FROM : write_access
FIRE_RAYS_TO : write_access
FIELD : write_access
TOKEN : write_access
WINDOW_SIZE : write_access
CLEAR_PLOTS : write_access
MACRO_DISPLAY : write_access
MACRO_FILE : write_access
IMAGE_FILE : write_access
PARAMETER_FILE : write_access
DENSITY_FILE : write_access
INTENSITY_FILE : write_access
IMAGE_PEAKS_FILE : write_access
IMAGE_INTENSITY_FILE : write_access
ENVIRONMENT_VECTORS : write_access
STATE : write_access

Description: Write accessor.

RT_UI.4 Hasoperatorpressedkey

Process Type: Test

Interface: UI_ID : data_in
NO : conditional_control_out
YES : conditional_control_out

Description: Checks if the operator has pressed a key.

RT_UI.5 LaunchSystemCalculator

Process Type: Accessor
Interface: UI_ID : data_in
CALCULATOR : data_out
Description: Opens a simple calculator.

RT_UI.6 SwapEnvironmentVectors

Process Type: Accessor
Interface: UI_ID : data_in
ENVIRONMENT_VECTORS : data_out
USER_INTERFACE.ENVIRONMENT_VECTORS : write_access
Description: Swaps the operating system environment vectors.

RT_UI.7 GetEnvironmentVectors

Process Type: Accessor
Interface: UI_ID : data_in
ENVIRONMENT_VECTORS : data_in
USER_INTERFACE.ENVIRONMENT_VECTORS : write_access
Description: Gets the operating system environment vectors.

RT_UI.8 InitialiseMenuSystem

Process Type: Accessor
Interface: UI_ID : data_in
MENUS : data_out
Description: Starts the text-based menu system (after graphics or an operating system shell call).

RT_UI.9 CloseMenuSystem

Process Type: Accessor
Interface: UI_ID : data_in
MENUS : data_out
Description: Closes the text-based menu system (before graphics or an operating system shell call).

RT_UI.10 GetAttributes

Process Type: Accessor
Interface: UI_ID : data_in
SHOCK : data_out
IMAGE : data_out
MENU_BAR : data_out
STATUS_LINE : data_out
CLOCK_DISPLAY : data_out
HEAP_SIZE_INDICATOR : data_out
VACUUM_PARAMETERS : data_out
TAIL_PARAMETERS : data_out
SADDLE_PARAMETERS : data_out
REAR_PARAMETERS : data_out
FRONT_PARAMETERS : data_out
INTERPOLATION_METHOD : data_out
FIRE_RAYS : data_out
FIRE_RAYS_FROM : data_out
FIRE_RAYS_TO : data_out
FIELD : data_out
TOKEN : data_out
WINDOW_SIZE : data_out

CLEAR_PLOTS : data_out
MACRO_DISPLAY : data_out
MACRO_FILE : data_out
IMAGE_FILE : data_out
PARAMETER_FILE : data_out
DENSITY_FILE : data_out
INTENSITY_FILE : data_out
IMAGE_PEAKS_FILE : data_out
IMAGE_INTENSITY_FILE : data_out
ENVIRONMENT_VECTORS : data_out
STATE : data_out
SHOCK : read_access
IMAGE : read_access
MENU_BAR : read_access
STATUS_LINE : read_access
CLOCK_DISPLAY : read_access
HEAP_SIZE_INDICATOR : read_access
VACUUM_PARAMETERS : read_access
TAIL_PARAMETERS : read_access
SADDLE_PARAMETERS : read_access
REAR_PARAMETERS : read_access
FRONT_PARAMETERS : read_access
INTERPOLATION_METHOD : read_access
FIRE_RAYS : read_access
FIRE_RAYS_FROM : read_access
FIRE_RAYS_TO : read_access
FIELD : read_access
TOKEN : read_access
WINDOW_SIZE : read_access
CLEAR_PLOTS : read_access
MACRO_DISPLAY : read_access
MACRO_FILE : read_access
IMAGE_FILE : read_access
PARAMETER_FILE : read_access
DENSITY_FILE : read_access
INTENSITY_FILE : read_access
IMAGE_PEAKS_FILE : read_access
IMAGE_INTENSITY_FILE : read_access
ENVIRONMENT_VECTORS : read_access
STATE : read_access

Description: Read accessor.

RT_UI.11 IsFireRaysSelected

Process Type: Test

Interface: FIRE_RAYS : data_in
YES : conditional_control_out
NO : conditional_control_out

Description: Determines if the fire rays option has been selected.

RT_UI.12 OpenMacroFile

Process Type: Accessor

Interface: MACRO_FILE : data_in
OPEN : data_out

Description: Opens a macro file of raytracing commands.

RT_UI.14 EndOfMacroFile

Process Type: Accessor

Interface: UI_ID : data_in
TOKEN : data_in
YES : conditional_control_out
NO : conditional_control_out

Description: Determines if the end of the macro file has been reached.

RT_UI.15 NextToken

Process Type: Accessor

Interface: TOKEN : data_in
REAR_PARAMETERS : data_out
VACUUM_PARAMETERS : data_out
TAIL_PARAMETERS : data_out
SADDLE_PARAMETERS : data_out
FRONT_PARAMETERS : data_out
FIRE_RAYS_FROM : data_out
FIRE_RAYS_TO : data_out
INTENSITY_FILE : data_out
DENSITY_FILE : data_out
PARAMETER_FILE : data_out
INTERPOLATION_METHOD : data_out

Description: Reads the next token and converts it to data once the data type is known. This can be determined from the preceding token type which comes into the process as a conditional control. Conditional and unconditional controls into a process are not shown on Process Specifications to enhance reuse.

RT_UI.16 ExamineTokenType

Process Type: Test

Interface: TOKEN : data_in
END_KEYWORD : conditional_control_out
VACUUM_KEYWORD : conditional_control_out
TAIL_KEYWORD : conditional_control_out
SADDLE_KEYWORD : conditional_control_out
REAR_KEYWORD : conditional_control_out
FRONT_KEYWORD : conditional_control_out
FIRE_RAYS_KEYWORD : conditional_control_out
INTERPOLATE_KEYWORD : conditional_control_out
DISPLAY_KEYWORD : conditional_control_out
PARAMETER_FILE_KEYWORD : conditional_control_out
DENSITY_FILE_KEYWORD : conditional_control_out
INTENSITY_FILE_KEYWORD : conditional_control_out
NEW_KEYWORD : conditional_control_out

Description: Determines the type of token read.

RT_UI.17 ClearFiles

Process Type: Accessor

Interface: UI_ID : data_in
CLEAR : data_out

Description: Clears the Intensity, Density and Parameter files. This is done whenever a NEW keyword in the parameter file indicates that a new set of density parameters must be raytraced.

RT_UI.18 CloseMacroFile

Process Type: Accessor

Interface: UI_ID : data_in
CLOSE : data_out

Description: Closes the macro file when the last command has been parsed and actioned.

RT_UI.19 deleteUSER_INTERFACE

Process Type: Accessor
Interface: UI_ID : data_in
Description: Delete accessor.

RT_UI.24 DisplaySystemCredits

Process Type: Accessor
Interface: UI_ID : data_in
CREDITS : data_out
Description: Displays the system credits.

RT_UI.25 CheckParameters

Process Type: Accessor
Interface: UI_ID : data_in
OK : conditional_control_out
VACUUM_PARAMETERS : read_access
TAIL_PARAMETERS : read_access
SADDLE_PARAMETERS : read_access
REAR_PARAMETERS : read_access
FRONT_PARAMETERS : read_access
FIRE_RAYS_FROM : read_access
FIRE_RAYS_TO : read_access
INTERPOLATION_METHOD : read_access
Description: Checks the density parameters and raytrace limits before attempting to raytrace. These must be within bounds and have monotonically increasing position values.

RT_UI.26 DisplayGraphics

Process Type: Test
Interface: MACRO_DISPLAY : data_in
TRUE : conditional_control_out
Description: Determines if the density profile and intensity pattern graphs and parameter data should be displayed during batch processing of a macro file. This is determined from a macro keyword in the macro file itself.

RT_UI.32 generateRT_UI2:RUN_SHOCK_MACRO_FILE

Process Type: Event Generator
Interface: UI_ID : data_in
MACRO_FILE : data_in
RT_UI2:RUN_SHOCK_MACRO_FILE : generated_event
Description: Event generator.

RT_UI.33 generateRT_UI3:PARSE_MACRO_FILE

Process Type: Event Generator
Interface: UI_ID : data_in
RT_UI3:PARSE_MACRO_FILE : generated_event
Description: Event generator.

RT_UI.34 generateRT_UI4:AUTO_SHOCK

Process Type: Event Generator
Interface: UI_ID : data_in
RT_UI4:AUTO_SHOCK : generated_event
Description: Event generator.

RT_UI.35 generateRT_UI5:DISPLAY_SHOCK

Process Type: Event Generator
Interface: UI_ID : data_in
RT_UI5:DISPLAY_SHOCK : generated_event
Description: Event generator.

RT_UI.36 generateRT_UI6:WAIT_FOR_SHOCK
Process Type: Event Generator
Interface: UI_ID : data_in
RT_UI6:WAIT_FOR_SHOCK : generated_event
Description: Event generator.

RT_UI.37 generateRT_UI7:LOAD_SHOCK_PARAMETERS
Process Type: Event Generator
Interface: UI_ID : data_in
PARAMETER_FILE : data_in
RT_UI7:LOAD_SHOCK_PARAMETERS(UI_ID, PARAMETER_FILE) : generated_event
Description: Event generator.

RT_UI.39 generateRT_UI9:SAVE_SHOCK_PARAMETERS
Process Type: Event Generator
Interface: UI_ID : data_in
PARAMETER_FILE : data_in
RT_UI9:SAVE_SHOCK_PARAMETERS(UI_ID, PARAMETER_FILE) : generated_event
Description: Event generator.

RT_UI.40 generateRT_UI10:SAVE_RAW_SHOCK_DENSITIES
Process Type: Event Generator
Interface: UI_ID : data_in
DENSITY_FILE : data_in
RT_UI10:SAVE_RAW_SHOCK_DENSITIES(UI_ID, DENSITY_FILE) : generated_event
Description: Event generator.

RT_UI.41 generateRT_UI11:SAVE_RAW_SHOCK_INTENSITIES
Process Type: Event Generator
Interface: UI_ID : data_in
INTENSITY_FILE : data_in
RT_UI11:SAVE_RAW_SHOCK_INTENSITIES(UI_ID, INTENSITY_FILE) : generated_event
Description: Event generator.

RT_UI.42 generateRT_UI12:SET_DENSITY
Process Type: Event Generator
Interface: USER_INTERFACE.INTERPOLATION_METHOD : data_in
USER_INTERFACE.VACUUM_PARAMETERS : data_in
USER_INTERFACE.TAIL_PARAMETERS : data_in
USER_INTERFACE.SADDLE_PARAMETERS : data_in
USER_INTERFACE.REAR_PARAMETERS : data_in
USER_INTERFACE.FRONT_PARAMETERS : data_in
USER_INTERFACE.FIRE_RAYS_FROM : data_in
USER_INTERFACE.FIRE_RAYS_TO : data_in
USER_INTERFACE.FIRE_RAYS : data_in
USER_INTERFACE.UI_ID : data_in
RT_UI12:SET_DENSITY : generated_event
Description: Event generator.

RT_UI.43 generateRT_UI13:LOAD_IMAGE_FILE

Process Type: Event Generator
Interface: UI_ID : data_in
IMAGE_FILE : data_in
RT_UI13:LOAD_IMAGE_FILE(UI_ID, IMAGE_FILE) : generated_event
Description: Event generator.

RT_UI.44 generateRT_UI14:EXTRACT_INTENSITY
Process Type: Event Generator
Interface: USER_INTERFACE.FIELD : data_in
USER_INTERFACE.WINDOW_SIZE : data_in
USER_INTERFACE.CLEAR_PLOTS : data_in
USER_INTERFACE.UI_ID : data_in
RT_UI14:EXTRACT_INTENSITY : generated_event
Description: Event generator.

RT_UI.45 generateRT_UI15:WAIT_FOR_IMAGE
Process Type: Event Generator
Interface: UI_ID : data_in
RT_UI15:WAIT_FOR_IMAGE : generated_event
Description: Event generator.

RT_UI.46 generateRT_UI16:DISPLAY_IMAGE
Process Type: Event Generator
Interface: UI_ID : data_in
RT_UI16:DISPLAY_IMAGE : generated_event
Description: Event generator.

RT_UI.47 generateRT_UI17:DETERMINE_SCALE
Process Type: Event Generator
Interface: UI_ID : data_in
RT_UI17:DETERMINE_SCALE : generated_event
Description: Event generator.

RT_UI.48 generateRT_UI18:SAVE_RAW_IMAGE_INTENSITIES
Process Type: Event Generator
Interface: UI_ID : data_in
IMAGE_INTENSITY_FILE : data_in
RT_UI18:SAVE_RAW_IMAGE_INTENSITIES(UI_ID, IMAGE_INTENSITY_FILE) : generated_event
Description: Event generator.

RT_UI.49 generateRT_UI19:SAVE_RAW_INTENSITY_PEAKS
Process Type: Event Generator
Interface: UI_ID : data_in
IMAGE_PEAKS_FILE : data_in
RT_UI19:SAVE_RAW_INTENSITY_PEAKS(UI_ID, IMAGE_PEAKS_FILE) : generated_event
Description: Event generator.

RT_UI.50 generateRT_UI20:DISPLAY_ABOUT
Process Type: Event Generator
Interface: UI_ID : data_in
RT_UI20:DISPLAY_ABOUT : generated_event
Description: Event generator.

RT_UI.51 generateRT_UI21:DISPLAY_CALCULATOR
Process Type: Event Generator

Interface: UI_ID : data_in
RT_UI21:DISPLAY_CALCULATOR : generated_event
Description: Event generator.

RT_UI.52 generateRT_UI22:GO_TO_DOS_SHELL
Process Type: Event Generator
Interface: UI_ID : data_in
RT_UI22:GO_TO_DOS_SHELL : generated_event
Description: Event generator.

RT_UI.53 generateRT_UI23:DONE
Process Type: Event Generator
Interface: UI_ID : data_in
RT_UI23:DONE : generated_event
Description: Event generator.

RT_UI.54 generateRT_UI24:DELETE
Process Type: Event Generator
Interface: UI_ID : data_in
RT_UI24:DELETE : generated_event
Description: Event generator.

D.16 Structure Graphs

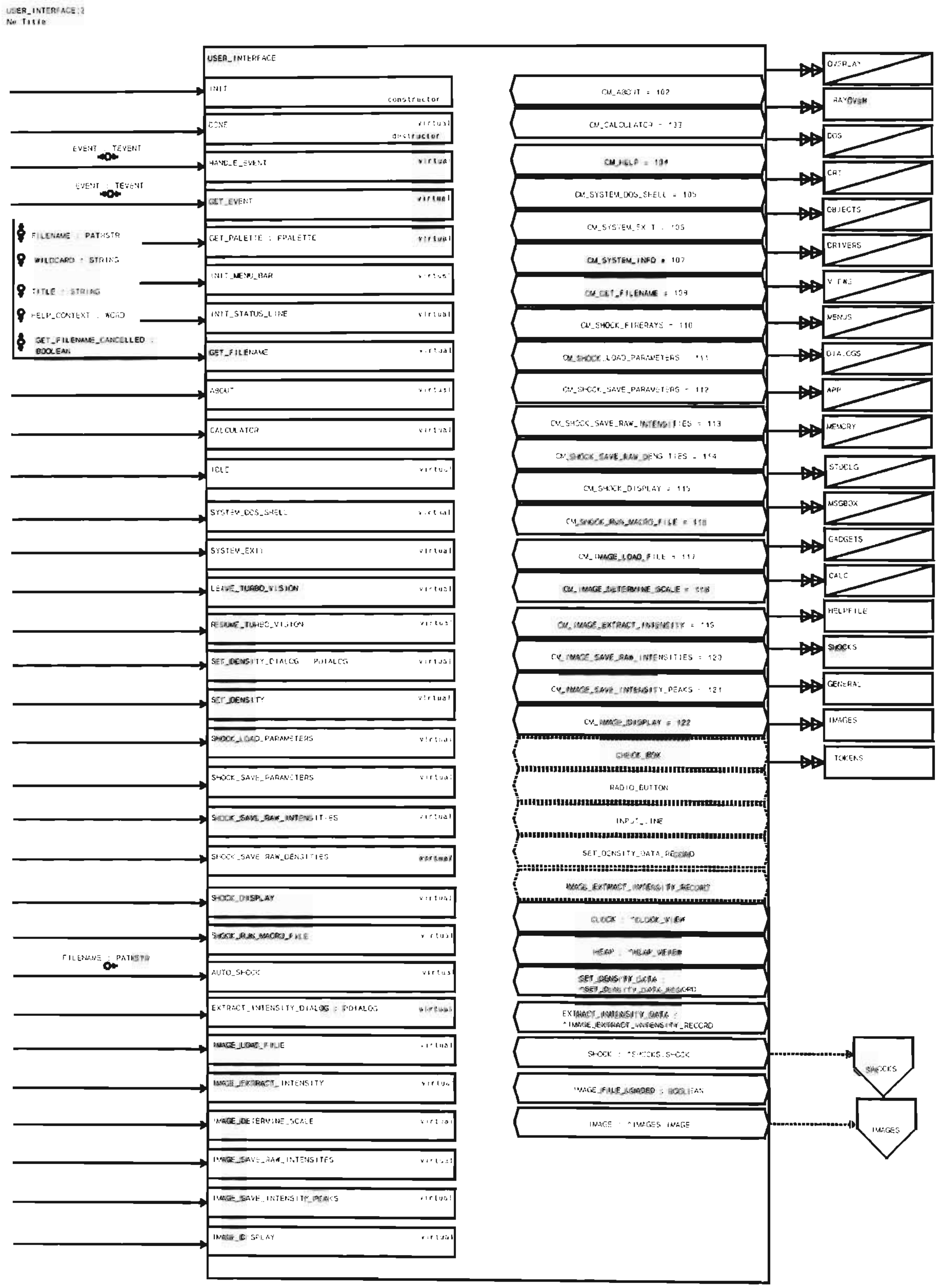


Figure D.96 Structure Graph for the User Interface main program / object

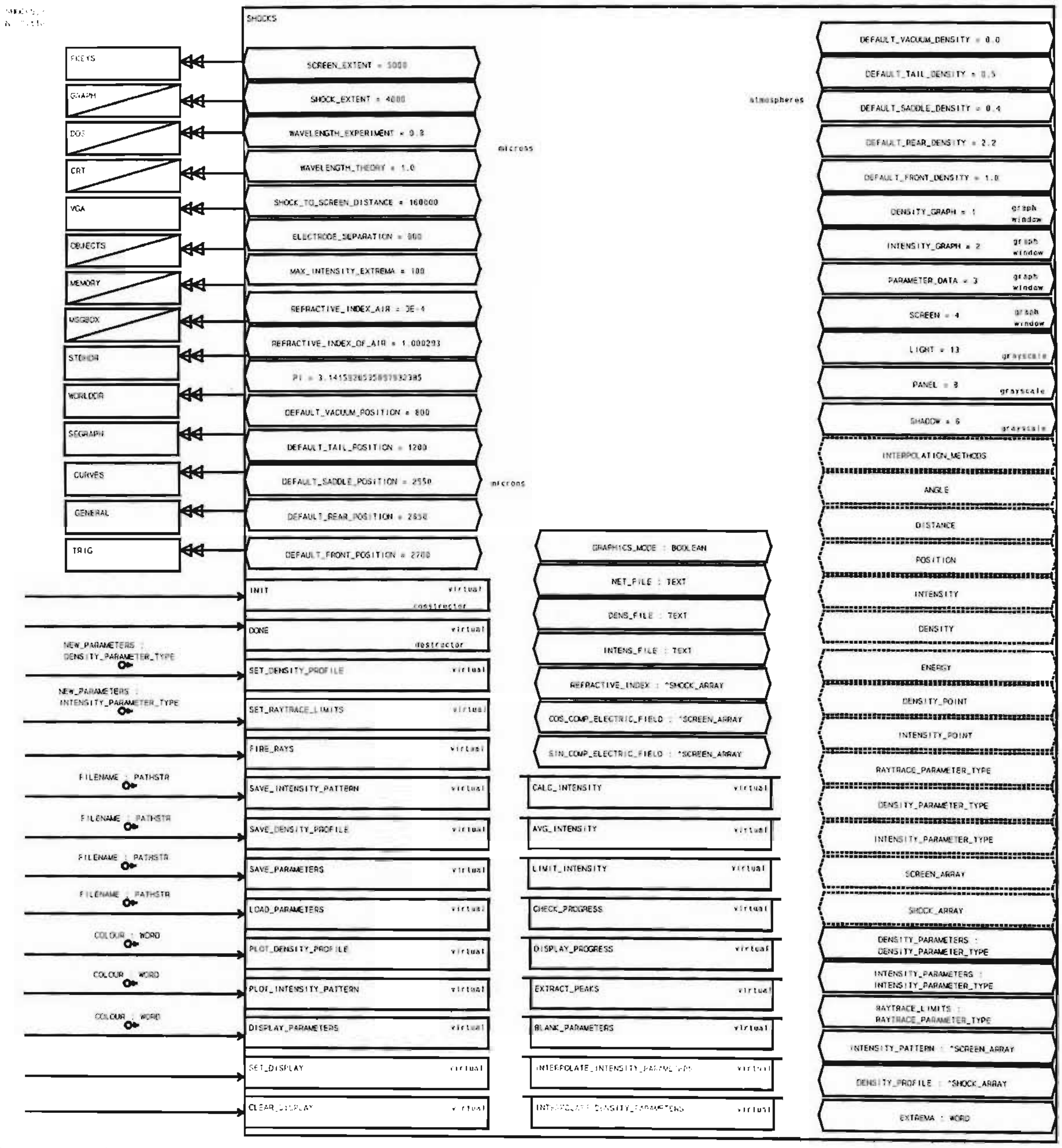


Figure D.97 Structure Graph for the Shocks unit / object

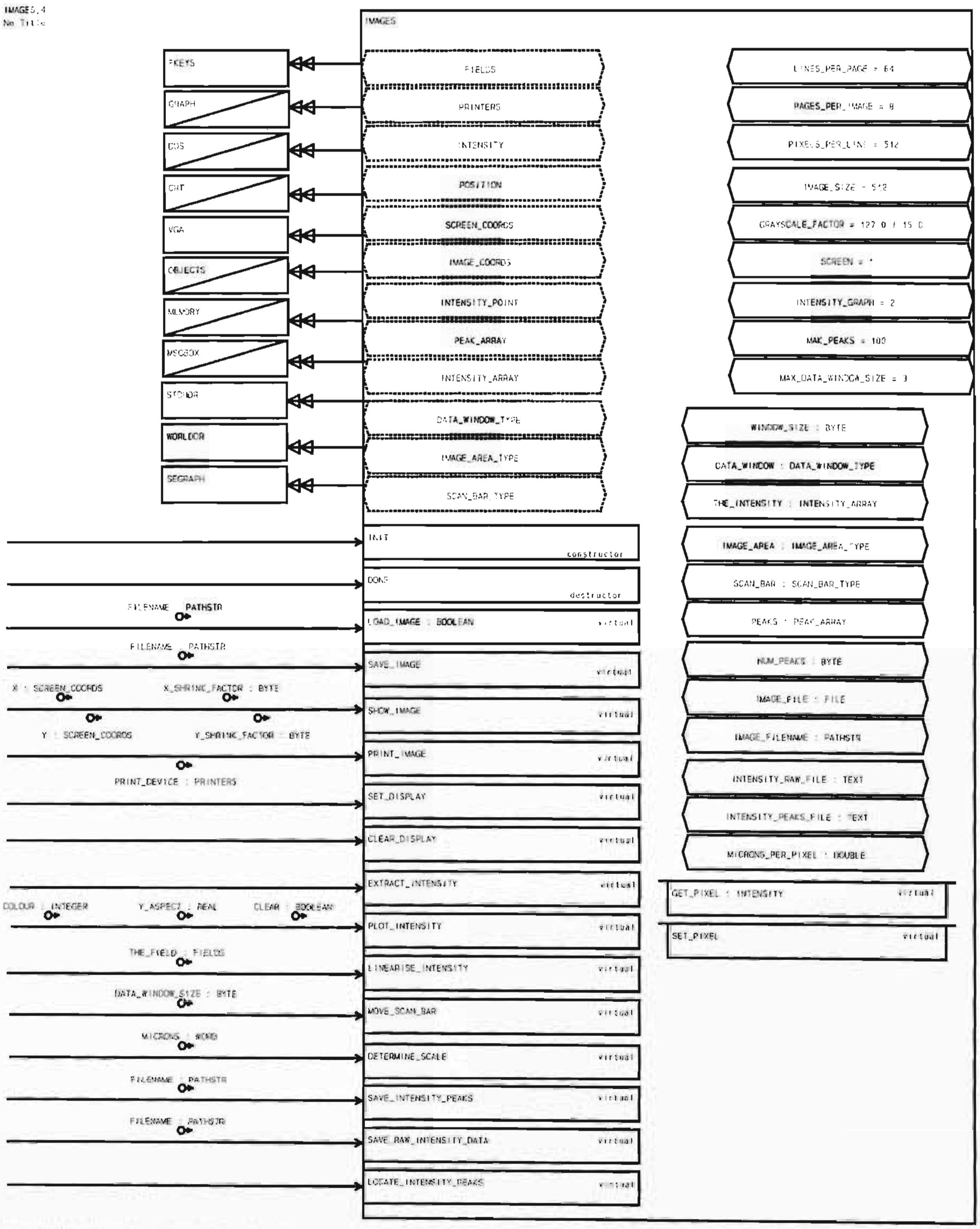


Figure D.98 Structure Graph for the Images unit / object

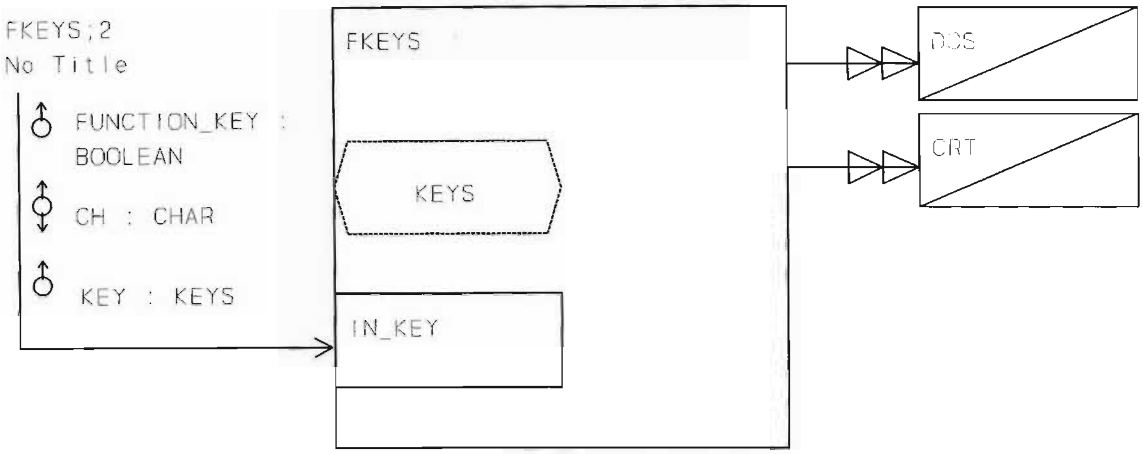


Figure D.99 Structure Graph for the Fkeys unit

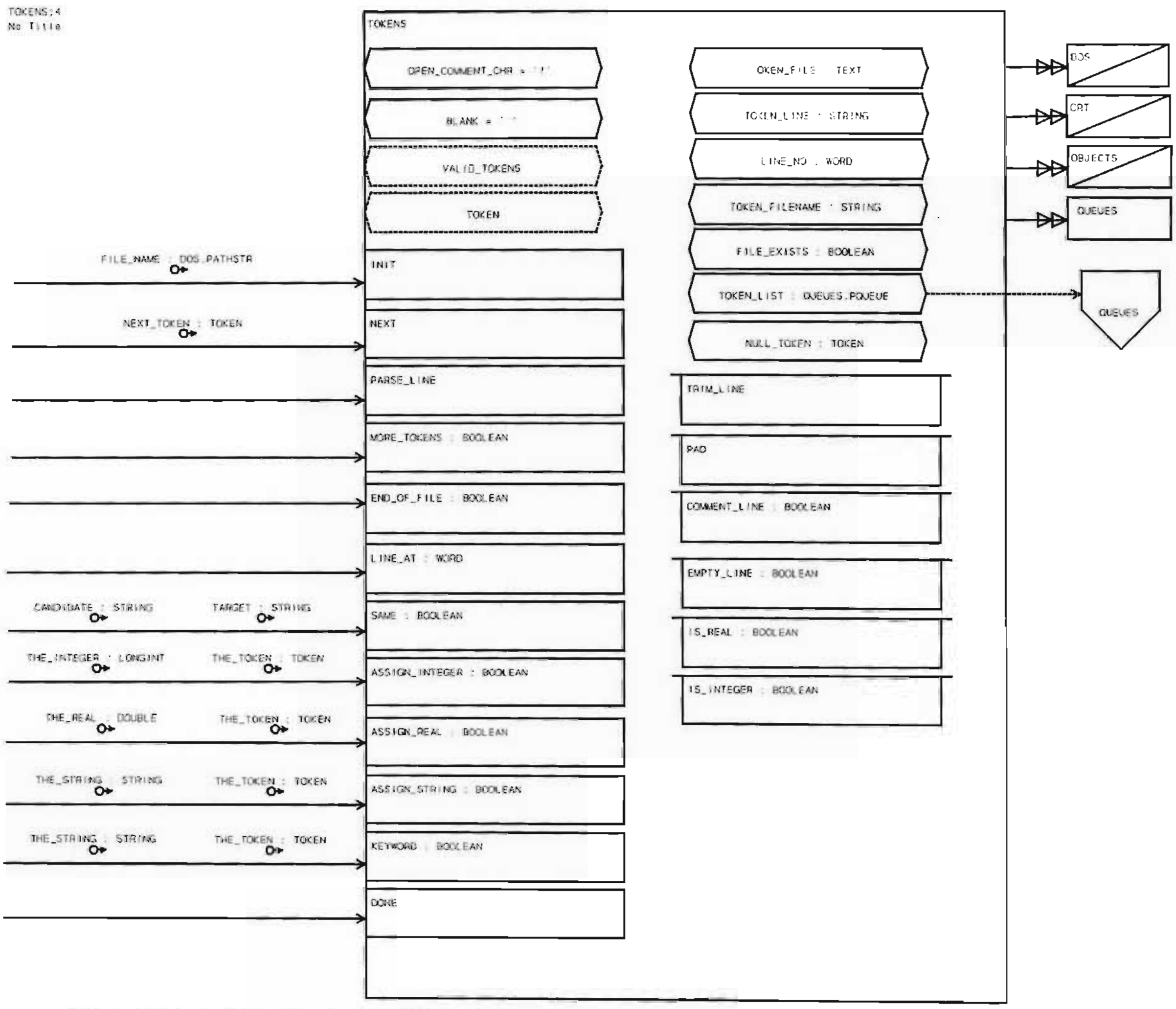


Figure D.100 Structure Graph for the Tokens unit / object

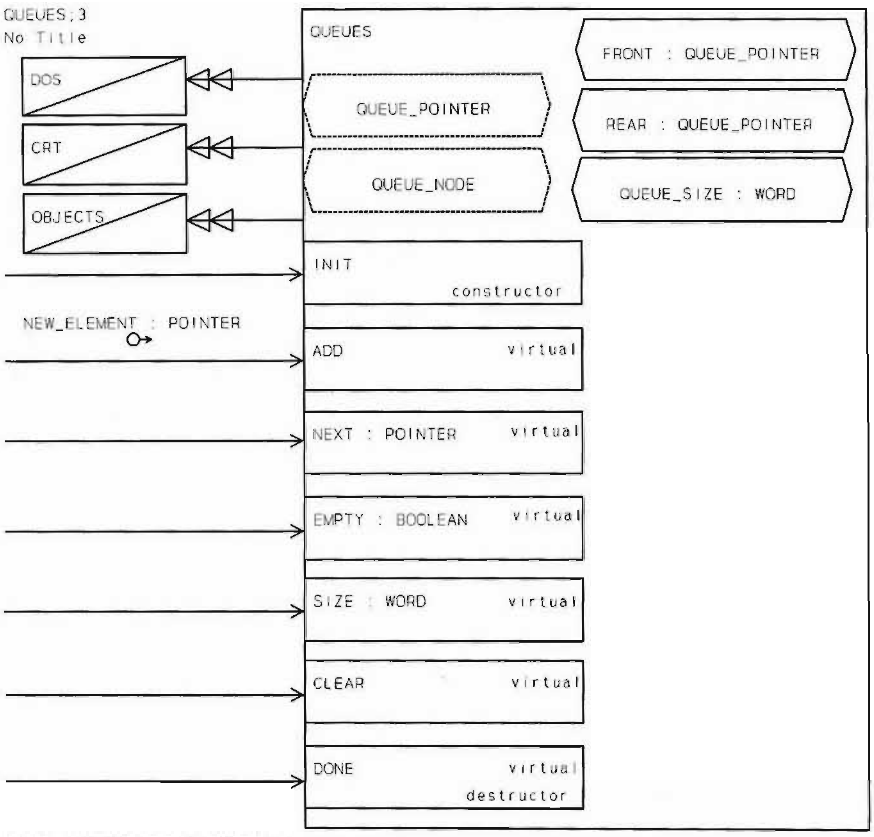


Figure D.101 Structure Graph for the Queues unit / object

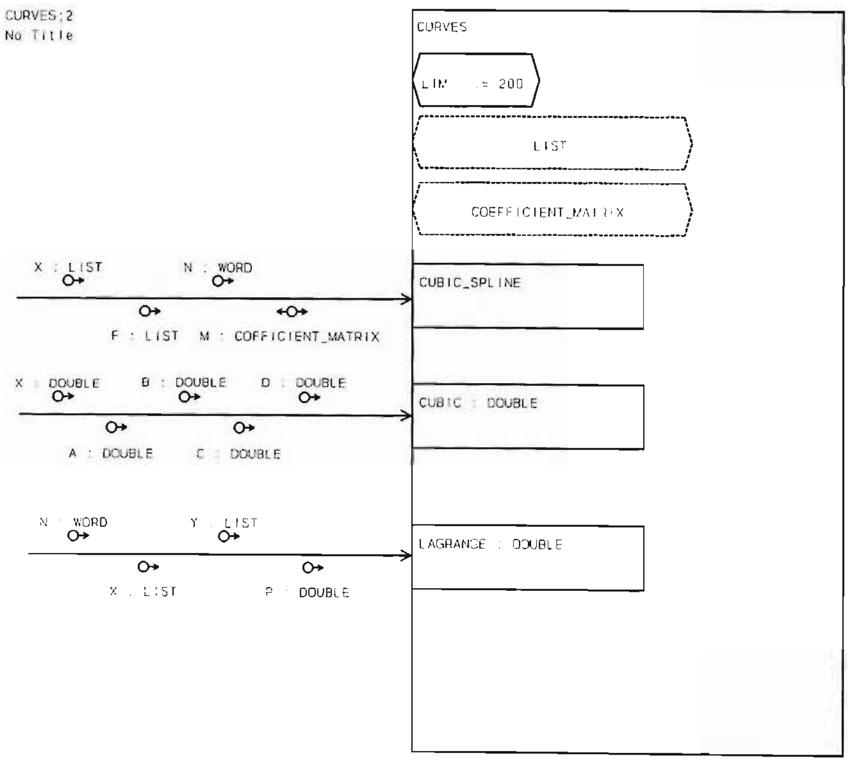


Figure D.102 Structure Graph for the Curves unit

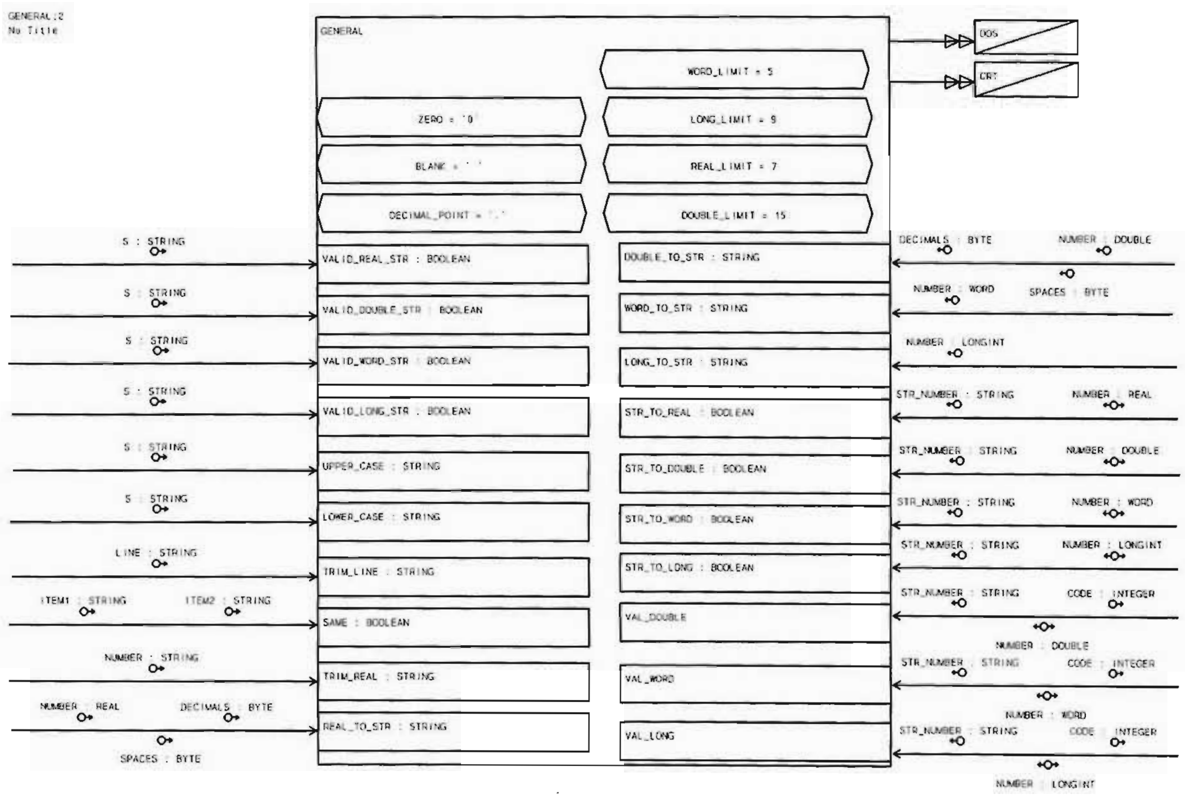


Figure D.103 Structure Graph for the General unit

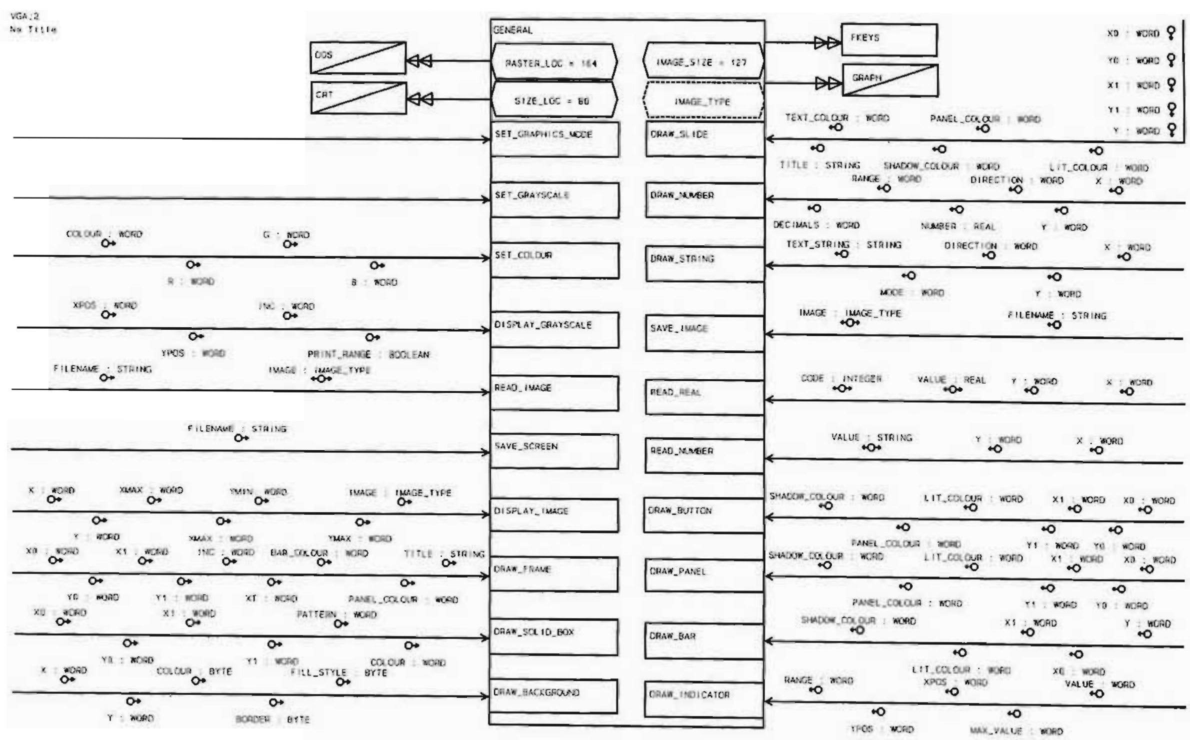


Figure D.104 Structure Graph for the Vga unit

D.17 Software Metrics

In this section we present some measurements of the ray-tracer code's size and complexity. The 'Lines of Code' metric most often used to indicate code size is quite primitive and sensitive to different programming languages and formatting styles [Mil88]. The Halstead Software Science metrics attempt to provide more sophisticated measurements of code size. It is also convenient to have some indication of the code's complexity. The McCabe Cyclomatic Complexity metric is commonly used for this purpose. DeMarco [DeM82] combines the previous sets of metrics to yield a single Implementation Weight metric. Note that these metrics can only be evaluated for existing code so their usefulness in estimating the time and cost of new projects is limited. Function point techniques [Mil88], statistical models (such as Cocomo [Boe81]) and analysis metrics (such as DeMarco's 'Bang' [DeM82]) may be used for such estimation. Other metrics attempt to measure quality, defect rates, reliability and maintainability [Mil88, IEE92].

D.17.1 Halstead Software Science Metrics

The Halstead metrics consider a computer program to be a sequence of operator and operand tokens [Mil88, IEE92, DeM82]. The vocabulary, n, of a program is then defined to be

$$n = n_1 + n_2 \tag{D.1}$$

where

- n_1 is the number of unique operators in the program, and
- n_2 is the number of unique operands in the program, and
- n the total number of unique tokens in the program.

The program length is a count of the total number of operators and operands in the program. Specifically

$$N = N_1 + N_2 \tag{D.2}$$

where

- N_1 is the total number of operators in the program, and
- N_2 the total number of operands.

Another measure of program size is the program volume, which is defined by Halstead as

$$V = N \cdot \log_2 n \qquad \textbf{(D.3).}$$

The effort, E, required to develop the software is given by

$$E = \frac{n_1 n_2 (n_1 \log_2 n_1 + n_2 \log_2 n_2) \log_2 n}{2 n_2} \qquad \textbf{(D.4).}$$

in units of *elementary mental discriminations*.

By using the Stroud number, S, the corresponding coding time T (in seconds) may be derived. The Stroud number is usually taken as 18 [Mil88].

$$T = \frac{E}{S} \approx \frac{N^2 \log_2 n}{4S} \qquad \textbf{(D.5).}$$

D.17.2 McCabe Cyclomatic Complexity

The cyclomatic complexity of a structured program may be computed by counting all its decision points, adding 1 for each 'if-then-else' type construct and n-1 for each n-way 'case' type construct [Mil88, IEE92, DeM82].

D.17.3 DeMarco Implementation Weight

DeMarco's Implementation Weight is a composite implementation predictor utilising both module size and complexity.

$$I = \frac{V \times C}{A} \qquad \textbf{(D.6),}$$

where

- V is the Halstead volume of the module,
- C is its McCabe Cyclomatic complexity and
- A is an adaption factor used to discount the weight of adapted or reused code. Adapted code is typically discounted by 21-22% [DeM82].

D.17.4 Metrics Listing

The results of applying the metrics reviewed above to the ray-tracer code are presented in table (D.1).

Table D.1 Software Metrics

Metric/File	raytrace	shocks	images	tokens	queues	curves
Source Size (bytes)	56835	43796	26083	11847	4003	5422
Lines of Code	1372	892	633	250	112	71
Lines of Comments	119	158	69	81	20	43
Code/Comment Ratio	6.89	5.65	9.17	3.09	5.60	1.65
Number of Procedures[1]	43	27	25	19	8	4
Average Procedure Size (LOC)	32	33	25	13	14	18
Halstead Length	4064	3206	2199	785	266	390
Halstead Vocabulary	798	537	431	176	71	83
Halstead Volume	39178	29075	19245	5856	1636	2486
McCabe Cyclomatic Complexity	129	50	43	32	12	2
Average Complexity/Procedure	3.00	1.85	1.72	1.68	1.5	0.5
DeMarco Information Weight	5053956	1453725	827520	187381	19630	4973
Software Science Stroud No.	18	18	18	18	18	18
Software Science Coding Time (hrs)	614.27	359.62	163.27	17.73	1.68	3.74
Coding Rate (lines/hr)	2.23	2.48	3.88	14.10	66.72	18.98

[1] The main **begin** .. **end.** code block is counted as a procedure.

Table D.1 Software Metrics

Metric/File	general	vga	fkeys	ray	rayover	metrics
Source Size (bytes)	14559	17081	2785	3378	1608	15122
Lines of Code	246	415	57	63	20	288
Lines of Comments	88	48	12	13	12	93
Code/Comment Ratio	2.80	8.65	4.75	4.85	1.67	3.10
Number of Procedures[1]	22	21	2	1	1	16
Average Procedure Size (LOC)	11	20	28	63	20	18
Halstead Length	866	1477	216	200	50	1020
Halstead Vocabulary	108	296	92	102	43	183
Halstead Volume	5850	12125	1409	1334	271	7666
McCabe Cyclomatic Complexity	32	32	30	10	5	86
Average Complexity/Procedure	1.45	1.52	15.00	10.00	5.00	5.38
DeMarco Information Weight	187191	388012	42273	13345	1357	659277
Software Science Stroud No.	18	18	18	18	18	18
Software Science Coding Time (hrs)	19.54	69.09	1.17	1.03	0.05	30.17
Coding Rate (lines/hr)	12.59	6.01	48.54	61.18	382.14	9.55

[1] The main **begin** .. **end.** code block is counted as a procedure.

The Coding Time depends on the selection of the Stroud Number. This is usually set to 18, but may vary depending on the software development processes and project history of a particular organisation. The Coding Time metric is therefore somewhat subjective and may appear to be overestimated in table (D.1). Remember though that we can interpret the metric to include the time needed to test, debug, integrate and formally validate the code modules. This time can be substantial in large projects, rivalling or exceeding the actual coding time. McCabe's Cyclomatic Complexity should actually be calculated individually for each procedure. Procedures with a complexity above some threshold (usually 10 [DeM82]) are considered overly complex and are targets for redesign. We now turn our attention from code metrics to analysis metrics.

D.17.5 DeMarco's Bang Metric

As mentioned in section (D.17), for early, pre-coding estimates, DeMarco's Bang metric may be calculated once the system software analysis is complete. Bang is a measure of the functionality delivered by software, from the perspective of the user. It is an implementation-independent measure of the system size based on the system specification and is therefore available very early in the project when initial forecasts for cost and duration are often requested. Calculation of Bang before and after the addition of new requirements gives an indication of how much the system requirements have grown.

Bang operates on the system's functional primitives. In our methodology, functional primitives correspond to the bubbles in the data flow diagram. To get an idea of the complexity of each functional primitive, each bubble is assigned to one of the weighted classes listed in table (D.2) due to DeMarco [DeM82]. The size of the functional primitive is measured by its token count. A token is a data item passed or received by a functional primitive without further subdivision. The size of the functional primitive is proportional to the log of the token count and DeMarco [DeM82] provides the transformation table (D.3). Bang for a functional primitive is calculated by multiplying the weighting factor from table (D.2) by the corrected size factor obtained by accessing table (D.3) with the token count. The bang of a data flow diagram is the sum of the bangs of all its component functional primitives.

Table D.2 Complexity Weighting Factors for Functional Primitives

Class	Function	Weight
Separation	Divide incoming data items	0.6
Amalgamation	Combine incoming data items	0.6
Data Direction	Steer data according to control variable	0.3
Simple Update	Update one or more items of stored data	0.5
Storage Management	Analyse stored data and act on its state	1.0
Edit	Evaluate net input data at the mmi boundary	0.8
Verification	Check for and report on internal consistency	1.0
Text Manipulation	Handle text strings	1.0
Synchronisation	Decide when to act or prompt others to act	1.5
Output Generation	Format non-tabular net output data flows	1.0
Display	Construct 2d output (graphs etc.)	1.8
Tabular Analysis	Perform formatting & simple tabular reporting	1.0
Arithmetic	Perform simple arithmetic calculation	0.7
Initiation	Establish starting values for stored data	1.0
Computation	Perform complex mathematics	2.0
Device Management	Control devices next to the computer boundary	2.5

Table D.3 Size Correction of Functional Primitives

Token Count	Corrected Size
2	1.0
3	2.4
4	4.0
5	5.8
6	7.8
7	9.8
8	12.0
9	14.3
10	16.6
11	19.0
12	21.5
13	24.1
14	26.7
15	29.3
16	32.0
17	34.7
18	37.6
19	40.4
20	43.2

The CASE tool can compute Bang for the analysis model's data flow diagrams. The results for some diagrams are presented in table (D.4).

Table D.4 DeMarco's Bang for Selected Data Flow Diagrams from the Software Analysis

Title	Class	Weight	Tokens In	Tokens Out	Data Weight	Bang
RT_CS.1 Snells Law	Computation	2.000	4	1	5.805	11.610
RT_CS.2 Calculate Op	Computation	2.000	5	1	7.755	15.510
RT_CS.10 Get Attribu	Simple Update	0.500	3	2	5.805	2.902
RT_CS.4 Calculate Em	Computation	2.000	5	1	7.755	15.510
RT_CS.3 Set Attribut	Simple Update	0.500	6	0	7.755	3.877
Total Bang for DFD COMPUTATIONAL_SUBSHELL.1						49.409
RT_DP.1 create DENSI	Initiation	1.000	1	2	2.377	2.377
Total Bang for DFD DENSITY_PARAMETERS.2						2.377
RT_DB.10 Get Attribu	Simple Update	0.500	2	1	2.377	1.189
RT_UI.10 Get Attribu	Simple Update	0.500	2	1	2.377	1.189
RT_I.10 Get Attribut	Simple Update	0.500	3	2	5.805	2.902
RT_EI.2 Get Data For	Output Generation	1.000	2	0	1.000	1.000
RT_EI.3 Set Attribut	Simple Update	0.500	1	0	0.500	0.250
RT_EI.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
Total Bang for DFD EXTRACT_INTENSITY.1						6.830
RT_EI.5 Get Data Fro		0.000	0	0	0.000	0.000
RT_EI.3 Set Attribut		0.000	0	0	0.000	0.000
RT_UI.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_UI.35 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_S.32 generate RT_	Data Direction	0.300	0	2	1.000	0.300
RT_UI.10 Get Attribu	Simple Update	0.500	3	0	2.377	1.189

Total Bang for DFD USER_INTERFACE.30						3.789
RT_I.9 Close Graphic	Device Management	2.500	1	1	1.000	2.500
RT_I.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD IMAGE.10						3.000
RT_S.40 generate RT_	Data Direction	0.300	0	2	1.000	0.300
Total Bang for DFD USER_INTERFACE.33						3.789
RT_I.10 Get Attribut	Simple Update	0.500	5	4	14.265	7.132
RT_NB.10 Get Attribu	Simple Update	0.500	3	2	5.805	2.902
RT_I.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
RT_I.7 Locate Intens	Computation	2.000	5	2	9.826	19.651
Total Bang for DFD IMAGE.11						30.186
RT_UI.10 Get Attribu	Simple Update	0.500	3	0	2.377	1.189
Total Bang for DFD USER_INTERFACE.34						3.789
RT_I.10 Get Attribut	Simple Update	0.500	3	3	7.755	3.877
RT_I.3 Set Attribute	Simple Update	0.500	2	2	4.000	2.000
RT_I.8 Write Intensi	Output Generation	1.000	4	1	5.805	5.805
Total Bang for DFD IMAGE.12						11.682
RT_I.34 generate RT_	Data Direction	0.300	5	0	5.805	1.741
RT_UI.10 Get Attribu	Simple Update	0.500	2	1	2.377	1.189
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_I.37 generate RT_	Data Direction	0.300	0	1	0.500	0.150
RT_I.38 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_I.36 generate RT_	Data Direction	0.300	0	2	1.000	0.300
RT_UI.45 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_I.10 Get Attribut	Simple Update	0.500	6	5	19.027	9.513
Total Bang for DFD USER_INTERFACE.35						16.094
RT_I.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
RT_I.11 delete IMAGE	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD IMAGE.13						1.000
RT_I.34 generate RT_	Data Direction	0.300	5	0	5.805	1.741
RT_UI.10 Get Attribu	Simple Update	0.500	3	3	7.755	3.877
RT_I.35 generate RT_	Data Direction	0.300	2	2	4.000	1.200
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_S.10 Get Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_I.10 Get Attribut	Simple Update	0.500	5	4	14.265	7.132
Total Bang for DFD USER_INTERFACE.36						16.240
RT_I.10 Get Attribut	Simple Update	0.500	3	2	5.805	2.902
RT_EB.10 Get Attribu	Simple Update	0.500	3	2	5.805	2.902
RT_I.3 Set Attribute	Simple Update	0.500	3	0	2.377	1.189
RT_I.13 Scale Image	Computation	2.000	4	1	5.805	11.610
RT_EB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
Total Bang for DFD IMAGE.2						18.903
RT_I.10 Get Attribut	Simple Update	0.500	4	3	9.826	4.913
RT_I.3 Set Attribute	Simple Update	0.500	5	0	5.805	2.902
RT_I.15 Plot Pixel	Output Generation	1.000	4	1	5.805	5.805
RT_I.16 Get Pixel Po	Computation	2.000	4	2	7.755	15.510
RT_I.14 Map Pixel Va	Arithmetic	0.700	4	1	5.805	4.063
Total Bang for DFD IMAGE.3						33.193
RT_I.10 Get Attribut	Simple Update	0.500	3	2	5.805	2.902
RT_NB.10 Get Attribu	Simple Update	0.500	4	3	9.826	4.913
RT_I.3 Set Attribute	Simple Update	0.500	3	3	7.755	3.877
RT_I.21 Position and	Computation	2.000	6	1	9.826	19.651
RT_I.22 Average and	Computation	2.000	1	1	1.000	2.000
Total Bang for DFD IMAGE.4						33.344
RT_UI.10 Get Attribu	Simple Update	0.500	2	2	4.000	2.000
RT_I.38 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_UI.45 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_I.10 Get Attribut	Simple Update	0.500	5	4	14.265	7.132
Total Bang for DFD USER_INTERFACE.39						12.273
RT_I.10 Get Attribut	Simple Update	0.500	4	3	9.826	4.913
RT_I.23 Plot Intensi	Device Management	2.500	3	1	4.000	10.000
RT_I.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD IMAGE.5						15.413
RT_I.4 Initialise Gr	Device Management	2.500	2	1	2.377	5.944
RT_I.5 Setup Graph W	Initiation	1.000	1	2	2.377	2.377
RT_I.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD IMAGE.6						8.821
RT_I.10 Get Attribut	Simple Update	0.500	3	2	5.805	2.902
RT_NB.10 Get Attribu	Simple Update	0.500	3	2	5.805	2.902
RT_I.3 Set Attribute	Simple Update	0.500	2	2	4.000	2.000
RT_I.6 Write Raw In	Output Generation	1.000	4	1	5.805	5.805
Total Bang for DFD IMAGE.7						13.610
RT_I.2 Load Image Da	Initiation	1.000	3	2	5.805	5.805
RT_I.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD IMAGE.8						6.305
RT_UI.6 Swap Environ	Storage Management	1.000	1	1	1.000	1.000
RT_UI.7 Get Environm	Storage Management	1.000	2	1	2.377	2.377
RT_UI.6 Swap Environ	Storage Management	1.000	1	1	1.000	1.000
RT_UI.8 Initialise M	Initiation	1.000	1	1	1.000	1.000
RT_UI.9 Close Menu S	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD USER_INTERFACE.42						6.677
RT_I.1 create IMAGE		0.000	0	0	0.000	0.000
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_S.38 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_UI.10 Get Attribu	Simple Update	0.500	2	1	2.377	1.189
Total Bang for DFD USER_INTERFACE.44						2.289
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_UI.53 generate RT	Data Direction	0.300	2	1	2.377	0.713
RT_UI.45 generate RT	Data Direction	0.300	2	1	2.377	0.713

RT_UI.4 Has operator	Data Direction	0.300	1	2	2.377	0.713
Total Bang for DFD USER_INTERFACE.45						2.640
RT_UI.47 generate RT	Data Direction	0.300	2	1	2.377	0.713
RT_UI.46 generate RT	Data Direction	0.300	2	1	2.377	0.713
Total Bang for DFD IMAGES.1						15.936
RT_DP.10 Get Attribu	Simple Update	0.500	2	1	2.377	1.189
RT_S.10 Get Attribut	Simple Update	0.500	2	0	1.000	0.500
RT_R.1 Determine Fir	Computation	2.000	2	1	2.377	4.755
RT_CS.10 Get Attribu	Simple Update	0.500	3	2	5.805	2.902
RT_CS.5 Subshell Bou	Arithmetic	0.700	3	2	5.805	4.063
RT_CS.6 Previous Sub	Arithmetic	0.700	1	2	2.377	1.664
RT_CS.10 Get Attribu	Simple Update	0.500	2	1	2.377	1.189
RT_CS.31 generate RT	Data Direction	0.300	3	3	7.755	2.326
RT_R.4 Increment OPL	Arithmetic	0.700	3	1	4.000	2.800
RT_R.2 Current Subsh	Arithmetic	0.700	1	1	1.000	0.700
RT_R.5 Determine Sec	Computation	2.000	2	1	2.377	4.755
RT_S.10 Get Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_R.10 Get Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_R.6 Determine Tot	Arithmetic	0.700	1	1	1.000	0.700
RT_R.7 Determine Pha	Computation	2.000	3	1	4.000	8.000
RT_R.8 Determine Fin	Computation	2.000	3	1	4.000	8.000
RT_R.10 Get Attribut	Simple Update	0.500	2	1	2.377	1.189
Total Bang for DFD RAY.1						47.921
RT_RST.1 create RAY_	Initiation	1.000	2	2	4.000	4.000
Total Bang for DFD RAY_SUBSHELL_TRAVERSAL.1						4.000
RT_UI.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_UI.12 Open Macro	Initiation	1.000	1	1	1.000	1.000
RT_UI.33 generate RT	Data Direction	0.300	1	1	1.000	0.300
Total Bang for DFD USER_INTERFACE.48						3.300
RT_UI.35 generate RT	Data Direction	0.300	2	1	2.377	0.713
RT_UI.32 generate RT	Data Direction	0.300	3	2	5.805	1.741
Total Bang for DFD RAYTRACE.1						21.545
RT_DP.10 Get Attribu	Simple Update	0.500	1	1	1.000	0.500
RT_RL.10 Get Attribu	Simple Update	0.500	2	1	2.377	1.189
RT_S.10 Get Attribut	Simple Update	0.500	2	0	1.000	0.500
RT_RL.2 Constrain Li	Verification	1.000	4	2	7.755	7.755
RT_RL.3 Set Attribut	Simple Update	0.500	3	3	7.755	3.877
Total Bang for DFD RAYTRACE_LIMITS.1						13.821
RT_RL.1 create RAYTR	Initiation	1.000	1	2	2.377	2.377
Total Bang for DFD RAYTRACE_LIMITS.2						2.377
RT_EB.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD SCALE_BAR.1						0.500
RT_EB.1 create SCALE	Initiation	1.000	3	5	12.000	12.000
RT_EB.33 generate RT	Data Direction	0.300	1	1	1.000	0.300
Total Bang for DFD SCALE_BAR.2						12.300
RT_EB.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_EB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_EB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_EB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_EB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_EB.3 Set Attribut	Data Direction	0.300	2	1	2.377	0.713
RT_EB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_EB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_EB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_EB.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_EB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_EB.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_EB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_EB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_EB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_EB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_EB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_EB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_EB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_EB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_EB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_EB.33 generate RT	Data Direction	0.300	2	1	2.377	0.713
RT_EB.4 Examine Curr	Data Direction	0.300	1	11	21.510	6.453
RT_EB.2 Get Current	Edit	0.800	2	1	2.377	1.902
Total Bang for DFD SCALE_BAR.3						25.602
RT_NB.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_NB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_NB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_NB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_NB.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_NB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_NB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_NB.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_NB.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_NB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_NB.3 Set Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_NB.33 generate RT	Data Direction	0.300	2	1	2.377	0.713
RT_NB.4 Examine Curr	Data Direction	0.300	1	11	21.510	6.453
RT_NB.2 Get Current	Edit	0.800	2	1	2.377	1.902
Total Bang for DFD SCAN_BAR.1						26.889
RT_NB.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD SCAN_BAR.2						0.500

RT_NB.1 create SCAN_	Initiation	1.000	3	5	12.000	12.000
RT_NB.33 generate RT	Data Direction	0.300	1	1	1.000	0.300
Total Bang for DFD SCAN_BAR.3						12.300
RT_SD.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD SET_DENSITY.4						0.500
RT_SD.1 create SET_	Initiation	1.000	0	0	0.000	0.000
Total Bang for DFD SET_DENSITY.5						0.000
RT_S.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD SHOCK.1						0.500
RT_S.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
RT_S.7 Initialise Gr	Device Management	2.500	3	1	4.000	10.000
RT_S.10 Get Attribut	Simple Update	0.500	4	3	9.826	4.913
Total Bang for DFD SHOCK.11						15.413
RT_S.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
RT_S.12 Plot Density	Display	1.800	3	1	4.000	7.200
RT_S.10 Get Attribut	Simple Update	0.500	4	3	9.826	4.913
Total Bang for DFD SHOCK.12						12.613
RT_S.42 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_S.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
RT_S.20 Close Graphi	Device Management	2.500	1	1	1.000	2.500
Total Bang for DFD SHOCK.13						3.300
RT_S.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
RT_S.13 Plot Intensi	Display	1.800	3	1	4.000	7.200
RT_S.10 Get Attribut	Simple Update	0.500	4	3	9.826	4.913
Total Bang for DFD SHOCK.14						12.613
RT_S.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
RT_S.21 delete SHOCK	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD SHOCK.15						1.000
RT_S.3 Set Attribute	Simple Update	0.500	1	1	1.000	0.500
RT_S.10 Get Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_RL.10 Get Attribu	Simple Update	0.500	4	1	5.805	2.902
RT_R.31 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_R.10 Get Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_S.8 Increment Ele	Computation	2.000	4	0	4.000	8.000
RT_S.9 Determine Lig	Computation	2.000	3	1	4.000	8.000
RT_S.11 Smooth and L	Computation	2.000	2	1	2.377	4.755
Total Bang for DFD SHOCK.6						26.835
RT_S.3 Set Attribute	Simple Update	0.500	2	2	4.000	2.000
RT_S.10 Get Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_S.16 Save Intensi	Output Generation	1.000	2	1	2.377	2.377
RT_S.42 generate RT_	Data Direction	0.300	1	1	1.000	0.300
Total Bang for DFD SHOCK.7						5.866
RT_S.3 Set Attribute	Simple Update	0.500	2	2	4.000	2.000
RT_S.10 Get Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_S.15 Save Density	Output Generation	1.000	2	1	2.377	2.377
RT_S.42 generate RT_	Data Direction	0.300	1	1	1.000	0.300
Total Bang for DFD SHOCK.8						5.866
RT_M.1 Get Menu Sele	Edit	0.800	2	2	4.000	3.200
RT_UI.52 generate RT	Data Direction	0.300	2	1	2.377	0.713
RT_UI.54 generate RT	Data Direction	0.300	2	1	2.377	0.713
Total Bang for DFD SYSTEM.1						4.626
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_UI.19 delete USER	Simple Update	0.500	1	1	1.000	0.500
RT_S.43 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_I.43 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_UI.10 Get Attribu	Simple Update	0.500	3	2	5.805	2.902
Total Bang for DFD USER_INTERFACE.23						4.502
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_S.34 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_S.36 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_UI.10 Get Attribu	Simple Update	0.500	2	1	2.377	1.189
RT_S.35 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_UI.36 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_S.37 generate RT_	Data Direction	0.300	1	1	1.000	0.300
Total Bang for DFD USER_INTERFACE.29						3.189
RT_UI.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_UI.35 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_S.32 generate RT_	Data Direction	0.300	0	2	1.000	0.300
RT_UI.10 Get Attribu	Simple Update	0.500	3	0	2.377	1.189
Total Bang for DFD USER_INTERFACE.30						3.789
RT_UI.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_UI.10 Get Attribu	Simple Update	0.500	3	0	2.377	1.189
RT_S.41 generate RT_	Data Direction	0.300	0	2	1.000	0.300
Total Bang for DFD USER_INTERFACE.31						3.789
RT_UI.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_UI.10 Get Attribu	Simple Update	0.500	3	0	2.377	1.189
RT_S.39 generate RT_	Data Direction	0.300	0	2	1.000	0.300
Total Bang for DFD USER_INTERFACE.32						3.789
RT_UI.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_UI.10 Get Attribu	Simple Update	0.500	3	0	2.377	1.189
RT_S.40 generate RT_	Data Direction	0.300	0	2	1.000	0.300
Total Bang for DFD USER_INTERFACE.33						3.789
RT_UI.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_I.32 generate RT_	Data Direction	0.300	0	2	1.000	0.300
RT_UI.10 Get Attribu	Simple Update	0.500	3	0	2.377	1.189

Total Bang for DFD USER_INTERFACE.34						3.789
RT_UI.3 Set Attribut	Simple Update	0.500	4	0	4.000	2.000
RT_I.33 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_I.34 generate RT_	Data Direction	0.300	5	0	5.805	1.741
RT_UI.10 Get Attribu	Simple Update	0.500	2	1	2.377	1.189
RT_NB.32 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_I.37 generate RT_	Data Direction	0.300	0	1	0.500	0.150
RT_I.38 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_I.36 generate RT_	Data Direction	0.300	0	2	1.000	0.300
RT_UI.45 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_I.10 Get Attribut	Simple Update	0.500	6	5	19.027	9.513
Total Bang for DFD USER_INTERFACE.35						16.094
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_I.33 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_I.34 generate RT_	Data Direction	0.300	5	0	5.805	1.741
RT_UI.10 Get Attribu	Simple Update	0.500	3	3	7.755	3.877
RT_I.35 generate RT_	Data Direction	0.300	2	2	4.000	1.200
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_S.10 Get Attribut	Simple Update	0.500	2	1	2.377	1.189
RT_I.10 Get Attribut	Simple Update	0.500	5	4	14.265	7.132
Total Bang for DFD USER_INTERFACE.36						16.240
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_I.33 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_I.34 generate RT_	Data Direction	0.300	5	0	5.805	1.741
RT_UI.10 Get Attribu	Simple Update	0.500	2	2	4.000	2.000
RT_I.38 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_UI.45 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_I.10 Get Attribut	Simple Update	0.500	5	4	14.265	7.132
Total Bang for DFD USER_INTERFACE.39						12.274
RT_UI.10 Get Attribu	Simple Update	0.500	3	2	5.805	2.902
RT_UI.2 Update Indic	Simple Update	0.500	3	2	5.805	2.902
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD USER_INTERFACE.4						6.305
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_UI.24 Display Sys	Output Generation	1.000	1	1	1.000	1.000
Total Bang for DFD USER_INTERFACE.40						1.800
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_UI.5 Launch Syste	Output Generation	1.000	1	1	1.000	1.000
Total Bang for DFD USER_INTERFACE.41						1.800
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_UI.6 Swap Environ	Storage Management	1.000	1	1	1.000	1.000
RT_UI.7 Get Environm	Storage Management	1.000	2	1	2.377	2.377
RT_UI.6 Swap Environ	Storage Management	1.000	1	1	1.000	1.000
RT_UI.8 Initialise M	Initiation	1.000	1	1	1.000	1.000
RT_UI.9 Close Menu S	Simple Update	0.500	1	1	1.000	0.500
Total Bang for DFD USER_INTERFACE.42						6.678
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_UI.53 generate RT	Data Direction	0.300	2	1	2.377	0.713
RT_UI.36 generate RT	Data Direction	0.300	2	1	2.377	0.713
RT_UI.4 Has operator	Data Direction	0.300	1	2	2.377	0.713
Total Bang for DFD USER_INTERFACE.43						2.640
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_S.38 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_UI.10 Get Attribu	Simple Update	0.500	2	1	2.377	1.189
Total Bang for DFD USER_INTERFACE.44						2.289
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_UI.53 generate RT	Data Direction	0.300	2	1	2.377	0.713
RT_UI.45 generate RT	Data Direction	0.300	2	1	2.377	0.713
RT_UI.4 Has operator	Data Direction	0.300	1	2	2.377	0.713
Total Bang for DFD USER_INTERFACE.45						2.640
RT_UI.3 Set Attribut	Simple Update	0.500	1	1	1.000	0.500
RT_UI.53 generate RT	Data Direction	0.300	1	1	1.000	0.300
RT_I.42 generate RT_	Data Direction	0.300	1	1	1.000	0.300
RT_UI.10 Get Attribu	Simple Update	0.500	2	1	2.377	1.189
Total Bang for DFD USER_INTERFACE.46						2.289
RT_UI.3 Set Attribut	Simple Update	0.500	2	2	4.000	2.000
RT_UI.12 Open Macro	Initiation	1.000	1	1	1.000	1.000
RT_UI.33 generate RT	Data Direction	0.300	1	1	1.000	0.300
Total Bang for DFD USER_INTERFACE.48						3.300

D.17.6 Software Metrics and Neural Networks

Neural networks find application even in the field of software metrics. As the metrics in this appendix indicate, the functional complexity of a software model can be measured by static complexity metrics of the program text. These metrics may be used to classify software modules into different fault-prone categories. Karunanithi [Kar94] notes that this difficult software engineering problem may be tackled by neural network classifiers. Such systems can play a role during the software development cycle of subsequent releases or similar software products as testing can be concentrated on those modules identified as fault-prone. Karunanithi selected a random sample of 390 high level (Pascal, Fortran, PL/M) routines from medical imaging software consisting of 400 000 lines of code in 4500 modules. Eleven software metrics were extracted for each module and presented as input to the neural network. The number of program changes for each module was used as an indication of software fault and used to classify the modules into low fault-prone ($\leq$ 1 change) and high fault-prone ($\geq$ 10 changes) groups. The network achieved an accuracy of 80-95% when classifying various test-sets.

D.18 References for Appendix D

[Bhu84] Bhur R.J.A.
System Design with Ada
Prentice Hall, Englewood Cliffs, N.J. (1984).

[Boe81] Boehm B.W.
Software Engineering Economics,
Prentice Hall, Englewood Cliffs N.J. (1981).

[DeM78] DeMarco T.
Structured Analysis and System Specification,
Yourdon Press, New York (1978).

[DeM82] DeMarco T.
Controlling Software Projects, Management Measurement & Estimation, 114-128, 268-272.
1'st edition, Prentice Hall, Englewood Cliffs N.J. (1982).

[Hil90] Hill R.
Object-Orientated Design in Ada: A Transformational Approach Based on Object-Orientated Analysis
Project Technology Internal Paper, Berkeley, CA (1990).

[IEE92] IEEE Std 1061-1992
IEEE Standard for a Software Quality Metrics Methodology, 24, 27-30
Institute of Electrical and Electronics Engineers, New York (1992).

[Kar94] Karunanithi N.
"Identifying Fault-Prone Software Modules Using Feed-Forward Networks: A Case Study", in
Advances in Neural Information Processing Systems 6 (1994) 793,
edited by Cowan J.D., Tesauro G. and Alspector J., Morgan Kaufmann Publishers, San Francisco 1994.

[Kol94] Kolen J.F.
"Fool's Gold: Extracting Finite State Machines From Recurrent Network Dynamics", in *Advances in Neural Information Processing Systems* 6 (1994) 501, edited by Cowan J.D., Tesauro G. and Alspector J., Morgan Kaufmann Publishers, San Francisco 1994.

[Mil88] Mills E.E.
Software Metrics SEI Curriculum Module SEI-CM-12-1.1, 7-9, 12
Software Engineering Institute, Carnegie Mellon University (1988).

[Moo56] Moore E.F.
"Gadanken-experiments on Sequential Machines", in
Automata Studies, Princeton University Press, Princeton, N.J. (1956).

[Shl88] Shlaer S. & Mellor S.J.
Object-Orientated Systems Analysis: Modeling the World in Data
Prentice Hall, Englewood Cliffs, N.J. (1988).

[Shl92] Shlaer S. & Mellor S.J.
Object Lifecycles: Modeling the World in States
Prentice Hall, Englewood Cliffs, N.J. (1992).

Appendix E Source Code Listings

E.1 Module raytrace

```
{---------------------------------***-------------------------------------}
{ Program     : RayTrace.pas                                              }
{ Language    : Turbo Pascal v7.0                                         }
{ Function    : Integrated raytrace, neural net, image processing user    }
{               interface.                                                }
{ Course      : MSc (Computer Science) 1993 - University of Natal (Durban)}
{ Author      : Trevor Kistan                                             }
{ Last Update : 29-12-1994                                                }
{---------------------------------***-------------------------------------}

program RayTrace;

uses
  Overlay, RayOver, Dos, Crt, Objects, Drivers, Views, Menus, Dialogs, App,
  Memory, StdDlg, MsgBox, Gadgets, Calc, HelpFile, Shocks, General, Images,
  Tokens;
 {, Status}

  {$O Objects}
  {$O Views}
  {$O Menus}
  {$O Dialogs}
  {$O App}
  {$O Memory}
  {$O StdDlg}
  {$O MsgBox}
  {$O Gadgets}
  {$O Calc}
  {$O HelpFile}
  {$O Shocks}
  {$O General}
  {$O Images}


const
  cmAbout                     = 102;
  cmCalculator                = 103;
  cmHelp                      = 104;
  cmSystemDosShell            = 105;
  cmSystemExit                = 106;
  cmSystemInfo                = 107;
  cmGetFileName               = 108;
  cmShockFireRays             = 110;
  cmShockLoadParameters       = 111;
  cmShockSaveParameters       = 112;
  cmShockSaveRawIntensities   = 113;
  cmShockSaveRawDensities     = 114;
  cmShockDisplay              = 115;
  cmShockRunMacroFile         = 116;
  cmImageLoadFile             = 117;
  cmImageDetermineScale       = 118;
  cmImageExtractIntensity     = 119;
  cmImageSaveRawIntensities   = 120;
  cmImageSaveIntensityPeaks   = 121;
  cmImageDisplay              = 122;
  cmShockDisplayExtrema       = 123;

type
  Checkbox = Word;

type
  RadioButton = Word;

type
  Inputline = String[20];

type
  SetDensityDataRec = record
    Smooth    : RadioButton;
```

```
    VacuumX  : Inputline;
    VacuumY  : Inputline;
    TailX    : Inputline;
    TailY    : Inputline;
    SaddleX  : Inputline;
    SaddleY  : Inputline;
    RearX    : Inputline;
    RearY    : Inputline;
    FrontX   : Inputline;
    FrontY   : Inputline;
    FireRays : Checkbox;
    RaysFrom : Inputline;
    RaysTo   : Inputline;
  end; {record}

type
  PSetDensityDataRec = ^SetDensityDataRec;

type
  ImageExtractIntensityRec = record
    LinearizeField : Word;  {Radiobuttons}
    WindowSize     : Word;  {Radiobuttons}
    ClearPlot      : Word;  {Checkbox}
  end; {record}

type
  PImageExtractIntensityRec = ^ImageExtractIntensityRec;

{-----------------------------------------------}
{ The user interface - a turbo vision application }
{-----------------------------------------------}

type
  TRay = object( TApplication )
    Clock                 : PClockView;
    Heap                  : PHeapView;
    SetDensityData        : PSetDensityDataRec;
    ExtractIntensityData  : PImageExtractIntensityRec;
    Shock                 : Shocks.PShock;
    Image                 : Images.PImage;
    ImageFileLoaded       : Boolean;
    {------}
    {system}
    {------}
    constructor Init;
    destructor Done; virtual;
    procedure HandleEvent(var Event : TEvent); virtual;
    procedure GetEvent(var Event: TEvent); virtual;
    function  GetPalette: PPalette; virtual;
    procedure InitMenuBar; virtual;
    procedure InitStatusLine; virtual;
    procedure GetFileName(var FileName             : PathStr;
                              WildCard, Title      : string;
                              HelpContext          : Word;
                          var GetFileNameCancelled : Boolean); virtual;
    procedure About; virtual;
    procedure Calculator; virtual;
    procedure Idle; virtual;
    procedure SystemDosShell; virtual;
    procedure SystemExit; virtual;
    procedure LeaveTurboVision;
    procedure ResumeTurboVision;
    {-----}
    {shock}
    {-----}
    function  SetDensityDialog : PDialog; virtual;
    procedure SetDensity; virtual;
    procedure ShockLoadParameters;  virtual;
    procedure ShockSaveParameters; virtual;
    procedure ShockSaveRawIntensities; virtual;
    procedure ShockSaveRawDensities; virtual;
    procedure ShockDisplay; virtual;
    procedure ShockDisplayExtrema; virtual;
    procedure ShockRunMacroFile; virtual;
    procedure AutoShock( Filename : PathStr ); virtual;
```

```
    {-----}
    {image}
    {-----}
    function  ExtractIntensityDialog : PDialog; virtual;
    procedure ImageLoadFile; virtual;
    procedure ImageExtractIntensity; virtual;
    procedure ImageDetermineScale; virtual;
    procedure ImageSaveRawIntensities; virtual;
    procedure ImageSaveIntensityPeaks; virtual;
    procedure ImageDisplay; virtual;

    {network}


  end; {object}

type
  PRay = ^TRay;

{--------------------------------------------------------------}
{ Define a dialog object that inherets TDialog in case we want }
{ to add as yet unseen functionality sometime in the future.   }
{--------------------------------------------------------------}


type
  TMenuDialog = object( TDialog )
  end; {object}

type
  PMenuDialog = ^TMenuDialog;

var
  Ray            : TRay;

{-----------------------------------***-----------------------------------}

function Confirm(Msg : String) : Word;

{confimation dialog - returns button pressed}

var
  D       : PDialog;
  Control : Word;
  R       : TRect;

begin
  R.Assign(0, 0, 49, 15);
  D := New(PDialog, Init(R, 'Confirmation'));
  with D^ do begin
    Options := Options or ofCentered;
    R.Grow(-1, -1);
    Dec(R.B.Y, 3);
    R.Assign(5, 2, 45, 10);
    Insert(New(PStaticText, Init(R, Msg)));
    R.Assign(15, 11, 25, 13);
    Insert(New(PButton, Init(R, '~N~o', cmNo, bfNormal)));
    R.Assign(3,  11, 13, 13);
    Insert(New(PButton, Init(R, '~Y~es', cmYes, bfDefault)));
  end; {with}
  Control := Desktop^.ExecView(D);
  Confirm := Control;
  Dispose(D, Done);
end; {Confirm}

{-----------------------------------***-----------------------------------}

procedure Message( Title, Msg : String; DelayInterval : Word );

var
  D       : PDialog;
  Control : Word;
  R       : TRect;

begin
```

```
  R.Assign(0, 0, 29, 8);
  D := New( PDialog, Init( R, Title ));
  with D^ do begin
    Options := Options or ofCentered;
    R.Grow(-1, -1);
    Dec(R.B.Y, 3);
    R.Assign(5, 2, 25, 4);
    Insert(New(PStaticText, Init(R, Msg)));
  end; {with}
  Desktop^.Insert(D);
  Delay( DelayInterval );
  DeskTop^.Delete( D );
  Dispose(D, Done);
end; {Confirm}


{-----------------------------------***-----------------------------------}

function CalcHelpName: PathStr;

{ get the name of the help file }

var
  EXEName : PathStr;
  Dir     : DirStr;
  Name    : NameStr;
  Ext     : ExtStr;

begin
  if Lo(DosVersion) >= 3
  then begin
    EXEName := ParamStr(0)
  end {then}
  else begin
    EXEName := FSearch('RayTrace.exe', GetEnv('PATH'));
  end; {else}
  FSplit(EXEName, Dir, Name, Ext);
  if Dir[Length(Dir)] = '\' then begin
    Dec(Dir[0]);
  end; {then}
  CalcHelpName := FSearch('RayHelp.hlp', Dir);
end; {CalcHelpFileName}


{-----------------------------------***-----------------------------------}
{ Initialises the application.                                            }
{-----------------------------------***-----------------------------------}

constructor TRay.Init;

var
  R : TRect;

begin
  {-------------------------}
  {call ancestor's init method}
  {-------------------------}
  TApplication.Init;

  {------------------------------}
  {register turbo vision units used}
  {------------------------------}
  RegisterObjects;
  RegisterViews;
  RegisterMenus;
  RegisterDialogs;
  RegisterApp;
  RegisterCalc;
  RegisterHelpFile;

  {----------------------}
  {set initial help context}
  {----------------------}
{  HelpCtx := hcUserInterfaceHelp;}
```

```
  {-------------------------}
  {insert clock in staus line}
  {-------------------------}
  GetExtent(R);
  R.A.X := R.B.X - 9;
  R.B.Y := R.A.Y + 1;
  Clock := New(PClockView, Init(R));
  Insert(Clock);

  {------------------------------}
  {insert heap size in status line}
  {------------------------------}
  GetExtent(R);
  Dec(R.B.X);
  R.A.X := R.B.X - 9;
  R.A.Y := R.B.Y - 1;
  Heap := New(PHeapView, Init(R));
  Insert(Heap);

  {------------------------------}
  {set up the shock              }
  {------------------------------}
  New( SetDensityData );
  Shock := New( Shocks.PShock, Init );

  with SetDensityData^ do begin
    Smooth := 0;
    with Shock^.DensityParameters do begin
      VacuumX := General.WordToStr( Vacuum.x );
      VacuumY := General.DoubleToStr( Vacuum.y, 10, 6 );
      TailX   := General.WordToStr( Tail.x );
      TailY   := General.DoubleToStr( Tail.y, 10, 6 );
      SaddleX := General.WordToStr( Saddle.x );
      SaddleY := General.DoubleToStr( Saddle.y, 10, 6 );
      RearX   := General.WordToStr( Rear.x );
      RearY   := General.DoubleToStr( Rear.y, 10, 6 );
      FrontX  := General.WordToStr( Front.x );
      FrontY  := General.DoubleToStr( Front.y, 10, 6 );
    end; {with}
    FireRays := 0;
    with Shock^.RaytraceLimits do begin
      RaysFrom := General.WordToStr( FireRaysFrom );
      RaysTo   := General.WordToStr( FireRaysTo );
      CurrentRay := 0;
    end; {with}
  end; {with}

  {-------------------------------}
  {Set up the image               }
  {-------------------------------}
  New( ExtractIntensityData );
  Image := New( Images.PImage, Init );
  ImageFileLoaded := False;

  with ExtractIntensityData^ do begin
    LinearizeField := 0;
    WindowSize     := 0;
    ClearPlot      := 1;
  end; {with}

  About;

end; {TRay.Init}

{-----------------------------------***-----------------------------------}

destructor TRay.Done;

{Dispose of any dynamic structures and quit the program.}

begin
  Shock^.Done;
  Dispose( SetDensityData );
  {--------------------------}
  {Call ancestor's done method}
```

```
  {--------------------------}
  TApplication.Done;
  {-----------------------------------------------------}
  {inform user that application was successfully terminated}
  {-----------------------------------------------------}
  Writeln('RayTrace session successfully terminated.');
end; {TRay.Done}

{-----------------------------------***------------------------------------}

procedure TRay.Idle;

{Use idle time to update heap and clock displays.}

begin
  TApplication.Idle;
  Clock^.UpDate;
  Heap^.UpDate;
end; {TRay.Idle}


{-----------------------------------***------------------------------------}

function TRay.GetPalette: PPalette;

{set up color palette for application needed for help screens}

const
  CNewColor      = CColor + CHelpColor;
  CNewBlackWhite = CBlackWhite + CHelpBlackWhite;
  CNewMonochrome = CMonochrome + CHelpMonochrome;
  P : array[apColor..apMonochrome] of string[Length(CNewColor)] =
      (CNewColor, CNewBlackWhite, CNewMonochrome);

begin
  GetPalette := @P[AppPalette];
end; {TRay.GetPalette}



procedure TRay.GetEvent( var Event: TEvent );

{overide GetEvent to cater for context-sensitive help}

const
  HelpInUse : Boolean = False;

var
  W         : PWindow;
  HFile     : PHelpFile;
  HelpStrm  : PDosStream;

                    {----------------***---------------}

procedure ACommandEvent;
begin
  if (Event.Command = cmHelp) and not HelpInUse
    then begin
      HelpInUse := True;
      HelpStrm := New(PDosStream, Init(CalcHelpName, stOpenRead));
      HFile := New(PHelpFile, Init(HelpStrm));
      if HelpStrm^.Status <> stOk
        then begin
          MessageBox('Could not open help file.', nil, mfError + mfOkButton);
          Dispose(HFile, Done);
        end {then}
        else begin
          W := New(PHelpWindow,Init(HFile, GetHelpCtx));
          if ValidView(W) <> nil
            then begin
              {-------------------}
              {display help window}
              {-------------------}
              ExecView(W);
              Dispose(W, Done);
```

```
            end; {then}
          ClearEvent(Event);
        end; {else}
      HelpInUse := False;
    end; {then}
end; {ACommandEvent}

                    {----------------***---------------}

procedure AMouseEvent;
begin
  if Event.Buttons <> 1
    then begin
      Event.What := evNothing;
    end; {then}
end; {AMouseEvent}

                    {----------------***---------------}

begin
  {----------------------}
  {Call ancestor's method}
  {----------------------}
  TApplication.GetEvent( Event );
  case Event.What of
    evCommand   : ACommandEvent;
    evMouseDown : AMouseEvent;
  end; {case}
end; {TRay.GetEvent}

{------------------------------------***------------------------------------}

procedure TRay.HandleEvent(var Event: TEvent);

{run appropiate procedure for command selected by user}

begin
  {--------------------}
  {Call ancestor method}
  {--------------------}
  TApplication.HandleEvent( Event );
  if Event.What = evCommand then begin
     case Event.Command of
        cmAbout                    : About;
        cmCalculator               : Calculator;
        cmSystemDosShell           : SystemDosShell;
        cmSystemExit               : SystemExit;

        cmShockFireRays            : SetDensity;
        cmShockLoadParameters      : ShockLoadParameters;
        cmShockSaveParameters      : ShockSaveParameters;
        cmShockSaveRawIntensities  : ShockSaveRawIntensities;
        cmShockSaveRawDensities    : ShockSaveRawDensities;
        cmShockDisplay             : ShockDisplay;
        cmShockDisplayExtrema      : ShockDisplayExtrema;
        cmShockRunMacroFile        : ShockRunMacroFile;

        cmImageLoadFile            : ImageLoadFile;
        cmImageDetermineScale      : ImageDetermineScale;
        cmImageExtractIntensity    : ImageExtractIntensity;
        cmImageSaveRawIntensities  : ImageSaveRawIntensities;
        cmImageSaveIntensityPeaks  : ImageSaveIntensityPeaks;
        cmImageDisplay             : ImageDisplay;

        else                         Exit;
     end; {case}
     ClearEvent( Event );
  end; {if}
end; {TRay.HandleEvent}

{------------------------------------***------------------------------------}

procedure TRay.InitMenuBar;

{---------------------------------------------------------------------}
```

```
{set up the menu (letters within ~~ are higlighted)                   }
{parameters to NewItem:                                                }
{    title disp, key disp, keybinding, command generated, helpcontext }
{----------------------------------------------------------------------}

var
  R : TRect;

begin
  GetExtent( R );
  R.B.Y := R.A.Y + 1;
  MenuBar := New(PMenuBar, Init( R, NewMenu(
    NewSubMenu('~'#240'~', hcNoContext, NewMenu(
      NewItem('~A~bout...', 'Alt-A', kbAltA, cmAbout, hcNoContext,
      NewItem('~C~alculator', 'Alt-C', kbAltC, cmCalculator, hcNoContext,
      NewItem('~H~elp', 'Alt-H', kbAltH, cmHelp, hcNoContext,
      nil)))),
    NewSubMenu('~S~ystem', hcNoContext, NewMenu(
      NewItem('~I~nfo...','Alt-I', kbAltI, cmSystemInfo, hcNoContext,
      NewLine(
      NewItem('~D~os Shell', 'Alt-D', kbAltD, cmSystemDosShell, hcNoContext,
      NewItem('~E~xit', 'Alt-X', kbAltX, cmSystemExit, hcNoContext,
      nil))))),
    NewSubMenu('~R~aytrace', hcNoContext, NewMenu(
      NewItem('~F~ire Rays...', 'F1', kbF1, cmShockFireRays, hcNoContext,
      NewItem('~S~ave Parameters...', 'F2', kbF2, cmShockSaveParameters, hcNoContext,
      NewItem('~L~oad Parameters...', 'F3', kbF3, cmShockLoadParameters, hcNoContext,
      NewItem('Save Raw ~D~ensities...', 'F4', kbF4, cmShockSaveRawDensities, hcNoContext,
      NewItem('Save Raw ~I~ntensities...', 'F5', kbF5, cmShockSaveRawIntensities, hcNoContext,
      NewItem('Display Shock', 'F6', kbF6, cmShockDisplay, hcNoContext,
      NewItem('Display Extrema', 'F7', kbF7, cmShockDisplayExtrema, hcNoContext,
      NewLine(
      NewItem('Run ~M~acro File', 'F8', kbF8, cmShockRunMacroFile, hcNoContext,
      nil))))))))))),
    NewSubMenu('~N~etwork', hcNoContext, NewMenu(
      NewItem('~C~ompletion Test...', 'Alt-F1', kbAltF1, cmAbout, hcNoContext,
      NewItem('~P~ermuted Training', 'Alt-F2', kbAltF2, cmAbout, hcNoContext,
      NewItem('~S~equential Training', 'Alt-F3', kbAltF3, cmAbout, hcNoContext,
      NewItem('~T~est...', 'Alt-F4', kbAltF4, cmAbout, hcNoContext,
      NewItem('Test ~A~ll', 'Alt-F5', kbAltF5, cmAbout, hcNoContext,
      NewItem('T~r~ial', 'Alt-F6', kbAltF6, cmAbout, hcNoContext,
      NewItem('~F~orced-Choice...', 'Alt-F7', kbAltF7, cmAbout, hcNoContext,
      nil)))))))),
    NewSubMenu('~I~mage', hcNoContext, NewMenu(
      NewItem('~L~oad Image...', 'Ctrl-F1', kbCtrlF1, cmImageLoadFile, hcNoContext,
      NewItem('Determine ~S~cale', 'Ctrl-F2', kbCtrlF2, cmImageDetermineScale, hcNoContext,
      NewItem('~E~xtract Intensity...','Ctrl-F3',kbCtrlF3,cmImageExtractIntensity,hcNoContext,
      NewItem('Save   ~R~aw   Intensities...',   'Ctrl-F4',   kbCtrlF4,   cmImageSaveRawIntensities,
hcNoContext,
      NewItem('Save   Intensity   ~P~eaks...',   'Ctrl-F5',   kbCtrlF5,   cmImageSaveIntensityPeaks,
hcNoContext,
      NewItem('~D~isplay Image', 'Ctrl-F6', kbCtrlF6, cmImageDisplay, hcNoContext,
      nil))))))),
    nil)))))
  )));
end; {TRay.InitMenuBar}

{-----------------------------------***-----------------------------------}

procedure TRay.InitStatusLine;

{set up the status line - the keybindings are important for keyboard}
{users.  There is not enough space on the status line to display all}
{the keybindings                                                      }

var
  R : TRect;

begin
  GetExtent(R);
  R.A.Y := R.B.Y - 1;
  StatusLine := New(PStatusLine, Init(R,
    NewStatusDef(0, $FFFF,
      NewStatusKey('~F1~ Help', kbF1, cmHelp,
      NewStatusKey('~F10~ Menu', kbF10, cmMenu,
```

```
      NewStatusKey('~Alt-X~ Exit', kbAltX, cmSystemExit,
      nil))),
    nil)
  ));
end; {TRay.InitStatusLine}

{-------------------------------------***--------------------------------------}

procedure TRay.GetFileName(var FileName : PathStr; WildCard, Title : string;
                           HelpContext : Word;
                           var GetFileNameCancelled : Boolean);

{---------------------------------------------------------------------------}
{ a utility routine which sets up a file dialog box and returns a selected }
{ or specified filename to the calling routine                            }
{---------------------------------------------------------------------------}

var
  D          : PFileDialog;

begin
  GetFileNameCancelled := False;
  D := New(PFileDialog, Init(WildCard, Title, '~N~ame', fdOpenButton + fdHelpButton, 100));
  {----------------------------}
  {select help window to display}
  {----------------------------}
  D^.HelpCtx := HelpContext;
  if ValidView(D) <> nil
    then begin
      {---------------}
      {run file dialog}
      {---------------}
      if Desktop^.ExecView(D) <> cmCancel
        then begin
          {-------------}
          {get file name}
          {-------------}
          D^.GetFileName(FileName)
        end  {then}
        else begin
          {---------------------------------}
          {else indicate selection cancelled}
          {---------------------------------}
          GetFileNameCancelled := True;
        end; {else}
      Dispose(D, Done);
    end; {then}
end; {TRay.GetFileName}

{-------------------------------------***--------------------------------------}

procedure TRay.About;

{displays information about RayTrace}

var
  D       : PDialog;
  Control : Word {PView};
  R       : TRect;

begin
  R.Assign(0, 0, 61, 13);
  D := New(PDialog, Init(R, 'About'));
{  D^.HelpCtx := hcHelp; }
  with D^ do begin
    Options := Options or ofCentered;
    R.Grow(-1, -1);
    Dec(R.B.Y, 3);
    Insert(New(PStaticText, Init(R,
      #13 + {newline, centered}
      ^C'A Neural Network Assisted Raytracer for'#13 +
      #13 + {newline, centered}
      ^C'Refractive Fringe Diagnostics of Spherical Shocks in Air'#13 +
      #13 + {newline, centered}
      ^C'Depts of Computer Science & Physics - University of Natal'#13 +
```

```
      #13 + {newline, centered}
      ^C'————————————————————————————————————————————————')));
    R.Assign(30, 10, 40, 12);
    Insert(New(PButton, Init(R, '~H~elp', cmHelp, bfNormal)));
    R.Assign(18, 10, 28, 12);
    Insert(New(PButton, Init(R, '~O~k', cmOk, bfDefault)));
  end; {with}
  if ValidView(D) <> nil then begin
    Control := Desktop^.ExecView(D);
    Dispose(D, Done);
  end; {then}
end; {TRay.About}

{-----------------------------------***-------------------------------------}

procedure TRay.Calculator;

{allows turbo visions example calculator to be opened and used}

var
  P: PCalculator;

begin
  P := New(PCalculator, Init);
{  P^.HelpCtx := hcCalculator;}
  if ValidView(P) <> nil then begin
    Desktop^.Insert(P);
  end; {then}
end; {TRay.Calculator}

{-----------------------------------***-------------------------------------}

procedure TRay.SystemDosShell;

{-------------------------------------------}
{implements a dos shell - turbo vision example}
{-------------------------------------------}

begin
  DoneSysError;
  DoneEvents;
  DoneVideo;
  DoneMemory;
  SetMemTop(HeapPtr);
  PrintStr('Type EXIT to return to RayTrace...');
  SwapVectors;
  Exec(GetEnv('COMSPEC'), '');
  SwapVectors;
  SetMemTop(HeapEnd);
  InitMemory;
  InitVideo;
  InitEvents;
  InitSysError;
  Redraw;
end; {TRay.SystemDosShell}

{-----------------------------------***-------------------------------------}

procedure TRay.SystemExit;

{exits Ray2 - seeks confirmation first}

var
  D       : PDialog;
  Control : Word;
  R       : TRect;

begin
  R.Assign(0, 0, 29, 7);
  D := New(PDialog, Init(R, 'Confirmation'));
  with D^ do begin
    Options := Options or ofCentered;
    R.Grow(-1, -1);
    Dec(R.B.Y, 3);
    R.Assign(5, 2, 25, 3);
```

```
    Insert(New(PStaticText, Init(R, 'Exit from RayTrace ?')));
    R.Assign(15, 4, 25, 6);
    Insert(New(PButton, Init(R, '~N~o', cmNo, bfNormal)));
    R.Assign(3,  4, 13, 6);
    Insert(New(PButton, Init(R, '~Y~es', cmYes, bfDefault)));
  end; {with}
  Control := Desktop^.ExecView(D);
  if Control = cmYes
    then begin
      Dispose(D, Done);
      {---------------------------------------}
      {The desktop is currently modal so calling }
      {EndModal with cmQuit quits the application}
      {---------------------------------------}
      EndModal(cmQuit);
    end  {then}
    else begin
      Dispose(D, Done);
    end; {else}
end; {TRay.SystemExit}

{----------------------------------***-----------------------------------}

function  TRay.SetDensityDialog : PDialog;

var
  Dlg : PDialog;
  R   : TRect;
  Control, Labl, Histry : PView;

begin
  R.Assign( 6, 2, 71, 20 );
  New(Dlg, Init( R, 'Set Density Profile' ));

  R.Assign( 3, 4, 23, 5 );
  Control := New( PStaticText, Init( R, 'Interpolation Method' ));
  Dlg^.Insert( Control );

  R.Assign( 3, 5, 24, 9 );
  Control := New( PRadioButtons,
                Init( R,
                      NewSItem( 'Linear',
                      NewSItem( 'Lagrange',
                      NewSItem( 'Cubic Spline',
                      NewSItem( 'Curve Junctions',
                      nil ))))));
  PCluster( Control )^.Value := 0;
  Dlg^.Insert( Control );

  R.Assign( 26, 2, 32, 3 );
  Control := New( PStaticText, Init( R, 'End Of' ));
  Dlg^.Insert( Control );

  R.Assign( 26, 3, 32, 4 );
  Control := New( PStaticText, Init( R, 'Region' ));
  Dlg^.Insert( Control );

  R.Assign( 34, 2, 43, 3 );
  Control := New( PStaticText, Init( R, 'Quantised' ));
  Dlg^.Insert( Control );

  R.Assign( 34, 3, 47, 4 );
  Control := New( PStaticText, Init( R, 'Position (µm)' ));
  Dlg^.Insert( Control );

  R.Assign( 49, 2, 58, 3 );
  Control := New( PStaticText, Init( R, 'Continous' ));
  Dlg^.Insert( Control );

  R.Assign( 49, 3, 62, 4 );
  Control := New( PStaticText, Init( R, 'Density (atm)' ));
  Dlg^.Insert( Control );

  R.Assign( 34, 4, 47, 5 );
  Control := New( PInputLine, Init( R, 20 ));
```

```
Dlg^.Insert( Control );
R.Assign( 26, 4, 33, 5 );
Labl := New( PLabel, Init( R, 'Vacuum', Control ));
Dlg^.Insert( Labl );
R.Assign( 49, 4, 62, 5 );
Control := New( PInputLine, Init( R, 20 ));
Dlg^.Insert( Control );

R.Assign( 34, 5, 47, 6 );
Control := New( PInputLine, Init( R, 20 ));
Dlg^.Insert( Control );
R.Assign( 26, 5, 31, 6 );
Labl := New( PLabel, Init( R, 'Tail', Control ));
Dlg^.Insert( Labl );
R.Assign( 49, 5, 62, 6 );
Control := New( PInputLine, Init( R, 20 ));
Dlg^.Insert( Control );

R.Assign( 34, 6, 47, 7 );
Control := New( PInputLine, Init( R, 20 ));
Dlg^.Insert( Control );
R.Assign( 26, 6, 33, 7 );
Labl := New( PLabel, Init( R, 'Saddle', Control ));
Dlg^.Insert( Labl );
R.Assign( 49, 6, 62, 7 );
Control := New( PInputLine, Init( R, 20 ));
Dlg^.Insert( Control );

R.Assign( 34, 7, 47, 8 );
Control := New( PInputLine, Init( R, 20 ));
Dlg^.Insert( Control );
R.Assign( 26, 7, 31, 8 );
Labl := New( PLabel, Init( R, 'Rear', Control ));
Dlg^.Insert( Labl );
R.Assign( 49, 7, 62, 8 );
Control := New( PInputLine, Init( R, 20 ));
Dlg^.Insert( Control );

R.Assign( 34, 8, 47, 9 );
Control := New( PInputLine, Init( R, 20 ));
Dlg^.Insert( Control );
R.Assign( 26, 8, 32, 9 );
Labl := New( PLabel, Init( R, 'Front', Control ));
Dlg^.Insert( Labl );
R.Assign( 49, 8, 62, 9 );
Control := New( PInputLine, Init( R, 20 ));
Dlg^.Insert( Control );

R.Assign( 3, 11, 24, 12 );
Control := New( PCheckboxes,
                Init( R,
                      NewSItem( 'Fire Rays',
                      nil )));
PCluster( Control )^.Value := 0;
Dlg^.Insert( Control );

R.Assign( 34, 10, 44, 11 );
Control := New( PStaticText, Init( R, 'From (µm)' ));
Dlg^.Insert( Control );
R.Assign( 34, 11, 47, 12 );
Control := New( PInputLine, Init( R, 20 ));
Dlg^.Insert( Control );

R.Assign( 49, 10, 59, 11 );
Control := New( PStaticText, Init( R, 'To (µm)' ));
Dlg^.Insert( Control );
R.Assign( 49, 11, 62, 12 );
Control := New( PInputLine, Init( R, 20 ));
Dlg^.Insert( Control );

R.Assign( 1, 13, 64, 14 );
Control := New( PStaticText,
                Init( R,
                      '──────────────────────────────────────────────────────────────' ));
Dlg^.Insert( Control );
```

```
  R.Assign( 12, 15, 22, 17 );
  Control := New( PButton, Init( R, 'Ok', cmOK, bfNormal ));
  Dlg^.Insert( Control );

  R.Assign( 28, 15, 38, 17 );
  Control := New( PButton, Init( R, 'Cancel', cmCancel, bfNormal ));
  Dlg^.Insert( Control );

  R.Assign( 44, 15, 54, 17 );
  Control := New( PButton, Init( R, 'Help', cmHelp, bfNormal ));
  Dlg^.Insert( Control );

  Dlg^.SelectNext( False );
  SetDensityDialog := Dlg;
end; {TRay.SetDensityDialog}


{----------------------------------***-----------------------------------}

(*
function TRay.SetDensityDialog : PDialog;


var
  D                  : PDialog;
  R                  : TRect;
  Control            : Word;
  Put                : PView;

begin

  R.Assign( 13, 3, 64, 19 );
  D := New( PDialog, Init( R, 'Set Density Profile' ));
  with D^ do begin
    R.Assign( 3, 4, 18, 9 );
    Put := New( PCheckboxes, Init( R,
                                  NewSItem( '   Vacuum ',
                                  NewSItem( '   Tail',
                                  NewSItem( '   Saddle',
                                  NewSItem( '   Rear',
                                  NewSItem( '   Front',
                                  nil )))))));
    Insert( Put );

    R.Assign( 3, 2, 9, 3 );
    Insert( New( PStaticText, Init( R, 'Smooth' )));
    R.Assign( 3, 3, 9, 4 );
    Insert( New( PStaticText, Init( R, 'Junct.' )));

    R.Assign( 20, 2, 29, 3 );
    Insert( New( PStaticText, Init( R, 'Quantised' )));
    R.Assign( 20, 3, 33, 4 );
    Insert( New( PStaticText, Init( R, 'Position (µm)' )));

    R.Assign( 35, 2, 44, 3 );
    Insert( New( PStaticText, Init( R, 'Continous' )));
    R.Assign( 35, 3, 48, 4 );
    Insert( New( PStaticText, Init( R, 'Density (atm)' )));

    R.Assign( 11, 2, 17, 3 );
    Insert( New( PStaticText, Init( R, 'End Of' )));
    R.Assign( 11, 3, 17, 4 );
    Insert( New( PStaticText, Init( R, 'Region' )));

    R.Assign( 20, 4, 33, 5 );
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );

    R.Assign( 35, 4, 48, 5);
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );

    R.Assign( 20, 5, 33, 6 );
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );
```

```
    R.Assign( 35, 5, 48, 6 );
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );

    R.Assign( 20, 6, 33, 7 );
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );

    R.Assign( 35, 6, 48, 7 );
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );

    R.Assign( 20, 7, 33, 8  );
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );

    R.Assign( 35, 7, 48, 8 );
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );

    R.Assign( 20, 8, 33, 9 );
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );

    R.Assign( 35, 8, 48, 9 );
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );

    R.Assign( 3, 11, 18, 12 );
    Put := New( PCheckboxes, Init( R, NewSItem( 'Fire Rays', nil )));
    Insert( Put );

    R.Assign( 20, 10, 30, 11 );
    Insert( New( PStaticText, Init( R, 'From' )));

    R.Assign( 35, 10, 45, 11 );
    Insert( New( PStaticText, Init( R, 'To' )));

    R.Assign( 20, 11, 33, 12 );
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );

    R.Assign( 35, 11, 48, 12 );
    Put := New( PInputLine, Init( R, 20 ));
    Insert( Put );

    R.Assign( 6, 13, 16, 15 );
    Insert( New( PButton, Init( R, 'Ok', cmOK, bfNormal )));

    R.Assign( 21, 13, 31, 15 );
    Insert( New( PButton, Init( R, 'Cancel', cmCancel, bfNormal )));

    R.Assign( 36, 13, 46, 15 );
    Insert( New( PButton, Init( R, 'Help', cmHelp, bfNormal )));
  end; {with}

  SetDensityDialog := D;

end; {TRay.SetDensityDialog}
*)
{-----------------------------------***-----------------------------------}

procedure TRay.SetDensity;

var
  DensityProfileOK : Boolean;
  FireRaysLimitsOK : Boolean;
  FireRaysOK       : Boolean;
  DensityProfile   : Shocks.DensityParameterType;
  FireRaysLimits   : Shocks.RaytraceParameterType;
  Interpolate      : Shocks.InterpolationMethods;
  Control          : Word;

begin
```

```
DensityProfileOK := False;
FireRaysLimitsOK := False;
FireRaysOK       := False;

{-----------------------------------}
{Get the latest values for the dialog}
{-----------------------------------}

with SetDensityData^ do begin
  with Shock^.DensityParameters do begin
    VacuumX := General.WordToStr( Vacuum.x );
    VacuumY := General.DoubleToStr( Vacuum.y, 10, 6 );
    TailX   := General.WordToStr( Tail.x );
    TailY   := General.DoubleToStr( Tail.y, 10, 6 );
    SaddleX := General.WordToStr( Saddle.x );
    SaddleY := General.DoubleToStr( Saddle.y, 10, 6 );
    RearX   := General.WordToStr( Rear.x );
    RearY   := General.DoubleToStr( Rear.y, 10, 6 );
    FrontX  := General.WordToStr( Front.x );
    FrontY  := General.DoubleToStr( Front.y, 10, 6 );
    case Interpolate of
      Linear   : Smooth := 0;
      Lagrange : Smooth := 1;
      Cubic    : Smooth := 2;
      Junction : Smooth := 3;
      else       Smooth := 0;
    end; {case}
  end; {with}
  with Shock^.RaytraceLimits do begin
    RaysFrom := General.WordToStr( FireRaysFrom );
    RaysTo   := General.WordToStr( FireRaysTo );
  end; {with}
end; {with}

{-----------------------------------}
{Execute the dialog                 }
{-----------------------------------}

Control := ExecuteDialog( SetDensityDialog, SetDensityData );

{-----------------------------------}
{Read amended values back from dialog}
{-----------------------------------}

if Control <> cmCancel then begin
  with SetDensityData^ do begin
    with DensityProfile do begin
       DensityProfileOK := General.StrToLong( VacuumX, Vacuum.x )
                       and General.StrToDouble( VacuumY, Vacuum.y )
                       and General.StrToLong( TailX, Tail.x )
                       and General.StrToDouble( TailY, Tail.y )
                       and General.StrToLong( SaddleX, Saddle.x )
                       and General.StrToDouble( SaddleY, Saddle.y )
                       and General.StrToLong( RearX, Rear.x )
                       and General.StrToDouble( RearY, Rear.y )
                       and General.StrToLong( FrontX, Front.x )
                       and General.StrToDouble( FrontY, Front.y );
       case Smooth of
         0    : Interpolate := Shocks.Linear;
         1    : Interpolate := Shocks.Lagrange;
         2    : Interpolate := Shocks.Cubic;
         3    : Interpolate := Shocks.Junction;
         else   Interpolate := Shocks.Linear;
       end; {case}
    end; {with}
    with FireRaysLimits do begin
      FireRaysOK       := ( FireRays = 1 );
      FireRaysLimitsOK :=     General.StrToLong( RaysFrom, FireRaysFrom )
                          and General.StrToLong( RaysTo, FireRaysTo );
      CurrentRay       := 0;

    end; {with}
  end; {with}

  {-----------------------------------}
```

```
    {Execute the appropiate shock action}
    {---------------------------------}

    LeaveTurboVision;
    Shock^.SetDisplay;

    if DensityProfileOK then begin
       Shock^.SetDensityProfile( DensityProfile );
       Shock^.PlotDensityProfile( Crt.LightGreen );
       Shock^.DisplayParameters( Crt.LightRed );
    end; {then}
    if FireRaysLimitsOK then begin
       Shock^.SetRayTraceLimits( FireRaysLimits );
       Shock^.DisplayParameters( Crt.LightRed );
    end; {then}
    if FireRaysOK then begin
       Shock^.FireRays;
       Shock^.PlotIntensityPattern( Crt.Yellow );
    end; {then}
    repeat
      {null}
    until KeyPressed;

    Shock^.ClearDisplay;
    ResumeTurboVision;

  end; {then}

end; {TRay.SetDensity}

{----------------------------------***----------------------------------}

procedure TRay.LeaveTurboVision;

begin
  DoneSysError;
  DoneEvents;
  DoneVideo;
  DoneMemory;
end; {TRay.LeaveTurboVision}

{----------------------------------***----------------------------------}

procedure TRay.ResumeTurboVision;

begin
  InitMemory;
  InitVideo;
  InitEvents;
  InitSysError;
  Redraw;
end; {TRay.ResumeTurboVision}

{----------------------------------***----------------------------------}
procedure TRay.ShockLoadParameters;

var
  FileName  : PathStr;
  Cancelled : Boolean;

begin
  GetFileName( FileName, '*.shk', 'Shock Parameters File', hcNoContext, Cancelled);
  if not Cancelled then begin
    Shock^.LoadParameters( FileName );
  end; {then}
  ShockDisplay;
end; {TRay.ShockLoadParameters}

{----------------------------------***----------------------------------}

procedure TRay.ShockSaveParameters;

var
  FileName  : PathStr;
  Cancelled : Boolean;
```

```
begin
  GetFileName( FileName, '*.shk', 'Shock Parameters File', hcNoContext, Cancelled);
  if not Cancelled then begin
    Shock^.SaveParameters( FileName );
  end; {then}
end; {TRay.ShockSaveParameters}

{------------------------------------***------------------------------------}

procedure TRay.ShockSaveRawIntensities;

var
  FileName  : PathStr;
  Cancelled : Boolean;

begin
  GetFileName( FileName, '*.raw', 'Raw Intensities File', hcNoContext, Cancelled);
  if not Cancelled then begin
    Shock^.SaveIntensityPattern( FileName );
  end; {then}
end; {TRay.ShockSaveRawIntensities}

{------------------------------------***------------------------------------}

procedure TRay.ShockSaveRawDensities;

var
  FileName  : PathStr;
  Cancelled : Boolean;

begin
  GetFileName( FileName, '*.raw', 'Raw Densities File', hcNoContext, Cancelled);
  if not Cancelled then begin
    Shock^.SaveDensityProfile( FileName );
  end; {then}
end; {TRay.ShockSaveRawDensities}

{------------------------------------***------------------------------------}

procedure TRay.ShockDisplay;

begin
  LeaveTurboVision;
  Shock^.SetDisplay;
  Shock^.DisplayParameters( Crt.LightRed );
  Shock^.PlotDensityProfile( Crt.LightGreen );
  Shock^.PlotIntensityPattern( Crt.Yellow );
  repeat
    {null}
  until KeyPressed;
  Shock^.ClearDisplay;
  ResumeTurboVision;
end; {TRay.ShockDisplay}

{------------------------------------***------------------------------------}

procedure TRay.ShockDisplayExtrema;

begin
  LeaveTurboVision;
  Shock^.SetDisplay;
  Shock^.DisplayParameters( Crt.LightRed );
  Shock^.PlotDensityProfile( Crt.LightGreen );
  Shock^.PlotExtrema( Crt.Yellow );
  repeat
    {null}
  until KeyPressed;
  Shock^.ClearDisplay;
  ResumeTurboVision;
end; {TRay.ShockDisplayExtrema}

{------------------------------------***------------------------------------}


function TRay.ExtractIntensityDialog : PDialog;
```

```
var
  Dlg : PDialog;
  R   : TRect;
  Control, Labl, Histry : PView;

begin
  R.Assign( 22, 5, 58, 17 );
  New( Dlg, Init(R, 'Extract Image Intensities') );

  R.Assign( 4, 2, 19, 3 );
  Control := New( PStaticText, Init( R, 'Linearize Field' ));
  Dlg^.Insert( Control );

  R.Assign( 4, 3, 19, 6 );
  Control := New( PRadioButtons,
                  Init( R,
                        NewSItem('Bright',
                        NewSItem('Dark',
                        NewSItem('None',
                        Nil)))));
  PCluster( Control )^.Value := 0;
  Dlg^.Insert( Control );

  R.Assign( 21, 2, 32, 3 );
  Control := New(PStaticText, Init( R, 'Window Size' ));
  Dlg^.Insert( Control );

  R.Assign( 21, 3, 32, 6 );
  Control := New( PRadioButtons,
                  Init( R,
                        NewSItem('1',
                        NewSItem('2',
                        NewSItem('3',
                        Nil)))));
  PCluster( Control )^.Value := 0;
  Dlg^.Insert( Control );

  R.Assign( 4, 7, 32, 8 );
  Control := New( PCheckboxes, Init( R, NewSItem('Clear Previous Plots',Nil )));
  PCluster( Control )^.Value := 0;
  Dlg^.Insert( Control );

  R.Assign( 1, 8, 35, 9 );
  Control := New( PStaticText, Init( R, '──────────────────────────────────' ));
  Dlg^.Insert( Control );

  R.Assign( 4, 9, 10, 11 );
  Control := New( PButton, Init( R, 'Ok', cmOk, bfDefault ));
  Dlg^.Insert( Control );

  R.Assign( 12, 9, 22, 11 );
  Control := New( PButton, Init( R, 'Cancel', cmCancel, bfNormal ));
  Dlg^.Insert( Control );

  R.Assign( 24, 9, 32, 11 );
  Control := New( PButton, Init( R, 'Help', cmHelp, bfNormal ));
  Dlg^.Insert( Control );

  Dlg^.SelectNext( False );
  ExtractIntensityDialog := Dlg;
end; {TRay.ExtractIntensityDialog}

{-----------------------------------***-----------------------------------}

procedure TRay.ImageExtractIntensity;

var
  Control : Word;
  Field   : Images.Fields;
  Window  : Byte;
  Clear   : Boolean;
  Key     : Char;

begin
```

```
  if ImageFileLoaded then begin

    {-----------------------------------}
    {Execute the dialog                 }
    {-----------------------------------}

    Control := ExecuteDialog( ExtractIntensityDialog, ExtractIntensityData );

    {---------------------------}
    {Read values back from dialog}
    {---------------------------}

    if Control <> cmCancel then begin
      with ExtractIntensityData^ do begin
        case LinearizeField of
          0     : Field := Images.Bright;
          1     : Field := Images.Dark;
          2     : Field := Images.None;
          else    Field := Images.Bright;
        end; {case}
        case WindowSize of
          0     : Window := 1;
          1     : Window := 2;
          2     : Window := 3;
          else    Window := 1;
        end; {case}
        Clear := ( ClearPlot = 1 );
      end; {with}

      {----------------------------------}
      {Execute the appropiate image action}
      {----------------------------------}

      LeaveTurboVision;
      Image^.SetDisplay;

      repeat
        Image^.ShowImage( 100, 20, 1, 2 );
        Image^.MoveScanBar( Window );
        Image^.LineariseIntensity( Field );;
        Image^.ExtractIntensity;
        Image^.PlotIntensity( 15, 1, Clear );
        Key := ReadKey;
      until Ord( Key ) = 27;

      Image^.ClearDisplay;
      ResumeTurboVision;
    end {then}
    else begin
      {null}
    end; {else}

  end {then}
  else begin
    if Confirm( 'No Image File!'+#13+'Load one?' ) = cmYes then begin
      ImageLoadFile;
    end {then}
    else begin
      {null}
    end; {else}
  end; {else}

end; {TRay.ImageExtractIntensity}

{-----------------------------------***-----------------------------------}

procedure TRay.ImageDetermineScale;

begin
  if ImageFileLoaded then begin
    LeaveTurboVision;
    Image^.SetDisplay;
    Image^.ShowImage( 100, 20, 1, 2 );
    Image^.DetermineScale( Shocks.ElectrodeSeparation );
    Image^.ClearDisplay;
```

```
    ResumeTurboVision;
  end {then}
  else begin
   if Confirm( 'No Image File!'+#13+'Load one?' ) = cmYes then begin
     ImageLoadFile;
   end {then}
   else begin
     {null}
   end; {else}
  end; {else}
end; {TRay.DetermineScale}

{-----------------------------------***-----------------------------------}

procedure TRay.ImageLoadFile;

var
  FileName  : PathStr;
  Cancelled : Boolean;

begin
  GetFileName( FileName, '*.ima', 'Image File', hcNoContext, Cancelled);
  if not Cancelled then begin
    ImageFileLoaded := Image^.LoadImage( FileName );
    if not ImageFileLoaded then begin
      if Confirm( 'Invalid Image File!'+#13+'Load another?' ) = cmYes then begin
        ImageLoadFile;
      end {then}
      else begin
        {null}
      end; {else}
    end {then}
    else begin
      {null}
    end; {else}
  end {then}
  else begin
    ImageFileLoaded := False;
  end; {else}
end; {TRay.ImageLoadFile}

{-----------------------------------***-----------------------------------}

procedure TRay.ImageSaveRawIntensities;

var
  FileName  : PathStr;
  Cancelled : Boolean;

begin
  GetFileName( FileName, '*.raw', 'Image Raw Intensities File', hcNoContext, Cancelled);
  if not Cancelled then begin
    Image^.SaveRawIntensityData( FileName );
  end; {then}
end; {TRay.ImageSaveRawIntensities}

{-----------------------------------***-----------------------------------}
procedure TRay.ImageSaveIntensityPeaks;

var
  FileName  : PathStr;
  Cancelled : Boolean;

begin
  GetFileName( FileName, '*.pks', 'Image Intensity Peaks File', hcNoContext, Cancelled);
  if not Cancelled then begin
    Image^.SaveIntensityPeaks( FileName );
  end; {then}
end; {TRay.ImageSaveIntensityPeaks}

{-----------------------------------***-----------------------------------}

procedure TRay.ImageDisplay;
```

```
begin
  LeaveTurboVision;
  Image^.SetDisplay;
  if ImageFileLoaded then begin
    Image^.ShowImage( 100, 20, 1, 2 );
  end; {if}
  Image^.PlotIntensity( 15, 1, True );
  repeat
    {null}
  until KeyPressed;
  Image^.ClearDisplay;
  ResumeTurboVision;
end; {TRay.ImageDisplay}

{-----------------------------------***-----------------------------------}

procedure TRay.ShockRunMacroFile;

var
  FileName  : PathStr;
  Cancelled : Boolean;

begin
  GetFileName( FileName, '*.mac', 'Macro File', hcNoContext, Cancelled);
  if not Cancelled then begin
    AutoShock( FileName );
  end; {if}
end; {TRay.ShockRunMacroFile}

{-----------------------------------***-----------------------------------}

procedure TRay.AutoShock( Filename : PathStr );

const
  MaxMacroKeywords = 13;

  MacroKeyStrings : array [1..MaxMacroKeywords] of String =
                    ( 'new:',
                      'vacuum:',
                      'tail:',
                      'saddle:',
                      'rear:',
                      'front:',
                      'firerays:',
                      'interpolate:',
                      'display:',
                      'parameterfile:',
                      'densityfile:',
                      'intensityfile:',
                      'end:' );


type
  MacroKeywords = ( NewKeyword,
                    VacuumKeyword,
                    TailKeyword,
                    SaddleKeyword,
                    RearKeyword,
                    FrontKeyword,
                    FireRayKeyword,
                    InterpolateKeyword,
                    DisplayKeyword,
                    ParameterFileKeyword,
                    DensityFileKeyword,
                    IntensityFileKeyword,
                    EndKeyword,
                    InvalidKeyword );


type
  CheckSession = record
    Vacuum          : Boolean;
    Tail            : Boolean;
    Saddle          : Boolean;
    Rear            : Boolean;
```

```
    Front          : Boolean;
    FireRay        : Boolean;
    Interpolate    : Boolean;
    Display        : Boolean;
    ParameterFile  : Boolean;
    DensityFile    : Boolean;
    IntensityFile  : Boolean;
  end; {record}

type
  RaytraceSession = record
    RaytraceParameters  : Shocks.RaytraceParameterType;
    DensityParameters   : Shocks.DensityParameterType;
    Display             : Boolean;
    ParameterFileName   : PathStr;
    DensityFileName     : PathStr;
    IntensityFileName   : PathStr;
    Checks              : CheckSession;
  end;  {record}

label
  Abort;

var
  Macro               : Tokens.PTokenFile;
  Token0              : Tokens.Token;
  Token1              : Tokens.Token;
  Token2              : Tokens.Token;
  Session             : RaytraceSession;
  InterpolateString   : String;
  DisplayString       : String;
  FilenameString      : String;
  LineError           : String;
  Ok                  : Boolean;
  TheKeyword          : MacroKeywords;

  {----------------------------***----------------------------}

  procedure ResetSession;

  begin
    with Session do begin
      with RaytraceParameters do begin
        FireRaysFrom := 0;
        FireRaysTo   := 0;
        CurrentRay   := 0;
      end; {with}
      with DensityParameters do begin
        Vacuum.x    := 0;
        Vacuum.y    := 0.0;
        Tail.x      := 0;
        Tail.y      := 0.0;
        Saddle.x    := 0;
        Saddle.y    := 0.0;
        Rear.x      := 0;
        Rear.y      := 0.0;
        Front.x     := 0;
        Front.y     := 0.0;
        Interpolate := Linear;
      end; {with}
      Display           := False;
      ParameterFileName := '';
      DensityFileName   := '';
      IntensityFileName := '';
      with Checks do begin
        Vacuum        := False;
        Tail          := False;
        Saddle        := False;
        Rear          := False;
        Front         := False;
        FireRay       := False;
        Interpolate   := False;
        Display       := False;
        ParameterFile := False;
        DensityFile   := False;
```

```
        IntensityFile := False;
      end; {with}
    end; {with}
  end; {ResetSession}

  {---------------------------***----------------------------}
(*
  procedure Keyword( TheToken : Token; var TheKeyWord : MacroKeywords );

  var
    i : Word;

  begin
    if TheToken.Kind = tString then begin
      if TheToken.StringData =
      TheKeyword := NewKeyword;
      for i := 1 to MaxMacroKeywords do begin
        if TheToken.StringData = MacroKeyStrings[i] then begin
          Exit
        end {then}
        else begin
          TheKeyword := Succ( TheKeyword );
        end; {else}
      end; {for}
    end {then}
    else begin
      TheKeyword := InvalidKeyword;
    end; {else}
  end; {Keyword}

*)
  {---------------------------***----------------------------}

  procedure Keyword( TheToken : Token; var TheKeyWord : MacroKeywords );

  var
    i : Word;

  begin
    if TheToken.Kind = tString then begin
      TheKeyword := NewKeyword;
      for i := 1 to MaxMacroKeywords do begin
        if TheToken.StringData = MacroKeyStrings[i] then begin
          Exit
        end {then}
        else begin
          TheKeyword := Succ( TheKeyword );
        end; {else}
      end; {for}
    end {then}
    else begin
      TheKeyword := InvalidKeyword;
    end; {else}
  end; {Keyword}

  {---------------------------***----------------------------}

begin

  New( Macro, Init( FileName ));
  while not Macro^.EndOfFile do begin
    Ok := True;
    Macro^.ParseLine;
    Macro^.Next( Token0 );
    Macro^.Next( Token1 );
    Macro^.Next( Token2 );
    Keyword( Token0, TheKeyword );
    case TheKeyword of
      NewKeyword : begin
        ResetSession;
      end; {NewKeyword}
      VacuumKeyword : begin
        Ok := Macro^.AssignInteger( Session.DensityParameters.Vacuum.x, Token1 ) and
              Macro^.AssignReal( Session.DensityParameters.Vacuum.y, Token2 );
        Session.Checks.Vacuum := Ok;
```

```
end; {KeyWord}
TailKeyword : begin
  Ok := Macro^.AssignInteger( Session.DensityParameters.Tail.x, Token1 ) and
        Macro^.AssignReal( Session.DensityParameters.Tail.y, Token2 );
  Session.Checks.Tail := Ok;
end; {TailKeyWord}
SaddleKeyword : begin
  Ok := Macro^.AssignInteger( Session.DensityParameters.Saddle.x, Token1 ) and
        Macro^.AssignReal( Session.DensityParameters.Saddle.y, Token2 );
  Session.Checks.Saddle := Ok;
end; {SaddleKeyword}
RearKeyword : begin
  Ok := Macro^.AssignInteger( Session.DensityParameters.Rear.x, Token1 ) and
        Macro^.AssignReal( Session.DensityParameters.Rear.y, Token2 );
  Session.Checks.Rear := Ok;
end; {RearKeyWord}
FrontKeyword : begin
  Ok := Macro^.AssignInteger( Session.DensityParameters.Front.x, Token1 ) and
        Macro^.AssignReal( Session.DensityParameters.Front.y, Token2 );
  Session.Checks.Front := Ok;
end; {FrontKeyWord}
FireRayKeyword : begin
  Ok := Macro^.AssignInteger( Session.RayTraceParameters.FireRaysFrom, Token1 ) and
        Macro^.AssignInteger( Session.RayTraceParameters.FireRaysTo,   Token2 );
  Session.Checks.FireRay := Ok;
end; {FireRayKeyWord}
InterpolateKeyword : begin
  InterpolateString := 'linear';
  Ok := Macro^.AssignString( InterpolateString, Token1 );
  if InterpolateString = 'linear' then begin
    Session.DensityParameters.Interpolate := Linear;
  end {then}
  else begin
    if InterpolateString = 'lagrange' then begin
      Session.DensityParameters.Interpolate := Lagrange;
    end {then}
    else begin
      if InterpolateString = 'cubic' then begin
        Session.DensityParameters.Interpolate := Cubic;
      end {then}
      else begin
        if InterpolateString = 'junction' then begin
          Session.DensityParameters.Interpolate := Junction;
        end {then}
        else begin
          Session.DensityParameters.Interpolate := Linear;
        end; {else}
      end; {else}
    end; {else}
  end; {else}
  Session.Checks.Interpolate := Ok;
end; {InterpolateKeyWord}
DisplayKeyword : begin
  Ok := Macro^.AssignString( DisplayString, Token1 );
  Session.Display := ((DisplayString = 'yes') or (DisplayString = 'true'));
  Session.Checks.Display := Ok;
end; {DisplayKeyWord}
ParameterFileKeyword : begin
  Ok := Macro^.AssignString( FilenameString, Token1 );
  if Ok then Session.ParameterFilename := FilenameString;
  Session.Checks.ParameterFile := Ok;
end; {ParameterFileKeyWord}
DensityFileKeyword : begin
  Ok := Macro^.AssignString( FilenameString, Token1 );
  if Ok then Session.DensityFilename := FilenameString;
  Session.Checks.DensityFile := Ok;
end; {DensityFileKeyWord}
IntensityFileKeyword : begin
  Ok := Macro^.AssignString( FilenameString, Token1 );
  if Ok then Session.IntensityFilename := FilenameString;
  Session.Checks.IntensityFile := Ok;
end; {IntensityFileKeyWord}
EndKeyword : begin
   with Session do begin
     with Checks do begin
```

```
            if Vacuum and Tail and Saddle and Rear and Front and FireRay and
               Interpolate and ParameterFile then begin

              if Display then begin
                LeaveTurboVision;
                Shock^.SetDisplay;
                Shock^.SetDensityProfile( DensityParameters );
                Shock^.PlotDensityProfile( Crt.LightGreen );
                Shock^.DisplayParameters( Crt.LightRed );
                Shock^.SetRayTraceLimits( RaytraceParameters );
                Shock^.DisplayParameters( Crt.LightRed );
                Shock^.FireRays;
                Shock^.SaveParameters( ParameterFileName );
                if IntensityFile then begin
                  Shock^.SaveIntensityPattern( IntensityFileName );
                end {if};
                if DensityFile then begin
                  Shock^.SaveDensityProfile( DensityFileName );
                end; {if}
                Shock^.PlotIntensityPattern( Crt.Yellow );
                Delay( 10000 );
                Shock^.ClearDisplay;
                ResumeTurboVision;
              end {then}
              else begin
                Shock^.SetDensityProfile( DensityParameters );
                Shock^.SetRayTraceLimits( RaytraceParameters );
                Shock^.FireRays;
                Shock^.SaveParameters( ParameterFileName );
                if IntensityFile then begin
                  Shock^.SaveIntensityPattern( IntensityFileName );
                end; {if}
                if DensityFile then begin
                  Shock^.SaveDensityProfile( DensityFileName );
                end; {if}
              end; {else}

            end {then}
            else begin
              if (Confirm('Incomplete session' + #13 +
                          'Attempt to parse rest of file ?' ) = cmNo ) then
                goto Abort;

            end; {else}

          end; {with}
        end; {with}
        ResetSession;
      end; {EndKeyWord}
      InvalidKeyword : begin
        {null}
      end; {InvalidKeyWord}
      else
        {null}
    end; {case}

    if not Ok then begin
      Str( Macro^.LineAt, LineError );
      if (Confirm( 'Syntax Error in macro file : ' + FileName + #13 +
                   '              at line number: ' + LineError + #13 +
                   'Attempt to parse rest of file ?' ) = cmNo ) then
        goto Abort;

    end {then}
    else begin

    end; {then}

  end; {while}

  Abort:

  Dispose( Macro, Done );

end; {TRay.AutoShock}
```

```
{-----------------------------------***-----------------------------------}

begin
  Ray.Init;
  Ray.Run;
  Ray.Done;
end. {RayTrace}

{-----------------------------------***-----------------------------------}
```

E.2 Module shocks

```
{-----------------------------------***------------------------------------}
{ Unit        : Shocks.pas                                                   }
{ Compiler    : Borland Turbo Pascal v7.0                                    }
{ Function    : Trace laser beams through a spherical shock, defined by a   }
{               density profile, and produce an interferrence pattern.       }
{ Course      : Computer Science Masters - University of Natal (Durban)      }
{ Author      : Trevor Kistan                                                }
{ Notes       : The raytracing engine of this unit is a direct Fortran 77   }
{               to Turbo Pascal translation of Prof MM Michaelis' ray tracer.}
{ Purpose     : The ray tracer will generate the input-output pairs to be fed}
{               to the pattern associater neural network.  Density profiles }
{               need not correspond to physical reality as long as they      }
{               produce good enough interference patterns for the network to }
{               learn to behave as a ray tracer (one which can be run in     }
{               reverse!).                                                   }
{               Image processing will need to be done on the experimental    }
{               images to parameterise the interference pattern fed to net   }
{               once it has been trained on the ray tracer's data.           }
{ Last Update : 20/10/93                                                     }
{-----------------------------------***------------------------------------}

{$O+} {Overlays allowed}
{$F+} {Force far calls}

unit Shocks;

interface

uses
  Dos, Crt, Objects, Trig, Vga, FKeys, Graph, Curves, General,
  StdHdr, WorldDr, SEGraph;

const
  {-------------------------------------------}
  { Parameters of the experimental apparatus. }
  {-------------------------------------------}

  ScreenExtent           = 5000;   {microns} {should expand to 20000}
  ShockExtent            = 4000;   {microns}  {~3000}
  WavelengthExperiment   = 0.8;    {microns}
  WavelengthTheory       = 1.0;    {microns}
  ShockToScreenDistance  = 160000; {microns}
  ElectrodeSeparation    = 900;    {microns}
  MaxIntensityExtrema    = 100;     {program firewall}
  { LaserToScreenDistance = }

  {--------------------------------------}
  { Physical and mathematical constants. }
  {--------------------------------------}

  RefractiveIndexAir     = 3e-0004;
  RefractiveIndexOfAir   = 1.0002930000000000000;
  pi                     = 3.1415926535897932385;

  DefaultVacuumPosition  = 800;   {microns}
  DefaultTailPosition    = 1200;  {microns}
  DefaultSaddlePosition  = 2550;  {microns}
  DefaultRearPosition    = 2650;  {microns}
  DefaultFrontPosition   = 2700;  {microns}

  DefaultVacuumDensity   = 0.0;   {/atms}
  DefaultTaildensity     = 0.5;   {/atms}
  DefaultSaddleDensity   = 0.4;   {/atms}
  DefaultRearDensity     = 2.2;   {/atms}
  DefaultFrontDensity    = 1.0;   {/atms}


  {------------------------------------------}
  { Graph Windows.                           }
  {------------------------------------------}

  DensityGraph           = 1;
```

```
  IntensityGraph        = 2;
  ParameterData         = 3;
  Screen                = 4;

  {---------------------------------------}
  { Add some grayscales to the colour display }
  {---------------------------------------}

  Light  = 13;
  Panel  = 8;
  Shadow = 6;

type

  InterpolationMethods = ( Linear, Lagrange, Cubic, Junction );

  {-------------------------------------}
  { We need to work to double precision. }
  {-------------------------------------}

  Angle          = Double;
  Distance       = Double;    {continuous}
  Position       = LongInt;   {quantised into microns}
  Intensity      = Double;
  Density        = Double;
  Energy         = Double;

  DensityPoint = record
    x : Position;
    y : Density;
  end; {record}

  IntensityPoint = record
    x : Position;
    y : Intensity;
  end; {record}

  RaytraceParameterType = record
    FireRaysFrom : Position;
    FireRaysTo   : Position;
    CurrentRay   : Position;
  end; {record}

  DensityParameterType = record
    Vacuum      : DensityPoint;
    Tail        : DensityPoint;
    Saddle      : DensityPoint;
    Rear        : DensityPoint;
    Front       : DensityPoint;
    Interpolate : InterpolationMethods;
  end; {record}

  IntensityParameterType = array[1..MaxIntensityExtrema] of IntensityPoint;
  ScreenArray            = array[1..ScreenExtent] of Intensity;
  ShockArray             = array[1..ShockExtent] of Density;

  PShock = ^TShock;

  TShock = object( TObject )

    DensityParameters       : DensityParameterType;
    IntensityParameters     : IntensityParameterType;
    RaytraceLimits          : RaytraceParameterType;
    IntensityPattern        : ^ScreenArray;
    DensityProfile          : ^ShockArray;
    Extrema                 : Word;

    constructor Init;
    destructor Done; virtual;
    procedure SetDensityProfile( NewParameters : DensityParameterType ); virtual;
    procedure SetRaytraceLimits( NewParameters : RaytraceParameterType ); virtual;
    procedure FireRays; virtual;
    procedure SaveIntensityPattern( FileName : PathStr ); virtual;
    procedure SaveDensityProfile( FileName : PathStr ); virtual;
    procedure SaveParameters( FileName : PathStr ); virtual;
```

```
    procedure LoadParameters( FileName  : PathStr ); virtual;
    procedure PlotDensityProfile( Colour : Word ); virtual;
    procedure PlotIntensityPattern( Colour : Word ); virtual;
    procedure PlotExtrema( Colour : Word); virtual;
    procedure DisplayParameters( Colour : Word ); virtual;
    procedure SetDisplay; virtual;
    procedure ClearDisplay; virtual;

  private
    GraphicsMode          : Boolean;
    NetFile               : Text;
    DensFile              : Text;
    IntensFile            : Text;
    RefractiveIndex       : ^ShockArray;
    CosCompElectricField  : ^ScreenArray;
    SinCompElectricField  : ^ScreenArray;

    procedure CalcIntensity; virtual;
    procedure AvgIntensity; virtual;
    procedure LimitIntensity; virtual;
    procedure CheckProgress( Cycle : Position ); virtual;
    procedure DisplayProgress( Progress : Word ); virtual;
    procedure ExtractPeaks; virtual;
    procedure BlankParameters; virtual;

    procedure InterpolateIntensityParameters; virtual;
    procedure InterpolateDensityParameters; virtual;

  end; {object}

implementation

{------------------------------------***------------------------------------}

constructor TShock.Init;

var
  i : Word;

begin
  TObject.Init;

  Extrema := 0;
  GraphicsMode := False;

  with DensityParameters do begin
    Vacuum.x    := DefaultVacuumPosition;
    Vacuum.y    := DefaultVacuumDensity;
    Tail.x      := DefaultTailPosition;
    Tail.y      := DefaultTailDensity;
    Saddle.x    := DefaultSaddlePosition;
    Saddle.y    := DefaultSaddleDensity;
    Rear.x      := DefaultRearPosition;
    Rear.y      := DefaultRearDensity;
    Front.x     := DefaultFrontPosition;
    Front.y     := DefaultFrontDensity;
    Interpolate := Linear;
  end; {with}

  with RaytraceLimits do begin
    FireRaysFrom := DensityParameters.Vacuum.x;
    FireRaysTo   := DensityParameters.Front.x;
    CurrentRay   := 0;
  end; {with}

  for i := 1 to MaxIntensityExtrema do begin
    IntensityParameters[i].x := 0;
    IntensityParameters[i].y := 0.0;
  end; {for}

  {-------------------------------------}
  {Allocate storage for dynamic varibles.}
  {-------------------------------------}

  New( CosCompElectricField );
```

```
  New( SinCompElectricField );
  New( IntensityPattern );
  New( RefractiveIndex );
  New( DensityProfile );

  FillChar( IntensityPattern^, SizeOf( IntensityPattern^ ), 0 );
  FillChar( DensityProfile^, SizeOf( DensityProfile^ ), 1 );
  FillChar( RefractiveIndex^, SizeOf( RefractiveIndex^ ), 0);
  FillChar( CosCompElectricField^, SizeOf( CosCompElectricField^ ), 1 );
  FillChar( SinCompElectricField^, SizeOf( SinCompElectricField^ ), 0 );

  for i := 1 to ShockExtent do begin
    CosCompElectricField^[i] := 1.0;
    SinCompElectricField^[i] := 0.0;
  end; {for}

end; {TShock.Init}

{-----------------------------------***-----------------------------------}

destructor TShock.Done;

begin
  {-------------}
  {Recover memory}
  {-------------}
  Dispose( CosCompElectricField );
  Dispose( SinCompElectricField );
  Dispose( IntensityPattern );
  Dispose( RefractiveIndex );
  Dispose( DensityProfile );
  TObject.Done;
end; {TShock.Done}

{-----------------------------------***-----------------------------------}

procedure TShock.SetDensityProfile( NewParameters : DensityParameterType );

begin
  DensityParameters := NewParameters;
  InterpolateDensityParameters;
end; {TShock.SetDensityProfile}

{-----------------------------------***-----------------------------------}

procedure TShock.FireRays;

var
  i                 : Position;
  IE                : Position;
  IQ                : Position;
  R                 : Position;
  IIV               : Position;
  LIV               : Position;
  ii                : Position;

  IS                : Boolean;

  P                 : Angle;
  D                 : Angle;
  PHI               : Angle;

  U                 : Distance;
  OpticalPathLength : Distance;
  RN                : Distance;
  RM                : Distance;
  Z                 : Distance;
  Y                 : Distance;
  X                 : Distance;
  V                 : Distance;
  RL                : Distance;
  VL                : Distance;
  E                 : Distance;

begin
```

```
VL  := 5000.0;
PHI := 0.0;

{added - arbitary - TK}
liv := 1;

for i := RaytraceLimits.FireRaysFrom to RaytraceLimits.FireRaysTo do begin
  CheckProgress( i );
  D := 0.0;
  U := Sqrt( DensityParameters.Front.x*DensityParameters.Front.x - i*i );
  OpticalPathLength := DensityParameters.Front.x - U;
  IS := True;
  Y := I;
  IE := DensityParameters.Front.x - i;
  {---------------------------------------------------------}
  { These are the computational subshells.                  }
  {---------------------------------------------------------}
  for IQ := 1 to IE do begin
    if IS then begin
      R := (DensityParameters.Front.x + 1) - IQ;
      P := arcsin( Y / R );
      RN := 1.0 - ( R - ShockExtent ) / 1000.0 * RefractiveIndexAir;
      RM := 1.0 - ( R - ShockExtent ) / 1000.0 * RefractiveIndexAir;
      RN := RefractiveIndex^[ DensityParameters.Front.x + 1 - IQ ];
      RM := RefractiveIndex^[ DensityParameters.Front.x + 1 - IQ - 1 ];
      if P <= 1.414 then begin {Snells Law}
        D := arcsin(RN/RM * sin(D+P)) - P;
        Z := (tan(P+D) - tan(P)) * cos(P);
        Y := Y+Z;
        X := Sqrt( Sqr(R - 1.0) - Sqr(Y) );
        OpticalPathLength := OpticalPathLength + 2.0 * RM *
                             Sqrt( Sqr(X-U) + Sqr(Z) );
        U := X;
      end {then}
      else begin
        D := D + cos(P+D) * (RN - RM) * R;
        OpticalPathLength := OpticalPathLength + 2.0 * RM * R * cos(P+D);
        IS := False;
      end; {else}
    end; {if}
  end; {for}

  X  := -(Sqrt( DensityParameters.Front.x*DensityParameters.Front.x - i*i ));
  Y  := i;
  D  := -D;
  V  := Y*cos(D) + X*sin(D);

  U  := X*cos(D) - Y*sin(D);
  U  := -U;
  Y  := V*cos(D) - U*sin(D);

  X  := U*cos(D) + V*sin(D);
  D  := -D;
  D  := 2.0 * D;
  RL := ShockToScreenDistance;
  V  := (RL-X) * tan(D) + Y;
  OpticalPathLength := OpticalPathLength +
                       Sqrt( Sqr(V-Y) + Sqr(RL-X) );

  {--------------------------}
  { Keep within bounds.      }
  {--------------------------}
  if V < 0.0 then V := -V;
  if V > 5999.0 then V := 5800.0;

  Y := i;

  {careful here!}
  if V = VL then begin
    E := 1;
  end {then}
  else begin
    E := (1.0 / Abs(V-VL));  {check for /0}
  end; {else}
```

```
    OpticalPathLength := 2 * OpticalPathLength;

    OpticalPathLength := OpticalPathLength - Int( OpticalPathLength );
    VL := V;
    IIV := Trunc( V );

    if i = RaytraceLimits.FireRaysTo then LIV := IIV;

{    if i = DensityParameters.Vacuum.x-50 then LIV := IIV;}  {else ???}

    PHI := 2*pi * OpticalPathLength;
    CosCompElectricField^[IIV] := CosCompElectricField^[IIV] + E*cos(PHI);
    SinCompElectricField^[IIV] := SinCompElectricField^[IIV] + E*sin(PHI);

    if IIV = LIV then begin
      LIV := IIV;  {???} {null}{prevent /0 later on}
    end {then}
    else begin
      if IIV < LIV then begin
        for ii := IIV to LIV do begin
          CosCompElectricField^[II] := (CosCompElectricField^[IIV] -
                                        CosCompElectricField^[LIV]) *
                                        (II - LIV) / (IIV - LIV) +
                                        CosCompElectricField^[LIV];
          SinCompElectricField^[II] := (SinCompElectricField^[IIV] -
                                        SinCompElectricField^[LIV]) *
                                        (II - LIV) / (IIV - LIV) +
                                        SinCompElectricField^[LIV];

        end; {for}
      end {then}
      else begin
        for ii := LIV to IIV do begin
          CosCompElectricField^[II] := (CosCompElectricField^[IIV] -
                                        CosCompElectricField^[LIV] ) *
                                        (II - LIV) / (IIV - LIV) +
                                        CosCompElectricField^[LIV];
          SinCompElectricField^[II] := (SinCompElectricField^[IIV] -
                                        SinCompElectricField^[LIV] ) *
                                        (II - LIV) / (IIV - LIV) +
                                        SinCompElectricField^[LIV];

        end; {for}
      end; {else}
    end; {else}
    LIV := IIV;

  end; {for}

  CalcIntensity;
  AvgIntensity;
  LimitIntensity;

end; {TShock.FireRays}

{-----------------------------------***-----------------------------------}


{-----------------------------------***-----------------------------------}
{ The electric field intensity is proportional to the square of the       }
{ electric field.                                                         }
{-----------------------------------***-----------------------------------}

procedure TShock.CalcIntensity;

var
  {-----------------------------------}
  { Distance is quantised into microns. }
  {-----------------------------------}

  i : Word;

begin
  for i := 1 to ScreenExtent do begin
    IntensityPattern^[i] := Sqr( CosCompElectricField^[i] ) +
                            Sqr( SinCompElectricField^[i] );
  end; {for}
```

```
end; {TShock.CalcIntensity}

{-----------------------------------***-----------------------------------}


{-----------------------------------***-----------------------------------}
{ Average over 50 cells to smooth the intensity curve.                    }
{-----------------------------------***-----------------------------------}

procedure TShock.AvgIntensity;

const
  BinSize = 50;

var
  j, k, l       : LongInt;
  SLAV, AV, DAV : Double;

begin
  SLAV := IntensityPattern^[DensityParameters.Vacuum.x-50];
  j    :=  DensityParameters.Vacuum.x-50;
  repeat
    AV := IntensityPattern^[j];
    for K := 2 to BinSize do begin
      L := J+K-1;
      AV := AV + IntensityPattern^[L];
    end; {for}
    AV := AV / BinSize;
    DAV := (AV - SLAV) / BinSize;
    SLAV := AV;
    for K := 1 to BinSize do begin
      L := J+K-1;
      IntensityPattern^[L] := AV + (K-25) * DAV;
    end; {for}
    j := j + BinSize;
  until j >= 1300; {?????}{>added 27/1/95}
end; {TShock.AvgIntensity}

{-----------------------------------***-----------------------------------}


{-----------------------------------***-----------------------------------}
{                                                                         }
{-----------------------------------***-----------------------------------}

procedure TShock.LimitIntensity;

var
  i : Word;

begin
  for i := 1 to ShockExtent do begin
    if IntensityPattern^[i] < 0.0 then begin
      IntensityPattern^[i] := 0.0;
    end; {if}
    if IntensityPattern^[i] > 10.0 then begin
      IntensityPattern^[i] := 10.0;
    end; {if}
  end; {for}
end; {TShock.Intensity}

{-----------------------------------***-----------------------------------}

procedure TShock.SaveDensityProfile( FileName : PathStr );

var
  i : Word;
  d : Double;

begin
  Assign( DensFile, FileName );
  Rewrite( DensFile );
  for i := 1 to ShockExtent do begin
    d := DensityProfile^[i];
    Writeln( DensFile, d:30:30 );
```

```
  end; {for}
  Close( DensFile );
end; {TShock.SaveDensityProfile}

{-----------------------------------***-----------------------------------}

procedure TShock.SaveIntensityPattern( FileName : PathStr );

var
  i : Word;
  d : Double;

begin
  Assign( IntensFile, FileName );
  Rewrite( IntensFile );
  for i := 1 to ScreenExtent do begin
    d := IntensityPattern^[i];
    Writeln( IntensFile, d:30:30 );
  end; {for}
  Close( IntensFile );
end; {TShock.SaveIntensityPattern}

{-----------------------------------***-----------------------------------}

procedure TShock.PlotDensityProfile( Colour : Word );

const
  SolidLine  : Integer = 0;
  Resolution : Integer = 10; {20;}

var
  i          : Integer;
  x, y       : StdHdr.VeryLongVector;
  DataPoints : Integer;

begin

  DataPoints := ShockExtent div Resolution;
  for i := 1 to DataPoints do begin
    x[i-1] := i*Resolution;
    y[i-1] := DensityProfile^[i*Resolution];
  end; {for}
  SEGraph.SetCurrentWindow( DensityGraph );
  SEGraph.LinePlotData(x, y, DataPoints, Colour, SolidLine);

end; {TShock.PlotDensityProfile}

{-----------------------------------***-----------------------------------}

procedure TShock.PlotIntensityPattern( Colour : Word);

const
  SolidLine  : Integer = 0;
  Resolution : Integer = 10; {20;}

var
  i          : Word;
  x, y       : StdHdr.VeryLongVector;
  DataPoints : Integer;

begin

  DataPoints := ScreenExtent div Resolution;
  for i := 1 to DataPoints do begin
    x[i-1] := i*Resolution;
    y[i-1] := IntensityPattern^[i*Resolution];
  end; {for}
  SEGraph.SetCurrentWindow( IntensityGraph );
  SEGraph.LinePlotData( x, y, DataPoints, Colour, SolidLine );

end; {TShock.PlotIntensityPattern}

{-----------------------------------***-----------------------------------}

procedure TShock.PlotExtrema( Colour : Word);
```

```
const
  SolidLine  : Integer = 0;
  Resolution : Integer = 10; {20;}

var
  i          : Word;
  x, y       : StdHdr.VeryLongVector;
  DataPoints : Integer;

begin

  for i := 1 to Extrema do begin
    x[i] := IntensityParameters[i].x;
    y[i] := IntensityParameters[i].y;
  end; {for}
  SEGraph.SetCurrentWindow( IntensityGraph );
  SEGraph.LinePlotData( x, y, Extrema, Colour, SolidLine );

end; {TShock.PlotExtrema}

{-----------------------------------***------------------------------------}
{ Method:                                                                  }
{ =======                                                                  }
{ Some simplification of the intensity pattern is necessary before it can be }
{ fed to the pattern associator neural network.                            }
{ Each peak or trough of the intensity pattern is represented by two neurons,}
{ one records the x-position of the extrema, the other its intensity (y-pos).}
{ The intensity plot can be approximately reconstructed from these (x,y)   }
{ positions using a cubic-spline interpolation.                            }
{                                                                          }
{ Implementation:                                                          }
{ ===============                                                          }
{ We assume a plateau on either side of the intensity plot.                }
{                                                                          }
{                                                                          }
{-----------------------------------***------------------------------------}

procedure TShock.SaveParameters( Filename : PathStr );

var
  i : Word;

begin

  ExtractPeaks;
  Assign( NetFile, FileName );
  {$I-}
  Append( NetFile );
  {$I+}
  if IOResult <> 0 then begin
    Rewrite( NetFile );
  end; {then}

  Writeln( NetFile, 'RayTrace Neural Network Parameter File' );

  {Writeln( NetFile, '5 Input Neurons - Density Profile' );}
  with DensityParameters do begin
    Writeln( Netfile, Vacuum.x:10,  Vacuum.y:30:20 );
    Writeln( Netfile, Tail.x:10,    Tail.y:30:20   );
    Writeln( Netfile, Saddle.x:10,  Saddle.y:30:20 );
    Writeln( Netfile, Rear.x:10,    Rear.y:30:20   );
    Writeln( Netfile, Front.x:10,   Front.y:30:20  );
  end; {with}

  {Writeln( NetFile, Extrema, ' ', 'Output Neurons - Intensity Pattern' );}
  Writeln( NetFile, Extrema );
  for i := 1 to Extrema do begin
    Writeln( Netfile,
             IntensityParameters[i].x:30,
             IntensityParameters[i].y:30:20 );
  end; {for}

  Close( NetFile );

end; {TShock.SaveParameters}
```

```
{----------------------------------***-------------------------------------}

procedure TShock.LoadParameters( FileName : PathStr );

var
  i : Word;

begin

  Assign( NetFile, FileName );
  {$I-}
  Reset( NetFile );
  {$I+}
  if IOResult = 0 then begin
    {------}
    {Header}
    {------}
    Readln( NetFile );
    with DensityParameters do begin
      Readln( NetFile, Vacuum.x,  Vacuum.y  );
      Readln( NetFile, Tail.x,    Tail.y    );
      Readln( NetFile, Saddle.x,  Saddle.y  );
      Readln( NetFile, Rear.x,    Rear.y    );
      Readln( NetFile, Front.x,   Front.y   );
    end; {with}
    Readln( NetFile, Extrema );
    if Extrema > MaxIntensityExtrema then begin
      Extrema := MaxIntensityExtrema;
    end; {then}
    for i := 1 to Extrema do begin
      Readln( NetFile, IntensityParameters[i].x, IntensityParameters[i].y );
    end; {for}
    Close( NetFile );
    InterpolateDensityParameters;
    InterpolateIntensityParameters;
    LimitIntensity;
  end; {then}

end; {TShock.LoadParameters}

{----------------------------------***-------------------------------------}

procedure TShock.SetDisplay;

var
  IntensityGraphXAxis : SEGraph.StringVector;
  DensityGraphXAxis   : SEGraph.StringVector;

begin

  SEGraph.InitSEGraphics( '' );
  GraphicsMode := True;

{  SetTextStyle(smallfont, horizdir, 4);}

  Vga.SetColour( Light,  44, 44, 44 );
  Vga.SetColour( Panel,  40, 40, 40 );
  Vga.SetColour( Shadow, 36, 36, 36 );


  SEGraph.DefGraphWindow( 5, 20, 426, 196, DensityGraph );
  SEGraph.DefGraphWindow( 5, 219, 635, 475, IntensityGraph );
  SEGraph.DefGraphWindow( 434, 20, 635,196, ParameterData );
  SEGraph.DefGraphWindow( 0, 0, 639, 479, Screen );
  SEGraph.SetCurrentWindow( Screen );


  Vga.DrawSlide(0,0,430,200,15, Light, Panel, Shadow, Crt.LightGreen, 'Density Profile');
  Vga.DrawSlide(430,0,639,200,15, Light, Panel, Shadow, Crt.Blue, 'Parameters');
  Vga.DrawSlide(0,200,639,479,15, Light, Panel, Shadow, Crt.Yellow, 'Intensity Pattern');

  SEGraph.SetCurrentWindow( DensityGraph );
  SEGraph.SetViewBackGround( Panel );
  Graph.SetColor( Crt.White );
  SEGraph.SetPlotBackground( Crt.Black );
```

```
  SEGraph.SetCurrentWindow( IntensityGraph );
  SEGraph.SetViewBackGround( Panel );
  Graph.SetColor( Crt.White );
  SEGraph.SetPlotBackground( Crt.Black );

  SEGraph.SetCurrentWindow( ParameterData );
  SEGraph.SetViewBackGround( Panel );
  Graph.SetColor( Crt.White );

  SEGraph.SetCurrentWindow( DensityGraph );

{  SetTextStyle(smallfont, horizdir, 4);}

  SEGraph.ClearGraph;
  SEGraph.SetAxesType( 0, 0);
  SEGraph.ScalePlotArea( 0, 0, 4000, 3 );
  SEGraph.SetXYIntercepts( 0, 0 );
  SEGraph.DrawXAxis( 100, 0 );

  DensityGraphXAxis[0] := '1.0';
  DensityGraphXAxis[1] := '2.0';
  DensityGraphXAxis[2] := '3.0';
  DensityGraphXAxis[3] := '4.0';

  SEGraph.LabelXAxWithStrings( 10, DensityGraphXAxis, 4, 0 );
  SEGraph.DrawYAxis( 0.1, 0 );
  SEGraph.LabelYAxis( 10, 0 );
  Graph.SetColor( Crt.LightGreen );
  SEGraph.TitleXAxis( 'position (mm)' );
  SEGraph.TitleYAxis( 'density (/atms)' );
  Graph.SetColor( Crt.White );
  SetLineStyleXX( DottedLn, 0, NormWidth );
  SEGraph.DrawGridX( 10 );
  SEGraph.DrawGridY( 10 );
  SetLineStyleXX( SolidLn, 0, NormWidth );


  SEGraph.SetCurrentWindow( IntensityGraph  );

{  SetTextStyle(smallfont, horizdir, 4);}

  SEGraph.ClearGraph;
  SEGraph.SetAxesType( 0, 0 );
  SEGraph.ScalePlotArea( 0, 0, 5000, 10 );
  SEGraph.SetXYIntercepts( 0, 0 );
  SEGraph.DrawXAxis( 100, 0 );

  IntensityGraphXAxis[0] := '1.0';
  IntensityGraphXAxis[1] := '2.0';
  IntensityGraphXAxis[2] := '3.0';
  IntensityGraphXAxis[3] := '4.0';
  IntensityGraphXAxis[4] := '5.0';

  SEGraph.LabelXAxWithStrings( 10, IntensityGraphXAxis, 6, 0 );
  SEGraph.DrawYAxis( 2, 0 );
  SEGraph.LabelYAxis( 1, 0 );
  Graph.SetColor( Crt.Yellow );
  SEGraph.TitleXAxis( 'position (mm)' );
  SEGraph.TitleYAxis( 'intensity' );
  Graph.SetColor( Crt.White );
  SetLineStyleXX( DottedLn, 0, NormWidth );
  SEGraph.DrawGridX( 10 );
  SEGraph.DrawGridY( 1 );
  SetLineStyleXX( SolidLn, 0, NormWidth );

  SEGraph.SetCurrentWindow( ParameterData );

{  SetTextStyle(smallfont, horizdir, 4);}

  Graph.SetColor( Crt.Blue );
  SEGraph.LabelGraphWindow( 20, 800, '        Position Density', 0, 1 );

  Vga.DrawButton(2, 42, 56, 54,light,panel,shadow);
  Vga.DrawButton(2, 55, 56, 67,light,panel,shadow);
  Vga.DrawButton(2, 68, 56, 80,light,panel,shadow);
```

```
  Vga.DrawButton(2, 81, 56, 93,light,panel,shadow);
  Vga.DrawButton(2, 94, 56, 106,light,panel,shadow);

  Graph.SetColor( Crt.Blue );
  SEGraph.LabelGraphWindow( 20, 725, 'Vacuum', 0, 1 );
  SEGraph.LabelGraphWindow( 20, 650, 'Tail', 0, 1 );
  SEGraph.LabelGraphWindow( 20, 575, 'Saddle', 0, 1 );
  SEGraph.LabelGraphWindow( 20, 500, 'Rear', 0, 1 );
  SEGraph.LabelGraphWindow( 20, 425, 'Front', 0, 1 );
  SEGraph.LabelGraphWindow( 20, 300, '        From     To', 0, 1 );
  SEGraph.LabelGraphWindow( 20, 225, 'Rays', 0, 1 );
  SEGraph.LabelGraphWindow( 100, 120, '       Progress ', 0, 1 );

  BlankParameters;
  Graph.SetColor( Crt.White );

end; {TShock.SetDisplay}

{-----------------------------------***-----------------------------------}

procedure TShock.ClearDisplay;

begin
  SEGraph.CloseSEGraphics;
end; {TShock.ClearDisplay}

{-----------------------------------***-----------------------------------}

procedure TShock.BlankParameters;

var
  Colour : Word;

begin
  SEGraph.SetCurrentWindow( ParameterData );

{  SetTextStyle(smallfont, horizdir, 4);}

  Colour := Graph.GetColor;
  Graph.SetColor( Crt.Black );

  SEGraph.LabelGraphWindow( 100, 60, '[redacted]', 0, 1 );

  SEGraph.LabelGraphWindow( 328, 725, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 725, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 328, 717, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 717, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 328, 735, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 735, '[redacted]' , 0, 1);

  SEGraph.LabelGraphWindow( 328, 650, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 650, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 328, 642, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 642, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 328, 660, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 660, '[redacted]' , 0, 1);

  SEGraph.LabelGraphWindow( 328, 575, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 575, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 328, 567, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 567, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 328, 585, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 585, '[redacted]' , 0, 1);

  SEGraph.LabelGraphWindow( 328, 500, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 500, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 328, 492, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 492, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 328, 510, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 510, '[redacted]' , 0, 1);

  SEGraph.LabelGraphWindow( 328, 425, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 425, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 328, 417, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 417, '[redacted]' , 0, 1);
```

```
  SEGraph.LabelGraphWindow( 328, 435, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 435, '[redacted]' , 0, 1);

  SEGraph.LabelGraphWindow( 328, 225, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 225, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 328, 217, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 217, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 328, 235, '[redacted]' , 0, 1);
  SEGraph.LabelGraphWindow( 665, 235, '[redacted]' , 0, 1);

  Graph.SetColor( Colour );
end; {TShick.BlankParameters}

{-----------------------------------***-----------------------------------}

procedure TShock.DisplayParameters( Colour : Word );

const
  length   = 8;
  decimals = 6;

var
  VacuumPositionString : String;
  VacuumDensityString  : String;
  TailPositionString   : String;
  TailDensityString    : String;
  SaddlePositionString : String;
  SaddleDensityString  : String;
  RearPositionString   : String;
  RearDensityString    : String;
  FrontPositionString  : String;
  FrontDensityString   : String;
  FireFromString       : String;
  FireToString         : String;
  ColourState          : Word;

begin

  VacuumPositionString := General.WordToStr( DensityParameters.Vacuum.x );
  VacuumDensityString  := General.RealToStr( DensityParameters.Vacuum.y, length, decimals );
  TailPositionString   := General.WordToStr( DensityParameters.Tail.x );
  TailDensityString    := General.RealToStr( DensityParameters.Tail.y,   length, decimals );
  SaddlePositionString := General.WordToStr( DensityParameters.Saddle.x );
  SaddleDensityString  := General.RealToStr( DensityParameters.Saddle.y, length, decimals );
  RearPositionString   := General.WordToStr( DensityParameters.Rear.x );
  RearDensityString    := General.RealToStr( DensityParameters.Rear.y,   length, decimals );
  FrontPositionString  := General.WordToStr( DensityParameters.Front.x );
  FrontDensityString   := General.RealToStr( DensityParameters.Front.y,  length, decimals );
  FireFromString       := General.WordToStr( RaytraceLimits.FireRaysFrom );
  FireToString         := General.WordToStr( RaytraceLimits.FireRaysTo );

  BlankParameters;

  SEGraph.SetCurrentWindow( ParameterData );

{  SetTextStyle(smallfont, horizdir, 4);}

  ColourState := Graph.GetColor;
  Graph.SetColor( Colour );
  SEGraph.LabelGraphWindow( 333, 725, VacuumPositionString, 0, 1);
  SEGraph.LabelGraphWindow( 670, 725, VacuumDensityString,  0, 1);
  SEGraph.LabelGraphWindow( 333, 650, TailPositionString,   0, 1);
  SEGraph.LabelGraphWindow( 670, 650, TailDensityString,    0, 1);
  SEGraph.LabelGraphWindow( 333, 575, SaddlePositionString, 0, 1);
  SEGraph.LabelGraphWindow( 670, 575, SaddleDensityString,  0, 1);
  SEGraph.LabelGraphWindow( 333, 500, RearPositionString,   0, 1);
  SEGraph.LabelGraphWindow( 670, 500, RearDensityString,    0, 1);
  SEGraph.LabelGraphWindow( 333, 425, FrontPositionString,  0, 1);
  SEGraph.LabelGraphWindow( 670, 425, FrontDensityString,   0, 1);
  SEGraph.LabelGraphWindow( 333, 225, FireFromString,       0, 1);
  SEGraph.LabelGraphWindow( 670, 225, FireToString,         0, 1);
  Graph.SetColor( ColourState );

end; {TShock.DisplayParameters}
```

```
{-----------------------------------***-----------------------------------}

procedure TShock.InterpolateDensityParameters;

{-----------------------------------***-----------------------------------}
{ Interpolation the density parameters to yield a smooth density curve.    }
{ Densitys are expressed as a fraction of sea level.                        }
{ At the same time compute the refractive index of the shocked air.        }
{ Ignoring the complex chemical composition of air we note that, empirically }
{ the refactive index (n) of a gas depends on its density (d) as follows :  }
{                                                                           }
{    n = 1 + ( n0 - 1 ) * d/d0                                              }
{                                                                           }
{ n0 is the refractive index of air at sea level, and d0 the density at the }
{ same level - in our units d0 = 1.                                         }
{-----------------------------------***-----------------------------------}

var
  i, j          : Position;
  r, k          : Distance;
  p             : Density;
  X, Y          : Curves.List;
  M             : Curves.CoefficientMatrix;
  StartLinear   : Word;
  EndLinear     : Word;

begin

  StartLinear := 0;
  EndLinear   := 3;
  FillChar( DensityProfile^, SizeOf( DensityProfile^ ), 1 );

  with DensityParameters do begin
    X[0]  := Vacuum.x;
    Y[0]  := Vacuum.y;
    X[1]  := Tail.x;
    Y[1]  := Tail.y;
    X[2]  := Saddle.x;
    Y[2]  := Saddle.y;
    X[3]  := Rear.x;
    Y[3]  := Rear.y;
    X[4]  := Front.x;
    Y[4]  := Front.y;
  end; {with}

  {--------------------------------------------}
  {The density within the shock vacuum is zero  }
  {--------------------------------------------}

  for j := 1 to DensityParameters.Vacuum.x do begin
    DensityProfile^[j]   := 0.0;
    RefractiveIndex^[j]  := 1.0;
  end; {for}

  {--------------------------------------------}
  {The density ahead of the shock front is unity }
  {--------------------------------------------}

  for j := DensityParameters.Front.x to ShockExtent do begin
    DensityProfile^[j]   := 1.0;
    RefractiveIndex^[j]  := RefractiveIndexOfAir;
  end; {for}

  {-------------------------------------------------------------}
  {The density profile is curved from the shock vacuum to the shock rear}
  {-------------------------------------------------------------}

  with DensityParameters do begin
    case Interpolate of
      Lagrange : begin
        for j := Vacuum.x to Rear.x do begin
          p := Curves.Lagrange( 3, X, Y, j );
          DensityProfile^[j] := p;
          RefractiveIndex^[j] := 1.0 + (RefractiveIndexOfAir - 1.0)*p;
        end; {for}
```

```
        StartLinear := 3;
      end; {Lagrange}
      Cubic : begin
        Curves.CubicSpline( X, Y ,3, M );
        for i := 0 to 2 do begin
          for j := Trunc(X[i]) to Trunc(X[i+1]) do begin
            p := Curves.Cubic( j-X[i], M[0,i], M[1,i], M[2,i], M[3,i]);
            DensityProfile^[j] := p;
            RefractiveIndex^[j] := 1.0 + (RefractiveIndexOfAir - 1.0)*p;
          end; {for}
        end; {for}
        StartLinear := 3;
      end; {Cubic}
      Linear : begin
        StartLinear := 0;
      end; {Linear}
      else
        {null}
    end; {case}
  end; {with}

  {--------------------------------------------------------}
  {The shock front is always a straight line: p = r*j + k  }
  {--------------------------------------------------------}

  for i := StartLinear to EndLinear do begin
    r := (Y[i+1] - Y[i]) / (X[i+1] - X[i]);
    k := Y[i] - r*X[i];
    for j := Trunc( X[i] ) to Trunc( X[i+1] ) do begin
      p := r*j + k;
      DensityProfile^[j]  := p;
      RefractiveIndex^[j] := 1.0 + (RefractiveIndexOfAir - 1.0)*p;
    end; {for}
  end; {for}

end; {TShock.InterpolateDensityParameters}

{-----------------------------------***-----------------------------------}

procedure TShock.InterpolateIntensityParameters;

var
  X, Y : Curves.List;
  M    : Curves.CoefficientMatrix;
  i, j : Word;

begin
  FillChar( IntensityPattern^, SizeOf( IntensityPattern^ ), 0 );
  if Extrema > 0 then begin
    for i := 0 to Extrema-1 do begin
      X[i] := IntensityParameters[i+1].x;
      Y[i] := IntensityParameters[i+1].y;
    end; {for}
    Curves.CubicSpline( X, Y, Extrema-1, M );
    for i := 0 to Extrema-2 do begin
      for j := Trunc(X[i]) to Trunc(X[i+1]) do begin
        IntensityPattern^[j] := Curves.Cubic( j-X[i], M[0,i], M[1,i], M[2,i], M[3,i]);
      end; {for}
    end; {for}
  end; {then}
end; {TShock.InterpolateIntensityParameters}


(*
procedure TShock.InterpolateIntensityParameters;

var
  X, Y         : StdHdr.LongVector;
  Coefficients : StdHdr.RecMat;
  i            : Word;
  Yest         : Real;

begin
  if Extrema > 0 then begin
    for i := 0 to Extrema-1 do begin
```

```
      X[i] := IntensityParameters[i+1].x;
      Y[i] := IntensityParameters[i+1].y;
    end; {for}
    CurveFit.CubicSplines( X, Y, Extrema-1, Coefficients );
    for i := 0 to ScreenExtent-1 do begin
      CurveFit.CalcSpline( X, Coefficients, Extrema-1, i, Yest );
      IntensityPattern^[i+1] := Yest;
    end; {for}
  end; {then}
end; {TShock.InterpolateIntensityParameters}
*)

{-----------------------------------***-------------------------------------}

procedure TShock.SetRaytraceLimits( NewParameters : RaytraceParameterType );

var
  TempPos : Position;

begin

  {-------------------------------------}
  { Update the parameters               }
  {-------------------------------------}

  RaytraceLimits := NewParameters;

  {---------------------------------------------------}
  { Keep raytracing within the boundries of the shock }
  {---------------------------------------------------}

  if NewParameters.FireRaysFrom < 1 {DensityParameters.Vacuum.x} then begin
    RaytraceLimits.FireRaysFrom := 1;{DensityParameters.Vacuum.x;}
  end; {then}

  if NewParameters.FireRaysTo > ShockExtent {DensityParameters.Front.x} then begin
    RaytraceLimits.FireRaysTo := ShockExtent; {DensityParameters.Front.x;}
  end; {then}

  {--------------------------------------}
  { Check the ordering of the bounds     }
  {--------------------------------------}

  if RayTraceLimits.FireRaysTo < RayTraceLimits.FireRaysFrom then begin
    TempPos                     := RayTraceLimits.FireRaysTo;
    RayTraceLimits.FireRaysTo   := RayTraceLimits.FireRaysFrom;
    RayTraceLimits.FireRaysFrom := TempPos;
  end; {then}

end; {TShock.SetRaytraceLimits}

{-----------------------------------***-------------------------------------}

procedure TShock.CheckProgress( Cycle : Position );

var
  Progress : Position;

begin
  if GraphicsMode then begin
    with RaytraceLimits do begin
      Progress := Round((Cycle-FireRaysFrom) / (FireRaysTo-FireRaysFrom) * 20);
      if Progress > CurrentRay then begin
        CurrentRay := Progress;
        DisplayProgress( Progress );
      end {then}
      else begin
        {null}
      end; {else}
    end; {with}
  end; {if}

end; {TShock.CheckProgress}

{-----------------------------------***-------------------------------------}
```

```
procedure TShock.DisplayProgress( Progress : Word );

var
  ColourState : Word;

begin

  SEGraph.SetCurrentWindow( ParameterData );
  ColourState := Graph.GetColor;
  Graph.SetColor( Crt.LightRed );

  case Progress of
    1 : SEGraph.LabelGraphWindow( 100, 60, '█', 0, 1 );
    2 : SEGraph.LabelGraphWindow( 100, 60, '██', 0, 1 );
    3 : SEGraph.LabelGraphWindow( 100, 60, '███', 0, 1 );
    4 : SEGraph.LabelGraphWindow( 100, 60, '████', 0, 1 );
    5 : SEGraph.LabelGraphWindow( 100, 60, '█████', 0, 1 );
    6 : SEGraph.LabelGraphWindow( 100, 60, '██████', 0, 1 );
    7 : SEGraph.LabelGraphWindow( 100, 60, '███████', 0, 1 );
    8 : SEGraph.LabelGraphWindow( 100, 60, '████████', 0, 1 );
    9 : SEGraph.LabelGraphWindow( 100, 60, '█████████', 0, 1 );
   10 : SEGraph.LabelGraphWindow( 100, 60, '██████████', 0, 1 );
   11 : SEGraph.LabelGraphWindow( 100, 60, '███████████', 0, 1 );
   12 : SEGraph.LabelGraphWindow( 100, 60, '████████████', 0, 1 );
   13 : SEGraph.LabelGraphWindow( 100, 60, '█████████████', 0, 1 );
   14 : SEGraph.LabelGraphWindow( 100, 60, '██████████████', 0, 1 );
   15 : SEGraph.LabelGraphWindow( 100, 60, '███████████████', 0, 1 );
   16 : SEGraph.LabelGraphWindow( 100, 60, '████████████████', 0, 1 );
   17 : SEGraph.LabelGraphWindow( 100, 60, '█████████████████', 0, 1 );
   18 : SEGraph.LabelGraphWindow( 100, 60, '██████████████████', 0, 1 );
   19 : SEGraph.LabelGraphWindow( 100, 60, '███████████████████', 0, 1 );
   20 : SEGraph.LabelGraphWindow( 100, 60, '████████████████████', 0, 1 );
  end; {case}

  Graph.SetColor( ColourState );

end; {TShock.DisplayProgress}

{-----------------------------------***-----------------------------------}
(*
procedure TShock.ExtractPeaks;

var
  i : Position;

begin
  for i := 2 to ScreenExtent-1 do begin
    if (
         (
           (
             (IntensityPattern^[i] > IntensityPattern^[i-1]) and
             (IntensityPattern^[i] >  IntensityPattern^[i+1])
           )
         or
           (
             (IntensityPattern^[i] >  IntensityPattern^[i-1]) and
             (IntensityPattern^[i] > IntensityPattern^[i+1])
           )
         or
           (
             (IntensityPattern^[i] <  IntensityPattern^[i-1]) and
             (IntensityPattern^[i] < IntensityPattern^[i+1])
           )
         or
           (
             (IntensityPattern^[i] <  IntensityPattern^[i-1]) and
             (IntensityPattern^[i] <  IntensityPattern^[i+1])
           )
         ) and
         (Extrema <= MaxIntensityExtrema)
       ) then begin
      Inc( Extrema );
      IntensityParameters[Extrema].x := i;
      IntensityParameters[Extrema].y := IntensityPattern^[i];
    end; {if}
```

```
  end; {for}
end; {TShock.ExtractPeaks}
*)
{-----------------------------------***------------------------------------}

procedure TShock.ExtractPeaks;

var
  i         : Position;
  Threshold : Double;

begin
  Threshold := 0.0005;
  for i := 2 to ScreenExtent-1 do begin
    if ( ( ((IntensityPattern^[i] > (IntensityPattern^[i-1]+Threshold)) and
            (IntensityPattern^[i] > (IntensityPattern^[i+1]+Threshold)))
         or
           ((IntensityPattern^[i] < (IntensityPattern^[i-1]-Threshold)) and
            (IntensityPattern^[i] < (IntensityPattern^[i+1]-Threshold))))
         and
           (Extrema <= MaxIntensityExtrema)) then begin
      Inc( Extrema );
      IntensityParameters[Extrema].x := i;
      IntensityParameters[Extrema].y := IntensityPattern^[i];
    end; {if}
  end; {for}
end; {TShock.ExtractPeaks}

{-----------------------------------***------------------------------------}


end. {Shocks}

{-----------------------------------***------------------------------------}
```

E.3 Module images

```
{----------------------------------***-----------------------------------}
{ Unit        : Images.pas                                               }
{ Language    : Turbo Pascal v7.0                                        }
{ Function    : Pixel intensity image object                             }
{ Course      : MSc(Computer Science) - University of Natal (Durban Campus) }
{ Author      : Trevor Kistan                                            }
{ Last Update : 30/11/94                                                 }
{----------------------------------***-----------------------------------}

{$O+} {Overlays allowed}
{$F+} {Force far calls}

unit Images;

interface

uses
  FKeys, Graph, Dos, Crt, Vga, Objects, Memory, MsgBox, StdHdr, WorldDr,
  Smooth, SEGraph;

const
  LinesPerPage      = 64;
  PagesPerImage     = 8;
  PixelsPerLine     = 512;
  ImageSize         = 512;           {Square, Default}
  GrayScaleFactor   = 127.0 / 15.0; {MaxIntensity/MaxGrayScale}
  Screen            = 1;             {SEGraph window}
  IntensityGraph    = 2;             {SEGraph window}
  MaxPeaks          = 100;           {Firewall}
  MaxDataWindowSize = 3;

type
  Fields       = ( Bright, Dark, None );
  Printers     = ( DotMatrix, Laser );

  Intensity    = Byte;
  Position     = Word;
  ScreenCoOrds = Word;
  ImageCoOrds  = Word;

  IntensityPoint = record
    x : Position;
    y : Intensity;
  end; {record}

  PeakArray = array[1..MaxPeaks] of IntensityPoint;

  IntensityArray = array[1..PixelsPerLine] of Intensity;

  DataWindowType = array[1..MaxDataWindowSize] of IntensityArray;

  ImageAreaType = record
    x0      : ScreenCoOrds;
    y0      : ScreenCoOrds;
    x1      : ScreenCoOrds;
    y1      : ScreenCoOrds;
    xAspect : Byte;
    yAspect : Byte;
  end; {record}

  ScanBarType = record
    x0    : ImageCoOrds;
    x1    : ImageCoOrds;
    y     : ImageCoOrds;
  end; {record}

  PImage = ^Image;
  Image  = object
    constructor Init;
    destructor  Done; virtual;
    function    LoadImage( FileName : PathStr ) : Boolean; virtual;
    procedure SaveImage( FileName : PathStr ); virtual;
```

```
    procedure ShowImage( x, y : ScreenCoOrds; XShrinkFactor, YShrinkFactor : Byte ); virtual;
    procedure PrintImage( PrintDevice : Printers ); virtual;
    procedure SetDisplay; virtual;
    procedure ClearDisplay; virtual;
    procedure ExtractIntensity; virtual;
    procedure PlotIntensity( Colour : Integer; YAspect : Real; Clear : Boolean ); virtual;
    procedure LineariseIntensity( TheField : Fields ); virtual;
    procedure MoveScanBar( DataWindowSize : Byte ); virtual;
    procedure DetermineScale( Microns : Word ); virtual;
    procedure SaveIntensityPeaks( FileName : PathStr ); virtual;
    procedure SaveRawIntensityData( FileName : PathStr ); virtual;
    procedure LocateIntensityPeaks; virtual;
    procedure ExtractLine( x, y : Word; Field : Fields; Image, Data : PathStr ); virtual;
    procedure SeparateFields( ImaFile, BrightFile, DarkFile : PathStr ); virtual;
    procedure Smooth( RawFile, SmoothFile : PathStr; Window, Derivative : Integer ); virtual;
  private
    WindowSize          : Byte;
    DataWindow          : DataWindowType;
    TheIntensity        : IntensityArray;
    ImageArea           : ImageAreaType;
    ScanBar             : ScanBarType;
    Peaks               : PeakArray;
    NumPeaks            : Byte;
    ImageFile           : File;
    ImageFileName       : PathStr;
    IntensityRawFile    : Text;
    IntensityPeaksFile  : Text;
    MicronsPerPixel     : Double;
    function  GetPixel( i, j : Position ) : Intensity; virtual;
    procedure SetPixel( i, j : Position; Value : Intensity ); virtual;
    function LineariseDarkFieldPixel( pixel : Integer ): Integer; virtual;
    function LineariseBrightFieldPixel( pixel : Integer ): Integer; virtual;
  end {object};

{-----------------------------------***-----------------------------------}

implementation

{-----------------------------------***-----------------------------------}

procedure BoundsCheck( a0 : Word; var a : Word; a1 : Word );

begin
  if a < a0 then a := a0;
  if a > a1 then a := a1;
end; {BoundsCheck}

{-----------------------------------***-----------------------------------}

constructor Image.Init;

var
  i : Word;

begin

  FillChar( TheIntensity, SizeOf( TheIntensity ), 0 );

  with ImageArea do begin
    x0 := 0;
    y0 := 0;
    x1 := 0;
    y1 := 0;
    xAspect := 1;
    yAspect := 1;
  end; {with}

  with ScanBar do begin
    x0 := 0;
    x1 := 0;
    y  := 0;
  end; {with}

  NumPeaks := 0;
```

```
  for i := 1 to MaxPeaks do begin
    Peaks[i].x := 0;
    Peaks[i].y := 0;
  end; {for}

  ImageFileName := '';
  WindowSize := 1;
  MicronsPerPixel := 1.0;

end; {Image.Init}

{-------------------------------------***-------------------------------------}

destructor Image.Done;

begin
  {null}
end; {Image.Done}

{-------------------------------------***-------------------------------------}

function Image.GetPixel( i, j : Position ) : Intensity;

begin
  GetPixel := DataWindow[i, j];
end; {Image.Get}

{-------------------------------------***-------------------------------------}

procedure Image.SetPixel( i, j : Position; Value : Intensity );

begin
  DataWindow[i, j] := Value;
end; {Image.Set}

{-------------------------------------***-------------------------------------}

procedure Image.ShowImage( x, y : ScreenCoOrds; XShrinkFactor, YShrinkFactor : Byte );

{Read in the image a line at a time - we value memory over speed!}

var
  i, j, k      : Position;
  LineBuffer   : IntensityArray;
  DummyBuffer  : IntensityArray;
  BytesRead    : Word;

begin
  SEGraph.SetCurrentWindow( Screen );

  Assign( ImageFile, ImageFileName );
  Reset( ImageFile, 1 );

  for i := 1 to (ImageSize div YShrinkFactor) do begin
    {-----------------------------------------------------------}
    {start with the first line - bright field seems to start there}
    {-----------------------------------------------------------}
    BlockRead( ImageFile, LineBuffer, SizeOf( LineBuffer ), BytesRead );
    {-------------------------------}
    {now skip over the unwanted lines}
    {-------------------------------}
    for k := 1 to YShrinkFactor - 1 do begin
      BlockRead( ImageFile, DummyBuffer, SizeOf( DummyBuffer ), BytesRead );
    end; {for}
    for j := 1 to ImageSize div XShrinkFactor do begin
      Graph.PutPixel( x+j,
                      y+i,
                      Round( LineBuffer[j*XShrinkFactor]/GrayScaleFactor ) );
    end; {for}
  end; {for}

  Close( ImageFile );

  with ImageArea do begin
    x0 := x;
```

```
    y0 := y;
    x1 := x + ImageSize div XShrinkFactor;
    y1 := y + ImageSize div YShrinkFactor;
    xAspect := XShrinkFactor;
    yAspect := YShrinkFactor;
  end; {with}
end; {Image.ShowImage}

{---------------------------------***----------------------------------}

procedure Image.MoveScanBar( DataWindowSize : Byte );

const
  border     : Byte = 5;

var
  FK         : Boolean;
  ch         : Char;
  key        : FKeys.Keys;
  size       : Word;
  p          : Pointer;
  y, x0, x1  : Word;
  BytesRead  : Word;
  k          : Word;
  DummyLine  : IntensityArray;

begin
  WindowSize := DataWindowSize;
  SEGraph.SetCurrentWindow( Screen );
  size := Graph.ImageSize( ImageArea.x0+border,
                           ImageArea.y0+border,
                           ImageArea.x1-border,
                           ImageArea.y1-border );

  GetMem( p, size );
  Graph.GetImage( ImageArea.x0+border,
                  ImageArea.y0+border,
                  ImageArea.x1-border,
                  ImageArea.y1-border,
                  p^ );

  y    := (ImageArea.y1-ImageArea.y0) div 2;
  x0   := ImageArea.x0+border;
  x1   := ImageArea.x1-border;
  key  := FKeys.NullKey;

  Graph.Line( x0, y, x1, y );
  Graph.Line( x0, y-border, x0, y+border);
  Graph.Line( x1, y-border, x1, y+border);

  repeat
    FKeys.InKey(FK, ch, key);
    Graph.PutImage( ImageArea.x0+border, ImageArea.y0+border, p^, NormalPut );
    case key of
      FKeys.UpArrow     : begin dec(y) end;
      FKeys.DownArrow   : begin inc(y) end;
      FKeys.LeftArrow   : begin dec(x0); dec(x1) end;
      FKeys.RightArrow  : begin inc(x0); inc(x1) end;
      FKeys.PgUp        : begin dec(y, 10) end;
      FKeys.PgDown      : begin inc(y, 10) end;
      FKeys.HomeKey     : begin dec(x0, 10); dec(x1, 10) end;
      FKeys.EndKey      : begin inc(x0, 10); inc(x1, 10) end;
      FKeys.DeleteKey   : begin dec(x1) end;
      FKeys.InsertKey   : begin inc(x1) end;
    end; {case}
    BoundsCheck( ImageArea.x0+border, x0, ImageArea.x1-border );
    BoundsCheck( ImageArea.x0+border, x1, ImageArea.x1-border );
    BoundsCheck( ImageArea.y0+border,  y, ImageArea.y1-border );
    Graph.Line( x0, y, x1, y );
    Graph.Line( x0, y-border, x0, y+border);
    Graph.Line( x1, y-border, x1, y+border);
  until key = FKeys.CarriageReturn;

  FreeMem( p, size );
```

```
  ScanBar.x0 := x0*ImageArea.xAspect - ImageArea.x0;
  ScanBar.x1 := x1*ImageArea.xAspect - ImageArea.x0;
  ScanBar.y  :=  y*ImageArea.yAspect - ImageArea.y0;

  Assign( ImageFile, ImageFileName );
  Reset( ImageFile, 1 );
  for k := 1 to ScanBar.y - WindowSize - 1 do begin
    BlockRead( ImageFile, DummyLine, SizeOf( DummyLine), BytesRead );
  end; {for}
    {Seek( ImageFile, ScanBar.y - WindowSize );}
  BlockRead( ImageFile, DataWindow, SizeOf( DataWindow ), BytesRead );
(*
  for k := 1 to WindowSize*2+1 do begin
    BlockRead( ImageFile, DataWindow[k], SizeOf( DataWindow[k] ), BytesRead );
  end; {for}
*)
  Close( ImageFile );

end; {MoveScanWindow}


{-----------------------------------***-----------------------------------}

function Image.LoadImage( FileName : Dos.PathStr ) : Boolean;

var
  FileOK : Boolean;

begin
  ImageFileName := FileName;
  {$I-}
  Assign( ImageFile, ImageFileName );
  Reset( ImageFile, 1 );
  {$I+}
  FileOK := (IOResult = 0) and ( ImageFileName <> '' );
  if FileOK then begin
    FileOK := ( FileSize(ImageFile) = (ImageSize*ImageSize) );
  end; {if}
  LoadImage := FileOK;
end; {Image.LoadImage}

{-----------------------------------***-----------------------------------}

procedure Image.SaveImage( FileName : PathStr );

begin
end; {Image.SaveImage}

{-----------------------------------***-----------------------------------}

procedure Image.PrintImage( PrintDevice : Printers );

begin
end; {Image.PrintImage}

{-----------------------------------***-----------------------------------}

procedure Image.SetDisplay;

var
  IntensityGraphXAxis  : SEGraph.StringVector;
  ImageTitle           : String;
  Dir                  : DirStr;
  Name                 : NameStr;
  Ext                  : ExtStr;

begin

  SEGraph.InitSEGraphics( '' );
  Vga.SetGrayScale;
  SEGraph.DefGraphWindow( 0, 0, 639, 479, Screen );
  SEGraph.DefGraphWindow( 5, 300, 634, 474, IntensityGraph );
  SEGraph.SetCurrentWindow( Screen );

  Graph.SetColor( Crt.White );
```

```
  Dos.FSplit( ImageFileName, Dir, Name, Ext );
  ImageTitle := 'Interference Pattern Image: ' + Name + Ext;
  Vga.DrawSlide(0, 0, 90, 280, 15, 11, 10, 9, 15, 'GrayScale');
  Vga.DrawSlide(90, 0, 639, 280, 15, 11, 10, 9, 15, ImageTitle );
  Vga.DrawSlide(0, 280, 639, 479, 15, 11, 10, 9, 15, 'Linearised Intensity Plot' );
  Vga.DisplayGrayScale(5, 30, 15, true);

  SEGraph.SetCurrentWindow( IntensityGraph );
  SEGraph.SetViewBackGround( 10 );
  SEGraph.SetPlotBackground( Crt.Black );
  Graph.SetColor( Crt.White );
  SEGraph.SetCurrentWindow( IntensityGraph  );
  SEGraph.ClearGraph;
  SEGraph.SetAxesType( 0, 0 );
  SEGraph.ScalePlotArea( 1, 0, 512, 109 );
  SEGraph.SetXYIntercepts( 0, 0 );
  SEGraph.DrawXAxis( 10, 0 );

  IntensityGraphXAxis[0] := '1.0';
  IntensityGraphXAxis[1] := '2.0';
  IntensityGraphXAxis[2] := '3.0';
  IntensityGraphXAxis[3] := '4.0';
  IntensityGraphXAxis[4] := '5.0';

  SEGraph.LabelXAxWithStrings( 10, IntensityGraphXAxis, 6, 0 );
  SEGraph.DrawYAxis( 20, 0 );
  SEGraph.LabelYAxis( 1, 0 );
  Graph.SetColor( 15 );
  SEGraph.TitleXAxis( 'position (mm)' );
  SEGraph.TitleYAxis( 'intensity' );
  Graph.SetColor( Crt.White );
  SetLineStyleXX( DottedLn, 0, NormWidth );
  SEGraph.DrawGridX( 10 );
  SEGraph.DrawGridY( 1 );
  SetLineStyleXX( SolidLn, 0, NormWidth );

end; {Image.SetDisplay}

{-----------------------------------***------------------------------------}

procedure Image.ClearDisplay;

begin
  SEGraph.CloseSEGraphics;
end; {Image.ClearDisplay}

{-----------------------------------***------------------------------------}

procedure Image.ExtractIntensity;

var
  i, j : Integer;
  Sum  : IntensityArray;

begin
  FillChar( Sum, SizeOf( Sum ), 0);
  for i := 1 to WindowSize*2+1 do begin
    for j := ScanBar.x0 to ScanBar.x1 do begin
      Sum[j] := Sum[j] + GetPixel( i, j );
    end; {for}
  end; {for}
  {we need to reverse the order here!!!}
  for i := ScanBar.x0 to ScanBar.x1 do begin
    TheIntensity[i] := Round( Sum[i] / (WindowSize*2+1) );
  end; {for}
end; {Image.ExtractIntensity}

{-----------------------------------***------------------------------------}

procedure Image.PlotIntensity( Colour : Integer; YAspect : Real; Clear : Boolean );

const
  SolidLine  : Integer = 0;
  Resolution : Integer = 2;
```

```
var
  i          : Word;
  x, y       : StdHdr.VeryLongVector;
  DataPoints : Integer;

begin

  DataPoints := PixelsPerLine div Resolution;
  for i := 1 to DataPoints do begin
    x[i-1] := i*Resolution;
    y[i-1] := TheIntensity[i*Resolution]*YAspect;
  end; {for}

  SEGraph.SetCurrentWindow( IntensityGraph );
  if Clear then begin
    SEGraph.ClearGraph;
    Graph.SetColor( Crt.White );
    SetLineStyleXX( DottedLn, 0, NormWidth );
    SEGraph.DrawGridX( 10 );
    SEGraph.DrawGridY( 1 );
    SetLineStyleXX( SolidLn, 0, NormWidth );
  end {then}
  else begin
    {null}
  end; {else}
  SEGraph.LinePlotData( x, y, DataPoints, Colour, SolidLine );

end; {Image.PlotIntensity}

{-----------------------------------***-----------------------------------}

{-----------------------------------***-----------------------------------}
{ The CCTV camera used to capture the image has a logarithmic response for  }
{ intensity.  This procedure selects the brighter field and linearises the  }
{ intensity.  It was provided by members of the Laser Group in the Department}
{ of Physics, University of Natal, Durban and is based on empirical data.   }
{-----------------------------------***-----------------------------------}

procedure Image.LineariseIntensity( TheField : Fields );

var
  i, j, line              : Integer;
  LogPixel, LinearPixel   : Integer;
  Field0, Field1          : LongInt;
  Field                   : Byte;

begin

  Field0 := 0;
  Field1 := 0;

  {---------------------------------------------}
  {find the brighter field within the data window}
  {---------------------------------------------}

  for line := 1 to WindowSize*2+1 do begin
    for i := 0 to (PixelsPerLine-2) div 2 do begin
      Field0 := Field0 + GetPixel( line, 2*i );
      Field1 := Field1 + GetPixel( line, 2*i+1 );
    end; {for}
  end; {for}

  case TheField of
    Bright : begin
      if Field0 > Field1 then begin
        Field := 0;
      end {then}
      else begin
        Field := 1;
      end; {else}
      {-----------------------------------------}
      {linearise each pixel in the brighter field}
      {-----------------------------------------}
      for i := 1 to WindowSize*2+1 do begin
        for j := 0 to (PixelsPerLine-2) div 2 do begin
```

```
        LogPixel    := GetPixel( i, 2*j+Field );
        LinearPixel := LineariseBrightFieldPixel( LogPixel );
        SetPixel( i, 2*j+1, LinearPixel );
        SetPixel( i, 2*j,   LinearPixel );
      end; {for}
    end {for}
  end; {Bright}

  Dark : begin
    if Field1 > Field0 then begin
      Field := 0;
    end {then}
    else begin
      Field := 1;
    end; {else}
    {---------------------------------------}
    {linearise each pixel in the darker field}
    {---------------------------------------}
    for i := 1 to WindowSize*2+1 do begin
      for j := 0 to (PixelsPerLine-2) div 2 do begin
        LogPixel    := GetPixel( i, 2*j+Field );
        LinearPixel := LineariseDarkFieldPixel( LogPixel );
        SetPixel( i, 2*j+1, LinearPixel );
        SetPixel( i, 2*j,   LinearPixel );
      end; {for}
    end {for}
  end; {Dark}

  None : begin
    {null}
  end; {None}

  else begin
    {null}
  end; {else}

 end; {case}


end; {Image.LineariseIntensity}

{------------------------------------***------------------------------------}

procedure Image.SaveIntensityPeaks( FileName : PathStr );

var
  i : Word;

begin
  LocateIntensityPeaks;
  Assign( IntensityPeaksFile, FileName );
  Rewrite( IntensityPeaksFile );
  for i := 1 to NumPeaks do begin
    Writeln( IntensityPeaksFile, Peaks[i].x, '    ', Peaks[i].y );
  end; {for}
  Close( IntensityPeaksFile );
end; {Image.SaveIntensityPeaks}

{------------------------------------***------------------------------------}

procedure Image.SaveRawIntensityData( FileName : PathStr );

var
  i : Word;

begin
  Assign( IntensityRawFile, FileName );
  Rewrite( IntensityRawFile );
  for i := ScanBar.x0 to ScanBar.x1 do begin
    Writeln( IntensityRawFile, {i, '    ',} TheIntensity[i] );
  end; {for}
  Close( IntensityRawFile );
end; {Image.SaveRawIntensityData}

{------------------------------------***------------------------------------}
```

```
procedure Image.LocateIntensityPeaks;

var
  i : Word;

begin
  for i := ScanBar.x0+1 to ScanBar.x1-1 do begin
    if ( ((TheIntensity[i] > TheIntensity[i-1])   and
          (TheIntensity[i] >= TheIntensity[i+1])) or
         ((TheIntensity[i] >= TheIntensity[i-1])  and
          (TheIntensity[i] > TheIntensity[i+1])) ) then begin

      Inc( NumPeaks );
      if NumPeaks <= MaxPeaks then begin
        Peaks[NumPeaks].x := Round( (ScanBar.x1 - i)*MicronsPerPixel );
        Peaks[NumPeaks].y := TheIntensity[i];
      end {then}
      else begin
        {null}
      end; {else}
    end {then}
    else begin
      {null}
    end; {then}
  end; {for}
end; {Image.LocateIntensityPeaks}

{-----------------------------------***-----------------------------------}

procedure Image.DetermineScale( Microns : Word );

const
  border     : Byte = 5;

var
  FK         : Boolean;
  ch         : Char;
  key        : FKeys.Keys;
  size       : Word;
  p          : Pointer;
  y0, x, y1  : Word;

begin
  SEGraph.SetCurrentWindow( Screen );
  size := Graph.ImageSize( ImageArea.x0+border,
                           ImageArea.y0+border,
                           ImageArea.x1-border,
                           ImageArea.y1-border );

  GetMem( p, size );
  Graph.GetImage( ImageArea.x0+border,
                  ImageArea.y0+border,
                  ImageArea.x1-border,
                  ImageArea.y1-border,
                  p^ );

  y0    := (ImageArea.y1-ImageArea.y0) div 2;
  y1    := y0 + 30;
  x     := ImageArea.x1-100;
  key   := FKeys.NullKey;

  Graph.Line( x-border, y0, x+border, y0 );
  Graph.Line( x-border, y1, x+border, y1 );
  Graph.Line( x, y0, x, y1 );

  repeat
    FKeys.InKey(FK, ch, key);
    Graph.PutImage( ImageArea.x0+border, ImageArea.y0+border, p^, NormalPut );

    case key of
      FKeys.UpArrow    : begin dec(y0) end;
      FKeys.DownArrow  : begin inc(y1) end;
      FKeys.LeftArrow  : begin dec(x) end;
      FKeys.RightArrow : begin inc(x) end;
      FKeys.PgUp       : begin dec(y0, 10); dec(y1, 10) end;
```

```
      FKeys.PgDown     : begin inc(y0, 10); inc(y1, 10) end;
      FKeys.HomeKey    : begin dec(x, 10) end;
      FKeys.EndKey     : begin inc(x, 10) end;
      FKeys.DeleteKey  : begin dec(y1) end;
      FKeys.InsertKey  : begin inc(y1) end;
    end; {case}

    BoundsCheck( ImageArea.x0+border, x, ImageArea.x1-border );
    BoundsCheck( ImageArea.y0+border, y0, ImageArea.y1-border );
    BoundsCheck( ImageArea.y0+border, y1, ImageArea.y1-border );

    Graph.Line( x-border, y0, x+border, y0 );
    Graph.Line( x-border, y1, x+border, y1 );
    Graph.Line( x, y0, x, y1 );
  until key = FKeys.CarriageReturn;

  FreeMem( p, size );

  MicronsPerPixel := Microns / (ImageArea.yAspect*(y1-y0));

end; {Image.DetermineScale}

{-----------------------------------***-----------------------------------}

function Image.LineariseBrightFieldPixel( pixel : Integer ): Integer;

type
  Map = array[1..12] of Integer;

const
  pi  : Map = ( 0, 22, 26, 33, 44, 63, 83, 101, 113, 122, 125, 127 );
  lpi : Map = ( 0,  0,  7, 10, 14, 20, 28,  40,  56,  77,  94, 109 );

var
  k   : Integer;

begin
  for k := 1 to 11 do begin
    if ((pixel>pi[k]) and (pixel<=pi[k+1])) then begin
      LineariseBrightFieldPixel := Round( lpi[k]
                                          +(lpi[k+1]-lpi[k])
                                          *(pixel-pi[k])
                                          /(pi[k+1]-pi[k]) );
    end {if}
  end; {for}
end; {Image.LineariseBrightFieldPixel}

{-----------------------------------***-----------------------------------}

function Image.LineariseDarkFieldPixel( pixel : Integer ): Integer;

type
  Map = array[1..9] of Integer;

const
  pi  : Map = ( 0, 22, 23, 27, 38, 48, 66, 103, 127 );
  lpi : Map = ( 0,  0,  7, 14, 28, 40, 56, 109, 127 );

var
  k : Integer;

begin
  for k:=1 to 11 do begin
    if ((pixel>pi[k]) and (pixel<=pi[k+1])) then begin
      LineariseDarkFieldPixel := Round( lpi[k]
                                        +(lpi[k+1]-lpi[k])
                                        *(pixel-pi[k])
                                        /(pi[k+1]-pi[k]));
    end; {if}
  end; {for}
end; {Image.LineariseDarkFieldPixel}

{-----------------------------------***-----------------------------------}

procedure Image.ExtractLine( x, y : Word; Field : Fields; Image, Data : PathStr );
```

```
var
  Raw           : File;
  Profile       : Text;
  Line, Intens  : IntensityArray;
  i, BytesRead  : Word;

begin
  Assign( Raw, Image );
  Reset( Raw, 1 );
  for i := 1 to y do begin
    BlockRead( Raw, Line, SizeOf( Line ), BytesRead );
  end; {for}
  Close( Raw );

  case Field of
    Bright : begin
        for i := 1 to SizeOf( Line ) do begin
          Line[i] := LineariseBrightFieldPixel( Line[i] );
        end; {for}
    end; {Bright}
    Dark : begin
        for i := 1 to SizeOf( Line ) do begin
          Line[i] := LineariseDarkFieldPixel( Line[i] );
        end; {for}
    end; {Dark}
    None : begin
      {null}
    end; {None}
  end; {case}

  for i := x downto 1 do begin
    Intens[x-i+1] := Line[i];
  end; {for}

  Assign( Profile, Data );
  Rewrite( Profile );
  for i := 1 to x do begin
    Writeln( Profile, Intens[i] );
  end; {for}
  Close( Profile );
end; {Image.ExtractLine}

{-----------------------------------***-------------------------------------}

procedure Image.SeparateFields( ImaFile, BrightFile, DarkFile : PathStr );

var
  Image, Bright, Dark         : File;
  Count1, Count2              : LongInt;
  Line                        : IntensityArray;
  i, BytesRead, BytesWritten  : Word;

begin
  Assign( Image, ImaFile );
  Assign( Bright, BrightFile );
  Assign( Dark, DarkFile );
  Reset( Image, 1 );
  Rewrite( Bright, 1 );
  Rewrite( Dark, 1 );

  Count1 := 0;
  Count2 := 0;
  BlockRead( Image, Line, SizeOf( Line ), BytesRead );
  for i := 1 to BytesRead div 4 do begin
    Count1 := Count1 + Line[i];
  end; {for}
  BlockRead( Image, Line, SizeOf( Line ), BytesRead );
  for i := 1 to BytesRead div 4 do begin
    Count2 := Count2 + Line[i];
  end; {for}

  Close( Image );
  Reset( Image, 1 );
  if Count1 > Count2 then begin
    for i := 1 to ImageSize div 2 do begin
```

```
      BlockRead( Image, Line, SizeOf( Line ), BytesRead );
      BlockWrite( Bright, Line, BytesRead, BytesWritten );
      BlockRead( Image, Line, SizeOf( Line ), BytesRead );
      BlockWrite( Dark, Line, BytesRead, BytesWritten );
    end; {for}
  end {then}
  else begin
    for i := 1 to ImageSize div 2 do begin
      BlockRead( Image, Line, SizeOf( Line ), BytesRead );
      BlockWrite( Dark, Line, BytesRead, BytesWritten );
      BlockRead( Image, Line, SizeOf( Line ), BytesRead );
      BlockWrite( Bright, Line, BytesRead, BytesWritten );
    end; {for}
  end; {else}

  Close( Image );
  Close( Bright );
  Close( Dark );
end; {Image.SeparateFields}

{-----------------------------------***-----------------------------------}

procedure Image.Smooth( RawFile, SmoothFile : PathStr; Window, Derivative : Integer );

var
  Raw, Red          : VeryLongVector;
  i, Points         : Integer;
  RawData, RedData  : Text;

begin
  Assign( RawData, RawFile );
  Assign( RedData, SmoothFile );
  Reset( RawData );
  Rewrite( RedData );

  Points := 0;
  while not Eof( RawData ) do begin
    Readln( RawData, Raw[Points] );
    Inc( Points );
  end; {while}

  DataSmoothSg( Raw, Points, Window, Derivative, Red );

  for i := 0 to Points-1 do begin
    Writeln( RedData, Red[i] );
  end; {for}

  Close( RawData );
  Close( RedData );
end; {Image.Smooth}

{-----------------------------------***-----------------------------------}

end. {Images}

{-----------------------------------***-----------------------------------}
```

E.4 Module tokens

```
{-----------------------------------***-----------------------------------}
{ Unit        : Tokens.pas                                                 }
{ Language    : Turbo Pascal v7.0                                          }
{ Function    : Object orientated text file token routines                 }
{ Course      : MSc(Computer Science) - University of Natal (Durban Campus) }
{ Author      : Trevor Kistan                                              }
{ Last Update : 14-01-1995                                                 }
{-----------------------------------***-----------------------------------}

unit Tokens;

interface

uses
  Dos, Crt, Objects, Queues;

const
  OpenCommentChr = '!';
  Blank          = ' ';

type


  ValidTokens = (tInteger, tReal, tString, tNull);

  PToken = ^Token;
  Token  = record
    case Kind : ValidTokens of
      tInteger : (IntegerData : LongInt);
      tReal    : (RealData    : Double);
      tString  : (StringData  : String);
    end; {case}
  {end; record}

  PTokenFile = ^TTokenFile;
  TTokenFile = object
    constructor Init( FileName : PathStr );
    procedure   Next( var NextToken : Token ); virtual;
    procedure   ParseLine;
    function    MoreTokens : Boolean; virtual;
    function    EndOfFile : Boolean; virtual;
    function    LineAt : Word; virtual;
    function    Same( Candidate, Target : String ) : Boolean; virtual;
    function    AssignInteger( var TheInteger : LongInt; TheToken : Token ) : Boolean; virtual;
    function    AssignReal( var TheReal : Double; TheToken : Token ) : Boolean; virtual;
    function    AssignString( var TheString : String; TheToken : Token ) : Boolean; virtual;
    function    Keyword( TheString : String; TheToken : Token ) : Boolean; virtual;
    destructor  Done; virtual;
  private
    TokenFile     : Text;
    TokenLine     : String;
    TokenList     : Queues.PQueue;
    TokenFilename : String;
    FileExists    : Boolean;
    LineNo        : Word;
    procedure TrimLine;  virtual;
    function  Pad( S : String; Len : Byte ) : String; virtual;
    function  CommentLine : Boolean; virtual;
    function  EmptyLine : Boolean; virtual;
    function  IsReal( RealString : String; var RealValue : Double ) : Boolean; virtual;
    function  IsInteger( IntegerString : String; var IntegerValue : LongInt ) : Boolean; virtual;
  end; {object}

implementation

const
  NullToken : Token = (Kind : tNull);

{-----------------------------------***-----------------------------------}

constructor TTokenFile.Init( Filename : PathStr );
```

```
{initialise file object}

begin
  New( TokenList, Init );
  {$I-}
  Assign( TokenFile, Filename );
  Reset( TokenFile );
  {$I+}
  FileExists := (IOResult = 0);
  LineNo     := 0;
end; {TTokenFile.Init}


{-----------------------------------***-----------------------------------}

destructor TTokenFile.Done;

begin
  {$I-}
  Close( TokenFile );
  {$I+}
  Dispose( TokenList, Done );
end; {TTokenFile.Done}

{-----------------------------------***-----------------------------------}

function TTokenFile.CommentLine : Boolean;

{--------------------------------------}
{determines if the line is a comment line}
{--------------------------------------}

var
  i : Word;

begin
  {-------------------------------}
  {get the first non-blank character}
  {-------------------------------}
  i := 0;
  repeat
    Inc(i);
  until ((TokenLine[i] <> Blank) or (i = Length(TokenLine)));
  {----------------------------------}
  {check if it is the comment indicator}
  {----------------------------------}
  CommentLine := (TokenLine[i] = OpenCommentChr);
end; {TTokenFile.CommentLine}

{-----------------------------------***-----------------------------------}

function TTokenFile.EmptyLine : Boolean;

{-----------------------------}
{determines if the line is blank}
{-----------------------------}

var
  i     : Word;
  Empty : Boolean;

begin
  i     := 0;
  Empty := True;
  repeat
    Inc(i);
    if TokenLine[i] <> Blank then begin
      Empty := False;
    end; {if}
  until ((i = Length( TokenLine )) or (not Empty));
  EmptyLine := Empty;
end; {TTokenFile.EmptyLine}

{-----------------------------------***-----------------------------------}
```

```
procedure TTokenFile.TrimLine;

begin
  {---------------------}
  {remove leading blanks}
  {---------------------}
  while (Length(TokenLine) > 0) and (TokenLine[1] = Blank) do begin
    Delete( TokenLine, 1, 1 );
  end; {while}

  {----------------------}
  {remove trailing blanks}
  {----------------------}
  while (Length(TokenLine) > 0) and (TokenLine[Length(TokenLine)] = Blank) do begin
    Dec(TokenLine[0]);
  end; {while}
end; {TTokenFile.TrimLine}

{-----------------------------------***-----------------------------------}

procedure TTokenFile.ParseLine;

{-----------------------------------------------------------------}
{parses a non-comment, non-blank line into a linked list of Tokens}
{-----------------------------------------------------------------}

var
  Buffer       : String;
  i, BufferLen : Word;
  IntegerNum   : LongInt;
  Realnum      : Double;
  NewToken     : Ptoken;

begin

  if FileExists then begin

    {------------------}
    {Clear the TokenList}
    {------------------}

    TokenList^.Clear;

    {----------------}
    {get a valid line}
    {----------------}

    if not Eof( TokenFile ) then begin
      repeat
        Readln( TokenFile, TokenLine );
        Inc( LineNo );
      until ( ((not CommentLine) and (not EmptyLine)) or Eof(TokenFile) );
    end; {then}


    {----------------------------------------------------------}
    {ensure that only single blanks pad the line on either end}
    {----------------------------------------------------------}

    TrimLine;
    TokenLine := Blank+TokenLine+Blank;

    i := 1;
    repeat
      BufferLen := 0;
      {-------------------}
      {skip leading blanks}
      {-------------------}
      while ((TokenLine[i] = Blank) and (i < Length(TokenLine))) do begin
        Inc( i );
      end; {while}
      {----------}
      {copy token}
      {----------}
      while ((TokenLine[i] <> Blank) and (i < Length(TokenLine))) do begin
```

```
        Inc( BufferLen );
        Buffer[BufferLen] := TokenLine[i];
        Inc( i );
      end; {while}
      Buffer[0] := Chr(BufferLen);

      New( NewToken );

      if IsInteger( Buffer, IntegerNum ) then begin
        NewToken^.Kind        := tInteger;
        NewToken^.IntegerData := IntegerNum;
      end {then}
      else begin
        if IsReal( Buffer, RealNum ) then begin
          NewToken^.Kind     := tReal;
          NewToken^.RealData := RealNum;
        end {then}
        else begin
          NewToken^.Kind       := tString;
          NewToken^.StringData := Buffer;
        end {else}
      end; {else}

      TokenList^.Add( NewToken );
    until ((i = Length(TokenLine)) or (TokenLine[i] = OpenCommentChr));

  end; {if}

end; {TTokenFile.ParseLine}

{-----------------------------------***------------------------------------}

function TTokenFile.Same(Candidate, Target: String) : Boolean;

{case insensitive substring check - n.b. blanks not removed}

var
  i : Word;

begin
  for i := 1 to Length(Candidate) do begin
    Candidate[i] := UpCase(Candidate[i]);
  end; {for}
  for i := 1 to Length(Target) do begin
    Target[i] := UpCase(Target[i]);
  end; {for}
  Same := (Candidate = Target)
end; {TTokenFile.Same}

{-----------------------------------***------------------------------------}

procedure TTokenFile.Next( var NextToken : Token );

{returns the next Token in the queue}

var
  PTempToken : PToken;

begin
  if TokenList^.Empty then begin
    NextToken := NullToken;
  end {then}
  else begin
    PTempToken := PToken( TokenList^.Next );
    NextToken  := Token( PTempToken^ );
    Dispose( PTempToken );
  end; {else}
end; {TTokenFile.Next}

{-----------------------------------***------------------------------------}

function TTokenFile.IsReal( RealString : String; var RealValue : Double ) : Boolean;

var
  code : Integer;
```

```
begin
  Val( RealString, RealValue, code );
  IsReal := (code = 0);
end; {TTokenFile.IsReal}

{-----------------------------------***------------------------------------}

function TTokenFile.IsInteger( IntegerString : String; var IntegerValue : LongInt ) : Boolean;

var
  code : Integer;

begin
  Val( IntegerString, IntegerValue, code );
  IsInteger := (code = 0);
end; {TTokenFile.IsInteger}

{-----------------------------------***------------------------------------}

function TTokenFile.EndOfFile : Boolean;

begin
  EndOfFile := Eof( TokenFile );
end; {TTokenFile.EndOfFile}

{-----------------------------------***------------------------------------}

function TTokenFile.Pad(S : String; Len : Byte) : String;

{right-pad string to length len with blanks - truncate if > Len}

var
  o : String;

begin
  if Length(S) >= Len then begin
    {-------------------}
    {overflow - truncate}
    {-------------------}
    o    := S;
    o[0] := Chr(Len);
    Pad  := o;
  end {then}
  else begin
    {---------}
    {underflow}
    {---------}
    o[0] := Chr(Len);
    Move(S[1], o[1], Length(S));
    FillChar(o[Succ(Length(S))], Len-Length(S), Blank);
    Pad := o;
  end; {else}
end; {TTokenFile.Pad}

{-----------------------------------***------------------------------------}

function TTokenFile.LineAt : Word;
{returns the current line number - used in error messages}

begin
  LineAt := LineNo;
end; {TTokenFile.LineAt}

{-----------------------------------***------------------------------------}

function TTokenFile.MoreTokens : Boolean;

begin
  MoreTokens := not( TokenList^.Empty );
end; {TTokenFile.MoreTokens}

{-----------------------------------***------------------------------------}

function TTokenFile.AssignInteger( var TheInteger : LongInt; TheToken : Token ) : Boolean;
```

```
begin
  if TheToken.Kind = tInteger then begin
    TheInteger    := TheToken.IntegerData;
    AssignInteger := True;
  end {then}
  else begin
    AssignInteger := False;
  end; {else}
end; {TTokenFile.AssignInteger}

{-----------------------------------***-----------------------------------}

function TTokenFile.AssignReal( var TheReal : Double; TheToken : Token ) : Boolean;

begin
  if TheToken.Kind = tReal then begin
    TheReal    := TheToken.RealData;
    AssignReal := True;
  end {then}
  else begin
    AssignReal := False;
  end; {else}
end; {TTokenFile.AssignReal}

{-----------------------------------***-----------------------------------}

function TTokenFile.AssignString( var TheString : String; TheToken : Token ) : Boolean;

begin
  if TheToken.Kind = tString then begin
    TheString    := TheToken.StringData;
    AssignString := True;
  end {then}
  else begin
    AssignString := False;
  end; {else}
end; {TTokenFile.AssignString}


{-----------------------------------***-----------------------------------}

function TTokenFile.Keyword( TheString : String; TheToken : Token ) : Boolean;


begin
  if TheToken.Kind = tString then begin
    Keyword := (TheToken.StringData = TheString);
  end {then}
  else begin
    Keyword := False;
  end; {else}
end; {TTokenFile.Keyword}


{-----------------------------------***-----------------------------------}

  {no initialisation code}

end. {Tokens}

{-----------------------------------***-----------------------------------}
```

E.5 Module queues

```
{-----------------------------------***-----------------------------------}
{ Unit        : Queues.pas                                                  }
{ Language    : Turbo Pascal v7.0                                           }
{ Function    : Generic queue object                                        }
{ Course      : MSc(Computer Science) - University of Natal (Durban Campus) }
{ Author      : Trevor Kistan                                               }
{ Last Update : 28-01-1995                                                  }
{-----------------------------------***-----------------------------------}

unit Queues;

interface

uses
  Dos, Crt, Objects;

type

  QueuePointer = ^QueueNode;
  QueueNode = record
    Data : Pointer;
    Link : QueuePointer;
  end; {record}

  PQueue = ^Queue;
  Queue  = object
    constructor Init;
    procedure   Add( NewElement : Pointer ); virtual;
    function    Next : Pointer; virtual;
    function    Empty : Boolean; virtual;
    function    Size : Word; virtual;
    procedure   Clear; virtual;
    destructor  Done; virtual;
  private
    Front     : QueuePointer;
    Rear      : QueuePointer;
    QueueSize : Word;
  end; {object}

implementation

{-----------------------------------***-----------------------------------}

constructor Queue.Init;

begin
  Front     := nil;
  Rear      := nil;
  QueueSize := 0;
end; {Queue.Init}

{-----------------------------------***-----------------------------------}
{ Calling procedures must allocate memory for NewElement                    }

procedure Queue.Add( NewElement : Pointer );

var
  Temp : QueuePointer;

begin
  New( Temp );
  Temp^.Data := NewElement;
  Rear^.Link := Temp;
  Temp^.Link := nil;
  Rear := Temp;
  if Empty then begin
    Front := Rear;
  end; {if}
  Inc( QueueSize );
end; {Queue.Add}

{-----------------------------------***-----------------------------------}
```

```
{ Calling procedures must deallocate memory after getting the next element    }

function Queue.Next : Pointer;

var
  Temp : QueuePointer;

begin
  if Empty then begin
    Next := nil;
  end {then}
  else begin
    Temp  := Front;
    Next  := Front^.Data;
    Front := Front^.Link;
    if Empty then begin
      Rear := nil;
    end; {if}
    Dispose( Temp );
    Dec( QueueSize );
  end; {else}
end; {Queue.Next}

{------------------------------------***------------------------------------}

function Queue.Empty : Boolean;

begin
  Empty := (Front = nil);
end; {Queue.Empty}

{------------------------------------***------------------------------------}

function Queue.Size : Word;

begin
  Size := QueueSize;
end; {Queue.Size}

{------------------------------------***------------------------------------}

procedure Queue.Clear;

var
  Temp : QueuePointer;

begin
  while not Empty do begin
    Temp  := Front;
    Front := Front^.Link;
    if Temp^.Data <> nil then begin
      Dispose( Temp^.Data );
    end; {if}
    Dispose( Temp );
  end; {while}
  if Front <> nil then begin
    Dispose( Front );
  end; {if}
  if Rear <> nil then begin
    Dispose( Rear );
  end; {if}
  QueueSize := 0;
end; {Queue.Clear}

{------------------------------------***------------------------------------}

destructor Queue.Done;

var
  Temp : QueuePointer;

begin
  while not Empty do begin
    Temp  := Front;
    Front := Front^.Link;
```

```
    if Temp^.Data <> nil then begin
      Dispose( Temp^.Data );
    end; {if}
    Dispose( Temp );
  end; {while}
  if Front <> nil then begin
    Dispose( Front );
  end; {if}
  if Rear <> nil then begin
    Dispose( Rear );
  end; {if}
  QueueSize := 0;
end; {Queue.Done}

{---------------------------------***----------------------------------}

  {no initialisation code}

end. {Queues}

{---------------------------------***----------------------------------}
```

E.6 Module curves

```
{-----------------------------------***-----------------------------------}
{ Unit     : Curves.pas                                                  }
{ Language : Turbo Pascal v7.0                                           }
{ Function : Interpolation                                               }
{ Course   : MSc(Computer Science) University of Natal (Durban) 1994     }
{ Author   : Trevor Kistan                                               }
{ Source   : Numerical Analysis II notes                                 }
{-----------------------------------***-----------------------------------}

unit Curves;

interface

const Limit = 200;

type List = array[0..Limit] of Double;

type CoefficientMatrix = array[0..3] of List;

procedure CubicSpline( x, f : List; n : Word; var m : CoefficientMatrix );

function Cubic( x : Double; a, b, c, d : Double ) : Double;

function Lagrange( n : Word; x, y : List; p : Double ) : Double;

{-----------------------------------***-----------------------------------}

implementation

const a = 0;
const b = 1;
const c = 2;
const d = 3;

{-----------------------------------***-----------------------------------}
{ Instead of trying to fit a high-order approximating polynomial to a set of }
{ data points, a cubic spline fits a separate cubic polynomial between every }
{ two points in the data set - thus reducing wild oscillations.             }
{                                                                           }
{ Given a set of points (x[j], f[j]) for j = 0..n, st x[0] < x[n]1 < .. x[n] }
{ a cubic spline interpolation for the set of points is a function S(x) that }
{ satisfies the following conditions :                                      }
{                                                                           }
{ (a) S(x) is a cubic polynomial S[j](x) on the sub-interval [x[j], x[j+1]] }
{     for each j = 0,1,..n-1.                                               }
{ (b) S[j](x[j])         = f[j] for j = 0..n-1 & S[n-1](x[n]) = f[n]        }
{ (c) S[j+1](x[j+1])     = S[j](x[j+1) for j = 0..n-2                       }
{ (d) S'[j+1](x[j+1])    = S'[j](x[j+1) for j = 0..n-2                      }
{ (e) S''[j+1](x[j+1])   = S''[j](x[j+1) for j = 0..n-2                     }
{ (f) S''[0](x[0])       = S''[n-1](x[n]) - free spline (endpts not clamped) }
{                                                                           }
{ i.e. in addition to passing through all the points in the data set, the  }
{ set of curves, & their 1'st & 2'nd derivatives must also match at all     }
{ endpoints where they meet - a doubly smooth curve.                        }
{                                                                           }
{ S[j](x) is now expressed as :                                             }
{                                                                           }
{ S[j](x) = a[j] + b[j]*(x-x[j]) + c[j]*(x-x[j])^2 + d[j]*(x-x[j])^3        }
{                                                                           }
{ Expressing the set of simultaneous equations yielded by conditions (a)-(f) }
{ as a tridiagonal matrix and using Crout reduction yields the following    }
{ algorithm for determining the coefficients a[j], b[j], c[j], d[j].        }
{-----------------------------------***-----------------------------------}

procedure CubicSpline( x, f : List; n : Word; var m : CoefficientMatrix );

var
  j             : Word;
  h, l, u, z, t : List;

begin
```

```
  for j := 0 to n-1 do begin
    m[a][j] := f[j];
    h[j] := x[j+1]-x[j];
  end; {for}

  for j := 1 to n-1 do begin
    t[j] := 3*( m[a][j+1]*h[j-1] - m[a][j]*(x[j+1]-x[j-1]) + m[a][j-1]*h[j] )
            / (h[j-1]*h[j]);
  end; {for}

  l[0] := 1;
  u[0] := 0;

  for j := 1 to n-1 do begin
    l[j] := 2*( x[j+1]-x[j-1] ) - h[j-1]*u[j-1];
    u[j] := h[j] / l[j];
  end; {for}

  l[n] := 1;
  z[0] := 0;

  for j := 1 to n-1 do begin
    z[j] := (t[j] - h[j-1]*z[j-1]) / l[j];
  end; {for}

  z[n] := 0;
  m[c][n] := z[n];

  for j := n-1 downto 0 do begin
    m[c][j] := z[j] - u[j]*m[c][j+1];
  end; {for}

  for j := n-1 downto 0 do begin
    m[b][j] := (m[a][j+1]-m[a][j]) / h[j]  - h[j]*(m[c][j+1]+2*m[c][j])/3;
    m[d][j] := (m[c][j+1]-m[c][j]) / (3*h[j]);
  end; {for}

end; {CubicSpline}

{-------------------------------------***-----------------------------------}

function Cubic( x : Double; a, b, c, d : Double ) : Double;

begin
  Cubic := a + b*x + c*x*x + d*x*x*x;
end; {Cube}

{-------------------------------------***-----------------------------------}

function Lagrange( n : Word; x, y : List; p : Double ) : Double;

var
  i, j         : Word;
  Sum, Product : Double;

begin
  Sum := 0.0;
  for i := 0 to n do begin
    Product := 1.0;
    for j := 0 to n do begin
      if (i <> j) {and (p <> x[j]))} then begin
        Product := Product * (p-x[j])/(x[i]-x[j])
      end {then}
      else begin
        {null}
      end; {else}
    end; {for}
    Sum := Sum + y[i]*Product;
  end; {for}
  Lagrange := Sum;
end; {Lagrange}

{-------------------------------------***-----------------------------------}

end. {Spline}
```

```
{-----------------------------------------***-------------------------------------}
```

E.7 Module general

```
{----------------------------------***-------------------------------------}
{ Unit        : General.pas                                                }
{ Language    : Turbo Pascal (c) v6.0                                      }
{ Function    : PDP2 - Useful Non - Object Orientated General Routines     }
{ Course      : Computer Science Honours Project 1992 - University of Natal }
{ Author      : Trevor Kistan                                              }
{ Last Update : 16-10-1992                                                 }
{----------------------------------***-------------------------------------}
{ Licence     : Pdp2 is a complete rewrite of the Microsoft C (c) software }
{               released by the PDP Research Group, in conjunction with the }
{               text - 'Explorations in Parallel Distributed Processing:   }
{                       A Handbook of Models, Programs and Exercises'.     }
{               The rewrite is undertaken within the terms of the licence  }
{               agreement contained in the above text. (ver 1.1 1987)      }
{----------------------------------***-------------------------------------}
{ Original    : This file is part of the PDP software package.             }
{ Header        Copyright 1987 by James L. McCelland and David E. Rummelhart }
{               Please refer to licencing information in the file licence.txt}
{               which is in the same directory with this source file and is }
{               included here by reference.                                }
{----------------------------------***-------------------------------------}

(*{$I Compile.pas} {compiler and memory directives}*)

unit General;

interface

uses
  Dos, Crt;

const
  Zero         = '0';
  Blank        = ' ';
  DecimalPoint = '.';

function ValidRealStr(s : String) : Boolean;
{is it ok to try to convert string to a real number}

function ValidDoubleStr(s : String) : Boolean;
{is it ok to try to convert string to a double-precision real number}

function ValidWordStr(s : String) : Boolean;
{is it ok to try to convert string to a word number}

function ValidLongStr(s : String) : Boolean;
{is it ok to try to convert string to a longint number}

function UpperCase(s : String) : String;
{convert string to uppercase}

function LowerCase(s : String) : String;
{convert string to lowercase}

function TrimLine(Line : String) : String;
{remove leading and trailing blanks from string}

function Same(Item1, Item2 : String) : Boolean;
{case insensitive non-trimed string conversion}

function TrimReal(Number : String) : String;
{remove trailing decimal zeros from reals - adds 1 decimal 0 to 'integers'}

function RealToStr(Number : Real; spaces, decimals : Byte) : String;
{converts a real number into a string of given length with given decimals}

function DoubleToStr(Number : Double; spaces, decimals : Byte) : String;
{converts a real number into a string of given length with given decimals}

function WordToStr(Number : Word) : String;
{converts a word number into a string}
```

```
function LongToStr(Number : LongInt) : String;
{converts a longint number into a string}

function StrToReal(strNumber : String; var Number : Real) : Boolean;
{converts a string into a real number - returns true if successful}

function StrToDouble(strNumber : String; var Number : Double) : Boolean;
{converts a string into a double real number - returns true if successful}

function StrToWord(strNumber : String; var Number : Word) : Boolean;
{converts a string into a word number - returns true if successful}

function StrToLong(strNumber : String; var Number : LongInt) : Boolean;
{converts a string into a longint number - returns true if successful}

procedure ValDouble(strNumber : String; var Number : Double; var code : Integer);
{converts a string into a double-precision real number - returns actual conversion code}

procedure ValWord(strNumber : String; var Number : Word; var code : Integer);
{converts a string into a word number - returns actual conversion code}

procedure ValLong(strNumber : String; var Number : LongInt; var code : Integer);
{converts a string into a longint number - returns actual conversion code}

{------------------------------------***-------------------------------------}

implementation

const
  WordLimit   = 5;   {maximum number of digits allowed in representation}
  LongLimit   = 9;
  RealLimit   = 7;
  DoubleLimit = 15;

{------------------------------------***-------------------------------------}

function ValidRealStr(s : String) : Boolean;

{ is it ok to try to convert string to a real number }

var
  p : Byte;

begin
  p := Pos(DecimalPoint, s);
  if p = 0
    then ValidRealStr := ((Length(s) < RealLimit) and (s <> ''))
    else ValidRealStr := (((p-1) < RealLimit) and (s <> ''))
end; {ValidRealStr}

{------------------------------------***-------------------------------------}

function ValidDoubleStr(s : String) : Boolean;

{ is it ok to try to convert string to a double-precision real number }

var
  p : Byte;

begin
  p := Pos(DecimalPoint, s);
  if p = 0
    then ValidDoubleStr := ((Length(s) < DoubleLimit) and (s <> ''))
    else ValidDoubleStr := (((p-1) < DoubleLimit) and (s <> ''))
end; {ValidDoubleStr}

{------------------------------------***-------------------------------------}

function ValidWordStr(s : String) : Boolean;

{ is it ok to try to convert string to a word number }

begin
  if ((Length(s) < WordLimit) and (s <> ''))
    then begin
```

```
          if s[1] <> '-' {words cannot be negative}
            then ValidWordStr := True
            else ValidWordStr := False;
        end {then}
    else ValidWordStr := False;
end; {ValidWordStr}

{-----------------------------------***-------------------------------------}

function ValidLongStr(s : String) : Boolean;

{ is it ok to try to convert string to a LongInt number }

begin
   ValidLongStr := ((Length(s) < LongLimit) and (s <> ''));
end; {ValidWordStr}

{-----------------------------------***-------------------------------------}

function TrimLine(Line : String) : String;

{ deletes leading and trailing blanks from a string - tabs not removed ! }

begin
  while (Length(Line) > 0) and (Line[1] = Blank)  do
    Delete(Line, 1, 1); {leading}
  while (Length(Line) > 0) and (Line[Length(Line)] = Blank) do
    Dec(Line[0]);       {trailing}
  TrimLine := Line;
end; {TrimLine}

{-----------------------------------***-------------------------------------}

function TrimReal(Number : String) : String;

{ removes trailing decimal zeros from string representation of a real }
{ if a trailing zero occurs after the decimal point, it is retained   }
{ to indicate that the number is a real                               }

var
  DecimalPointPresent : Boolean;
  DecimalPosition     : Byte;

begin
  DecimalPosition     := Pos(DecimalPoint, Number);
  DecimalPointPresent := (DecimalPosition > 0);
  if DecimalPointPresent
    then while ((Length(Number) > DecimalPosition)
                and ((Number[Length(Number)] = Zero)
                or (Number[Length(Number)] = Blank))) do
           Dec(Number[0]); {remove all trailing decimal zeros}
  if Number[Length(Number)] = DecimalPoint
    then Number := Number + Zero; {if no digit after point add zero}
  TrimReal := Number;
end; {TrimReal}

{-----------------------------------***-------------------------------------}

function Same(Item1, Item2 : String) : Boolean;

{ case insensitive comparision of two strings - n.b. blanks not removed }

var
  i : Word;
begin
  if Length(Item1) <> Length(Item2)
    then Same := False
    else begin
           for i := 1 to Length(Item1) do
             begin
               Item1[i] := UpCase(Item1[i]);
               Item2[i] := UpCase(Item2[i]);
             end; {for}
           Same := (Item1 = Item2);
         end; {else}
```

```
end; {Same}

{----------------------------------***------------------------------------}

function RealToStr(Number : Real; spaces, decimals : Byte) : String;

{ converts a real number into a string of given length with given decimals }

var
  strNumber : String;

begin
  Str(Number:spaces:decimals, strNumber);
  RealToStr := TrimReal(strNumber);
end; {RealToStr}

{----------------------------------***------------------------------------}

function DoubleToStr(Number : Double; spaces, decimals : Byte) : String;

{ converts a double number into a string of given length with given decimals }

var
  strNumber : String;

begin
  Str(Number:spaces:decimals, strNumber);
  DoubleToStr := TrimReal(strNumber);
end; {DoubleToStr}

{----------------------------------***------------------------------------}

function WordToStr(Number : Word) : String;

{ converts a word number into a string }

var
  strNumber : String;

begin
  Str(Number, strNumber);
  WordToStr := strNumber;
end; {WordToStr}

{----------------------------------***------------------------------------}

function LongToStr(Number : LongInt) : String;

{ converts a longint number into a string }

var
  strNumber : String;

begin
  Str(Number, strNumber);
  LongToStr := TrimReal(strNumber);
end; {LongToStr}

{----------------------------------***------------------------------------}

function StrToReal(strNumber : String; var Number : Real) : Boolean;

{ converts a string into a real number - true if successful }

var
  code : Integer;
  s    : String;

begin
  s := TrimLine(strNumber); {remove blanks}
  if ValidRealStr(s)        {check validity}
    then begin
           Val(s, Number, code);     {attempt conversion}
           StrToReal := (code = 0);  {check attempt}
         end {then}
```

```
    else StrToReal := False;
end; {StrToReal}

{-----------------------------------***-----------------------------------}

function StrToDouble(strNumber : String; var Number : Double) : Boolean;

{ converts a string into a double real number - true if successful }

var
  code : Integer;
  s    : String;

begin
  s := TrimLine(strNumber); {remove blanks}
  if ValidDoubleStr(s)        {check validity}
    then begin
           Val(s, Number, code);     {attempt conversion}
           StrToDouble := (code = 0);  {check attempt}
         end {then}
    else StrToDouble := False;
end; {StrToDouble}

{-----------------------------------***-----------------------------------}

function StrToWord(strNumber : String; var Number : Word) : Boolean;

{ converts a string into a word number - true if successful }

var
  code : Integer;
  s    : String;

begin
  s:= TrimLine(strNumber);     {remove blanks}
  if ValidWordStr(s)           {check validity}
    then begin
           Val(s, Number, code);     {attempt conversion}
           StrToWord := (code = 0);  {check attempt}
         end
    else StrToWord := False;
end; {StrToReal}

{-----------------------------------***-----------------------------------}

function StrToLong(strNumber : String; var Number : LongInt) : Boolean;

{ converts a string into a longint number - true if successful }

var
  code : Integer;
  s    : String;

begin
  s := TrimLine(strNumber);    {remove blanks}
  if ValidLongStr(s)           {check validity}
    then begin
           Val(s, Number, code);     {attempt conversion}
           StrToLong := (code = 0);  {check attempt}
         end
    else StrToLong := False;
end; {StrToLong}

{-----------------------------------***-----------------------------------}

procedure ValReal(strNumber : String; var Number : Real; var code : Integer);

{ converts a string into a real number - returns actual conversion code }

var
  s : String;

begin
  s := TrimLine(strNumber);    {remove blanks}
  if ValidRealStr(s)           {check validity}
```

```
    then begin
           Val(s, Number, code);  {attempt conversion}
         end {then}
    else code := -1;  {force incorrect conversion code}
end; {ValReal}

{-------------------------------------***-------------------------------------}

procedure ValDouble(strNumber : String; var Number : Double; var code : Integer);

{ converts a string into a double real number - returns actual conversion code }

var
  s : String;

begin
  s := TrimLine(strNumber);    {remove blanks}
  if ValidDoubleStr(s)           {check validity}
    then begin
           Val(s, Number, code);  {attempt conversion}
         end {then}
    else code := -1;  {force incorrect conversion code}
end; {ValDouble}

{-------------------------------------***-------------------------------------}

procedure ValWord(strNumber : String; var Number : Word; var code : Integer);

{ converts a string into a word number - returns the actual conversion code }

var
  s : String;

begin
  s := TrimLine(strNumber); {remove blanks}
  if ValidWordStr(s)        {check validity}
    then begin
           Val(s, Number, code);   {attempt conversion}
         end {then}
    else code := -1;  {force incorrect conversion code}
end; {ValWord}

{-------------------------------------***-------------------------------------}

procedure ValLong(strNumber : String; var Number : LongInt; var code : Integer);

var
  s : String;

begin
  s := TrimLine(strNumber);
  if ValidLongStr(s)
    then begin
           Val(s, Number, code);
         end {then}
    else code := -1;  {force incorrect conversion code}
end; {ValLong}

{-------------------------------------***-------------------------------------}

function UpperCase(s : String) : String;

{ change a string to uppercase }

var
  i : Word;

begin
  for i := 1 to Length(s) do
    UpperCase[i] := UpCase(s[i]);
end; {UpperCase}

{-------------------------------------***-------------------------------------}

function LowerCase(s : String) : String;
```

```
{ change a string to lowercase ? }

var
  i : Word;

begin
  for i := 1 to Length(s) do
    if Ord(s[i]) in [65..90]          {if lowercase character}
      then s[i] := Chr(Ord(s[i])+32)  {add ascii offset}
end; {LowerCase}

{-----------------------------------***-------------------------------------}

  {no initialisation code}

end. {General}

{-----------------------------------***-------------------------------------}
```

E.8 Module vga

```
{-----------------------------------***-----------------------------------}
{ Unit         : Vga.pas                                                  }
{ Language     : Turbo Pascal v6.0                                        }
{ Function     : Image and screen display routines for vga screens       }
{ Course       : Computer Science Honours 1992 - University of Natal (Dbn) }
{ Author       : Trevor Kistan                                            }
{ Contributors : Dominic Dumee                                            }
{ Last Update  : 22-07-1992                                               }
{-----------------------------------***-----------------------------------}

Unit Vga;

Interface

uses
   Dos, Crt, Graph, FKeys;

const
   rasterloc = 164; {ima file conventions}
   sizeloc   = 80;
   ImageSize = 127;

type
   ImageType = array[0..ImageSize, 0..ImageSize] of Byte;

procedure SetGraphicsMode;
procedure SetGrayScale;
procedure SetColour( Colour, R, G, B : Word );
procedure DisplayGrayScale(xpos, ypos, inc : Word; PrintRange : Boolean);
procedure ReadImage(FileName : String; var Image : ImageType);
procedure SaveScreen(Filename : String);
procedure DisplayImage(x, y, xmin, xmax, ymin, ymax : Word; Image : ImageType);
procedure DrawFrame(x0, y0, x1, y1, inc, xt, BarColour, PanelColour : Word; Title : String);
procedure DrawSolidBox(x0, y0, x1, y1, Pattern, Colour : Word);
procedure DrawBackground(x, y : Word; Colour, Border, FillStyle : Byte);
procedure DrawIndicator(Value, MaxValue, xpos, ypos, Range : Word);
procedure DrawNumber(x, y, direction : Word; Number : Real; Range, Decimals : Word);
procedure DrawString(x, y, direction, mode : Word; TextString : String);
procedure SaveImage(FileName : String; var Image : ImageType);
procedure ReadReal(x, y : Word; var Value : Real; var code : Integer);
procedure ReadNumber(x, y : Word; var Value : String);
procedure DrawButton(x0, y0, x1, y1, LitColour, PanelColour, ShadowColour : Word);
procedure DrawPanel(x0, y0, x1, y1, LitColour, PanelColour, ShadowColour : Word);
procedure DrawBar(y, x0, x1, LitColour, ShadowColour : Word);
procedure DrawSlide( x0, y0, x1, y1, y,
                     LitColour, PanelColour, ShadowColour, TextColour : Word;
                     Title : String );

{-----------------------------------***-----------------------------------}

Implementation

procedure SetGraphicsMode;

var
   GraphDriver, GraphMode : Integer;

begin
   DetectGraph(GraphDriver, GraphMode);
   InitGraph(GraphDriver, GraphMode, '');
   if GraphResult <> grOK
      then begin
             Writeln('Error: Unable to locate Turbo Pascal file EGAVGA.BGI or ');
             Writeln('Graphics Error: ', GraphErrorMsg(GraphResult));
             Halt(1);
           end; {then}
end; {SetGraphicsMode}

{-----------------------------------***-----------------------------------}

procedure SetGrayScale;
```

```
var
   i, j    : Integer;
   Palette : Palettetype;
   r       : Registers;

begin
(*
   j := 0;
   for i := 0 to MaxColors do    {16 Grayscales}
     begin
        SetRGBPalette(i, j, j, j);
        Palette.Colors[i] := i;
        j := j + 4;
     end;
   SetAllPalette(Palette);
*)

{ The above turbo pascal routine is unreliable with some vga cards. }
{ It is replaced by the following routine which sets the palette    }
{ directly through the bios - courtesy of Dominic Dumee (1992).     }

   for i := 0 to 15 do
     begin
        r.ax := $1007;
        r.bx := i;
        intr($10,r);
        r.bl := r.bh;
        r.bh := 0;
        r.ax := $1010;
        r.ch := i*4;
        r.cl := i*4;
        r.dh := i*4;
        intr($10,r);
     end; {for i}

end; {SetGrayScale}

{-------------------------------------***------------------------------------}

procedure SetColour( Colour, R, G, B : Word );

var
  reg    : Registers;

begin
  reg.ax := $1007;
  reg.bx := Colour;
  intr( $10, reg );
  reg.bl := reg.bh;
  reg.bh := 0;
  reg.ax := $1010;
  reg.ch := R;
  reg.cl := G;
  reg.dh := B;
  intr( $10, reg );
end; {SetGrayScale}

{-------------------------------------***------------------------------------}

procedure DisplayGrayScale(xpos, ypos, inc : Word; PrintRange : Boolean);

var
   i, j      : Word;
   n, n1, n2 : String;

begin
  j := 16; {number of Grayscales}
  for i := 0 to MaxColors do
     begin
       SetFillStyle(1, i);
       Bar(xpos, ypos+(inc*i), xpos+inc, ypos+inc+(inc*i));
       if PrintRange
         then begin
                Str(i*8 ,n1);
                Str(((i+1)*8)-1, n2);
```

```
                n := n1 + '-' + n2;
                OutTextXY(xpos+25, ypos+10+(15*i), n);
              end; {if}
     end; {for}
  SetFillStyle(1,0);
  Rectangle(xpos, ypos, xpos+inc, ypos+(16*inc));
end; {DisplayGrayScale}

{-------------------------------------***----------------------------------}

procedure DrawFrame(x0, y0, x1, y1, inc, xt, BarColour, PanelColour : Word; Title : String);

var
  spos : Integer;

begin
  if inc > 0 then DrawSolidBox( x0, y0, x1, y0+inc, SolidFill, BarColour );
  DrawSolidBox( x0, y0+inc, x1, y1, SolidFill, PanelColour );
  if Title <> ''
    then begin
      if xt = 0
        then begin
          spos := x0+(((x1-x0) - Length(Title)*8) div 2);
          OutTextXY(spos, y0+6, Title);
        end {then}
        else begin
          OutTextXY(x0+xt, y0+6, Title);
        end; {else}
    end; {then}
end; {DrawFrame}

{-------------------------------------***----------------------------------}

procedure DrawNumber(x, y, direction : Word; Number : Real; Range, Decimals : Word);

var
  NumberString: String;

begin
  SetTextStyle(0, direction, 1);
  if ((Range = 0) or (Decimals = 0))
     then Str(Number, NumberString)
     else Str(Number:Range:Decimals, NumberString);
  OutTextXY(x, y, NumberString);
  SetTextStyle(0, 0, 1);
end; {DrawNumber}

{-------------------------------------***----------------------------------}

Procedure DrawString(x, y, direction, mode : Word; TextString : String);

begin
  SetTextStyle(0, direction, 1);
  if mode = 1  {inverse video}
     then begin
            OutTextXY(x, y, '█');
            SetColor(0);
          end;
  OutTextXY(x, y, TextString);
  if mode = 1 then SetColor(15);
  SetTextStyle(0, 0, 1);
end; {DrawString}

{-------------------------------------***----------------------------------}

procedure DrawIndicator(Value, MaxValue, xpos, ypos, Range : Word);

var
  Done       : Integer;
  DoneString : String;

begin
  SetColor(0);
  case Range of
       0  : OutTextXY(xpos, ypos, '');
```

```
        1  : OutTextXY(xpos, ypos, '█');
        2  : OutTextXY(xpos, ypos, '██');
        3  : OutTextXY(xpos, ypos, '███');
        4  : OutTextXY(xpos, ypos, '████');
        5  : OutTextXY(xpos, ypos, '█████');
        6  : OutTextXY(xpos, ypos, '██████');
        7  : OutTextXY(xpos, ypos, '███████');
        8  : OutTextXY(xpos, ypos, '████████');
        9  : OutTextXY(xpos, ypos, '█████████');
        10 : OutTextXY(xpos, ypos, '██████████');
{     else : OutTextXY(xpos, ypos, '███████████'); }
  end; {case}
  SetColor(15);
  Done := Round(Value/MaxValue*100);
  Str(Done, DoneString);
  OutTextXY(xpos, ypos, DoneString);
end; {DrawIndicator}

{--------------------------------------***-----------------------------------}

procedure DrawBackGround(x, y : Word; colour, border, fillstyle : Byte);
begin
  SetFillStyle(fillstyle, colour);
  FloodFill(x, y, border);
end; {DrawBackGround}

{--------------------------------------***-----------------------------------}

procedure SaveScreen(Filename : String);

var
  i, j       : Integer;
  PseudoIma  : File;
  buffer     : array[0..479] of Byte;

begin
  {Save screen in pseudo ima format for later printing}
  {note - no header or size info added}
  Assign(PseudoIma, FileName);
  ReWrite(PseudoIma, 1);
  for j := 639 downto 0 do
     begin
         for i := 0 to 479 do
            buffer[i] := Round((15-GetPixel(j, i))*8);
            {note intensity inversion}
         BlockWrite(PseudoIma, buffer, SizeOf(buffer));
     end;
  Close(PseudoIma);
end; {SaveScreen}

{--------------------------------------***-----------------------------------}

procedure ReadImage(FileName : String; var Image : ImageType);

var
   Ima          : File;
   NumRead      : Word;

begin
   FillChar(Image, SizeOf(Image), 0);
   Assign(Ima, Filename);
   {$I-}
   Reset(Ima, 1);
   {$I+}
   if IOResult <> 0
     then begin {file error}
            Writeln('Error: Unable to open file ',Filename);
            Halt(1);
          end {then}
     else begin {file input}
            Seek(Ima, rasterloc);
            BlockRead(Ima, Image, SizeOf(Image), NumRead);
            Close(Ima);
            if NumRead <> SizeOf(Image)
              then begin
```

```
                     Writeln('Warning: Image size incompatible with size of file ',Filename);
                     Writeln('Proceeding regardless ...');
                     Delay(1000);
                   end; {then}
          end; {else}
end; {ReadImage}

{--------------------------------------***-----------------------------------}

procedure DisplayImage(x, y, xmin, xmax, ymin, ymax : Word; Image : ImageType);

var
  i, j : Integer;

begin
   for j := ymin to ymax do
      for i := xmin to xmax do
         PutPixel(x+j, y+i, Round(Image[i, j] / 8.4666666));
end; {DisplayImage}

{--------------------------------------***-----------------------------------}

procedure SaveImage(FileName : String; var Image : ImageType);

var
   Head          : array[0..163] of Byte;
   i, NumWritten : Integer;
   Ima           : File;

begin
   Assign(Ima, Filename);
   {$I-}
   Rewrite(Ima, 1);
   {$I+}
   if IOResult <> 0
     then begin {file error}
            Writeln('Error: Unable to create file ',Filename);
            Halt(1);
          end {then}
     else begin {file input}
            for i := 0 to 163 do
               Head[i] := 0;
            Head[80] := 128;
            BlockWrite(Ima, Head, SizeOf(Head), NumWritten);
            BlockWrite(Ima, Image, SizeOf(Image), NumWritten);
            Close(Ima);
            if NumWritten <> SizeOf(Image)
              then begin
                     Writeln('Warning: Image size incompatible with size of file ',Filename);
                     Writeln('Proceeding regardless ...');
                     Delay(1000);
                   end; {then}
          end; {else}
end; {SaveImage}

{--------------------------------------***-----------------------------------}

procedure ReadReal(x, y : Word; var Value : Real; var code : Integer);

{ read in a real number with echo in graphics mode }
{ with elementary error checking and editing       }

var
  ThrStr      : String;
  Digit       : Char;
  Number      : String;
  i           : Byte;
  Key         : FKeys.Keys;
  FunctionKey : Boolean;

begin
   Number := '';
   i := 0;
   repeat
      FKeys.InKey( FunctionKey, Digit, Key );
```

```
      case Key of
        NumberKey : begin
                      Number := Number+Digit;
                      SetColor(Crt.Red);
                      DrawString(x+i*8, y, 0, 0, '█');
                      SetColor(Crt.White);
                      DrawString(x+i*8, y, 0, 0, Digit);
                      inc(i);
                    end;
        BkSp      : begin
                      if i > 0 then
                        begin
                          dec(i);
                          SetColor(Crt.Red);
                          DrawString(x+i*8, y, 0, 0, '█');
                          SetColor(Crt.White);
                          Delete(Number, Length(Number), 1);
                        end; {if}
                    end;
      end; {case}
   until ((Key = FKeys.CarriageReturn) or (Key = FKeys.Tab));
   Val(Number, Value, code);
end; {ReadReal}

{-------------------------------------***-----------------------------------}

procedure DrawSolidBox(x0, y0, x1, y1, Pattern, Colour : Word);

type
  Quad = array[1..4] of Graph.PointType;

var
  Box : Quad;

begin
  Box[1].x := x0;
  Box[1].y := y0;
  Box[2].x := x1;
  Box[2].y := y0;
  Box[3].x := x1;
  Box[3].y := y1;
  Box[4].x := x0;
  Box[4].y := y1;
  SetFillStyle(Pattern, Colour);
  FillPoly(4, Box);
end; {DrawSolidBox}

{-------------------------------------***-----------------------------------}

procedure ReadNumber(x, y : Word; var Value : String);

{ read in a real number with echo in graphics mode }
{ with elementary error checking and editing       }

var
  Digit       : Char;
  Number      : String;
  i           : Byte;
  Key         : FKeys.Keys;
  FunctionKey : Boolean;

begin
   Number := '';
   i := 0;
   repeat
      FKeys.InKey( FunctionKey, Digit, Key );
      case Key of
        NumberKey : begin
                      Number := Number+Digit;
                      SetColor(Crt.Red);
                      DrawString(x+i*8, y, 0, 0, '█');
                      SetColor(Crt.White);
                      DrawString(x+i*8, y, 0, 0, Digit);
                      inc(i);
                    end;
```

```
        BkSp       : begin
                       if i > 0 then
                         begin
                           dec(i);
                           SetColor(Crt.Red);
                           DrawString(x+i*8, y, 0, 0, '█');
                           SetColor(Crt.White);
                           Delete(Number, Length(Number), 1);
                         end; {if}
                     end;
      end; {case}
   until ((Key = FKeys.CarriageReturn) or (Key = FKeys.Tab));
   Value := Number;
end; {ReadNumber}

{--------------------------------------***-----------------------------------}

procedure DrawButton(x0, y0, x1, y1, LitColour, PanelColour, ShadowColour : Word);

begin
{  SetColor( Crt.Black+1 );}
{  Rectangle( x0-1, y0-1, x1+1, y1+1 );}
  DrawSolidBox( x0, y0, x1, y1, Graph.SolidFill, PanelColour );
  Graph.SetColor( LitColour );
  Graph.Line( x0, y0, x1, y0 );
  Graph.Line( x0, y0, x0, y1 );
  Graph.Line( x0+1, y0+1, x1-1, y0+1 );
  Graph.Line( x0+1, y0+1, x0+1, y1-1 );
  Graph.SetColor( ShadowColour );
  Graph.Line( x0, y1, x1, y1 );
  Graph.Line( x1, y1, x1, y0 );
  Graph.Line( x0+1, y1-1, x1-1, y1-1 );
  Graph.Line( x1-1, y1-1, x1-1, y0+1 );
end; {DrawButton}


{--------------------------------------***-----------------------------------}


procedure DrawPanel(x0, y0, x1, y1, LitColour, PanelColour, ShadowColour : Word);

begin

  DrawSolidBox( x0, y0, x1, y1, Graph.SolidFill, PanelColour );

  Graph.SetColor( LitColour );
  Graph.Line( x0, y0, x1, y0 );
  Graph.Line( x0+1, y0+1, x1-1, y0+1 );

  Graph.SetColor( ShadowColour );
  Graph.Line( x0+2, y0+2, x1-2, y0+2 );
  Graph.Line( x0+3, y0+3, x1-3, y0+3 );

  Graph.SetColor( LitColour );
  Graph.Line( x0, y0, x0, y1 );
  Graph.Line( x0+1, y0+1, x0+1, y1-1 );

  Graph.SetColor( ShadowColour );
  Graph.Line( x0+2, y0+2, x0+2, y1-2 );
  Graph.Line( x0+3, y0+3, x0+3, y1-3 );


  Graph.SetColor( ShadowColour );
  Graph.Line( x0, y1, x1, y1 );
  Graph.Line( x0+1, y1-1, x1-1, y1-1 );

  Graph.SetColor( LitColour );
  Graph.Line( x0+2, y1-2, x1-2, y1-2 );
  Graph.Line( x0+3, y1-3, x1-3, y1-3 );

  Graph.SetColor( ShadowColour );
  Graph.Line( x1, y1, x1, y0 );
  Graph.Line( x1-1, y1-1, x1-1, y0+1 );

  Graph.SetColor( LitColour );
```

```
  Graph.Line( x1-2, y1-2, x1-2, y0+2 );
  Graph.Line( x1-3, y1-3, x1-3, y0+3 );

end; {DrawButton}


{-----------------------------------***----------------------------------}

procedure DrawBar(y, x0, x1, LitColour, ShadowColour : Word);

begin

  Graph.SetColor( LitColour );
  Graph.Line( x0, y, x1, y );
  Graph.Line( x0, y+1, x1, y+1 );

  Graph.SetColor( ShadowColour );
  Graph.Line( x0, y+2, x1, y+2 );
  Graph.Line( x0, y+3, x1, y+3 );

end; {DrawButton}

{-----------------------------------***----------------------------------}

procedure DrawSlide( x0, y0, x1, y1, y,
                     LitColour, PanelColour, ShadowColour, TextColour : Word;
                     Title : String );

var
  spos, col : Word;

begin

  col := Graph.GetColor;
  DrawPanel( x0, y0, x1, y1, LitColour, PanelColour, ShadowColour );
  DrawBar( y0+y, x0+2, x1-4, LitColour, ShadowColour );
(*  setfillstyle(solidfill, lightgreen);
  bar(x0+4, y0+4, x1-4, y0+y-1);*)
  spos := x0+(((x1-x0) - Length(Title)*8) div 2);
  Graph.SetColor( TextColour );
  OutTextXY( spos, y0+y-5, Title );
  Graph.SetColor( col );

end; {DrawSlide}

{-----------------------------------***----------------------------------}

end.

{-----------------------------------***----------------------------------}
```

E.9 Module fkeys

```
{-----------------------------------***-----------------------------------}
{ Unit     : FKeys.pas                                                     }
{ Language : Turbo Pascal v6.0                                             }
{ Function : Returns Function Keys                                         }
{ Course   : Computer Science Honours 1992                                 }
{ User     : Trevor Kistan                                                 }
{ Source   : Stephen O'Brien - Turbo Pascal 6 The Complete Reference       }
{            Osborne MaGraw Hill (1991) pp 419-420                         }
{-----------------------------------***-----------------------------------}

Unit FKeys;

Interface

uses
  Dos, Crt;

type
  Keys = (NullKey, F1, F2, F3, F4, F5, F6, F7, F8, F9, F10,
          CarriageReturn, Tab, ShiftTab, Bksp,  UpArrow,
          DownArrow, RightArrow, LeftArrow, DeleteKey,
          InsertKey, HomeKey, Esc, EndKey, TextKey, NumberKey,
          Space, PgUp, PgDown);

procedure InKey(var FunctionKey : Boolean; var ch : Char; var key : Keys);

{-----------------------------------***-----------------------------------}

Implementation

procedure InKey(var FunctionKey : Boolean; var ch : Char; var key : Keys);

begin
   FunctionKey := False;
   ch := ReadKey;
   if (ch = #0)
     then begin
            FunctionKey := True;
            ch := ReadKey;
          end; {if}

   if FunctionKey
     then case Ord(ch) of
            15 : key := ShiftTab;
            72 : key := UpArrow;
            80 : key := DownArrow;
            82 : key := InsertKey;
            75 : key := LeftArrow;
            77 : key := RightArrow;
            73 : key := PgUp;
            81 : key := PgDown;
            71 : key := HomeKey;
            79 : key := EndKey;
            83 : key := DeleteKey;
            82 : key := InsertKey;
            59 : key := F1;
            60 : key := F2;
            61 : key := F3;
            62 : key := F4;
            63 : key := F5;
            64 : key := F6;
            65 : key := F7;
            66 : key := F8;
            67 : key := F9;
            68 : key := F10;
          end {case}
     else case Ord(ch) of
            8  : key := Bksp;
            9  : key := Tab;
            13 : key := CarriageReturn;
            27 : key := Esc;
            32 : key := Space;
```

```
            33..44, 47, 58..254 : key := TextKey;
            45..46, 48..57      : key := NumberKey;
         end; {case}
end; {InKey}

{------------------------------------***-----------------------------------}

end.

{------------------------------------***-----------------------------------}
```

E.10 Module ray

```
{-----------------------------------***-----------------------------------}
{ Program      : Ray.pas                                                  }
{ Language     : Turbo Pascal v7.0                                        }
{ Function     : Simplified command line raytrace without a user interface. }
{                Computationally this program provides the same functionality }
{                as the program Raytrace.  By providing a command-line version}
{                callable from a Dos batch file, we improve robustness.    }
{                If the raytracer crashes for a particular set of parameters, }
{                it returns to the Dos interpreter and the batch file calls  }
{                this program with the next set of parameters.             }
{ Course       : MSc (Computer Science) 1996 - University of Natal (Durban)  }
{ Author       : Trevor Kistan                                            }
{-----------------------------------***-----------------------------------}

program Ray;

uses
  Dos, Crt, Shocks;

var
  Shock                        : Shocks.TShock;
  Fire                         : Shocks.RaytraceParameterType;
  Density                      : Shocks.DensityParameterType;
  ParameterFile                : PathStr;
  DensityFile                  : PathStr;
  IntensityFile                : PathStr;
  sInterpolate                 : String;
  cInterpolate                 : Char;
  c1, c2, c3, c4, c5, c6,
  c7, c8,  c9, c10, c11, c12 : Integer;

begin

  if ParamCount = 16 then begin
    Val( ParamStr(1), Density.Vacuum.x, c1 );
    Val( ParamStr(2), Density.Vacuum.y, c2 );
    Val( ParamStr(3), Density.Tail.x, c3 );
    Val( ParamStr(4), Density.Tail.y, c4 );
    Val( ParamStr(5), Density.Saddle.x, c5 );
    Val( ParamStr(6), Density.Saddle.y, c6 );
    Val( ParamStr(7), Density.Rear.x, c7 );
    Val( ParamStr(8), Density.Rear.y, c8 );
    Val( ParamStr(9), Density.Front.x, c9 );
    Val( ParamStr(10), Density.Front.y, c10 );
    Val( ParamStr(11), Fire.FireRaysFrom, c11 );
    Val( ParamStr(12), Fire.FireRaysTo, c12 );

    sInterpolate := ParamStr(13);
    cInterpolate := sInterpolate[1];
    case UpCase( cInterpolate ) of
      'i' : Density.Interpolate := Shocks.Linear;
      'a' : Density.Interpolate := Shocks.Lagrange;
      'c' : Density.Interpolate := Shocks.Cubic;
      'j' : Density.Interpolate := Shocks.Junction;
      else  Density.Interpolate := Shocks.Linear;
    end; {case}

    ParameterFile := ParamStr(14);
    DensityFile   := ParamStr(15);
    IntensityFile := ParamStr(16);

    if (c1+c2+c3+c4+c5+c6+c7+c8+c9+c10+c11+c12) <> 0 then begin
      Writeln( 'Numeric Error in: ', ParameterFile );
    end {then}
    else begin
      Shock.Init;
      Shock.SetDensityProfile( Density );
      Shock.SetRayTraceLimits( Fire );
      Shock.FireRays;
      Shock.SaveParameters( ParameterFile );
      if IntensityFile <> 'none' then begin
        Shock.SaveIntensityPattern( IntensityFile );
```

```
      end; {if}
      if DensityFile <> 'none' then begin
        Shock.SaveDensityProfile( DensityFile );
      end; {if}
      Shock.Done;
      Writeln( ParameterFile, ' done.' );
    end; {else}
  end {then}
  else begin
    Writeln( 'Format error in call for ', ParameterFile );
  end; {else}

end. {Ray}
```

E.11 Module rayover

```
{-----------------------------------***-----------------------------------}
{ Unit        : RayOver.pas                                                }
{ Language    : Turbo Pascal v7.0                                          }
{ Function    : Overlay initialisation code                                }
{ Course      : MSc Computer Science University of Natal (Durban Campus)   }
{ Author      : Trevor Kistan                                              }
{ Last Update : 29/12/1994                                                 }
{-----------------------------------***-----------------------------------}
{-----------------------------------***-----------------------------------}

{$O-}  {this unit must NOT be overlayed}
{$F+}  {force far calls}

unit RayOver;

interface

implementation

uses
  Overlay, dos, crt;

const
  OvrMaxSize = 40000;      {large size is necessary for speed}

begin
  OvrInit('RayTrace.ovr');     {overlay file name}
  OvrInitEMS;                  {use EMS if available}
  OvrSetBuf(OvrMaxSize);       {set overlay buffer size}
  case OvrResult of
    ovrError         : Writeln('Overlay manager error');
    ovrNotFound      : Writeln('Overlay file not found');
    ovrNoMemory      : Writeln('Not enough memory for overlay buffer');
    ovrIOError       : Writeln('Overlay file i/o error');
    ovrNoEMSDriver   : Writeln('EMS driver not installed');
    ovrNoEmsMemory   : Writeln('Not enough EMS memory');
  end; {case}
end. {RayOver}

{-----------------------------------***-----------------------------------}
```

E.12 Module metrics

```
{-----------------------------------***-----------------------------------}
{ Program  : Metrics.pas                                                   }
{ Language : Turbo Pascal v7.0                                             }
{ Function : Computes the Halstead metrics vocabulary, length, volume, the }
{            McCabe Cyclomatic Complexity, DeMarco Information Weight and  }
{            Software Science Coding Time for a Pascal program.            }
{ User     : Trevor Kistan                                                 }
{ Course   : Computer Science Masters, University of Natal (Durban) 1996   }
{ Source   : Adapted and extended from -                                   }
{            Tom DeMarco - Controlling Software Projects:                  }
{            Management, Measurement & Estimation                          }
{            New York: Yourdon Press, 1982, pp 269-272.                    }
{-----------------------------------***-----------------------------------}

program Metrics;

const
  Ok            = 0;
  TokenSize     = 20;
  Quote         = '''';
  MaxVocabulary = 3000;
  HistoryLength = 2;

type
  List = record
    Count : 0 .. MaxVocabulary;
    Word  : array[0 .. MaxVocabulary] of string[TokenSize];
  end; {record}
  ShortString = string[TokenSize];
  MaxString   = string[255];
  History     = array[1..HistoryLength] of ShortString;

var
  Vocabulary                        : List;
  Token                             : ShortString;
  Line                              : MaxString;
  Filename                          : string;
  Source                            : Text;
  TotalTokens, UniqueTokens, i      : LongInt;
  TokenHistory                      : History;
  L, CommentLines, LOC, Complexity : LongInt;
  Procedures                        : LongInt;
  Weight, Time                      : Double;
  Vol, Ratio, rate                  : Double;
  WithinCase, StartCount            : Boolean;

{-------------------------------------------------------------------------}

function BlankLine( var TextLine : MaxString ) : Boolean;

{----------------------------}
{Determines if a line is blank.}
{----------------------------}

var
  Index         : Integer;
  FoundNonBlank : Boolean;

begin
  Index         := 0;
  FoundNonBlank := False;
  while ((Index < Length(TextLine)) and not FoundNonBlank) do begin
    Index := Index + 1;
    if (TextLine[Index] <> ' ') then begin
      FoundNonBlank := True;
    end; {if}
  end; {while}
  BlankLine := not FoundNonBlank;
end; {BlankLine}

{-------------------------------------------------------------------------}
```

```
function RemoveComments( var TextLine : MaxString ) : Boolean;

{------------------------------------------------------------}
{Note: only removes UCSD-style comments (enclosed in braces)}
{------------------------------------------------------------}

const
  BeginComment = '{';
  EndComment   = '}';

var
  TestString, NextLine              : MaxString;
  Index, Endex                      : Integer;
  Quoting, FoundComment, CommentLine : Boolean;
  FirstNonBlank                     : Boolean;

begin
  CommentLine   := False;
  FirstNonBlank := True;
{  repeat}
    Quoting      := False;
    FoundComment := False;
    Index        := 0;

    while ((Index < Length(TextLine)) and not FoundComment) do begin
      Index := Index + 1;
      if ((TextLine[Index] <> ' ') and FirstNonBlank) then begin
        FirstNonBlank := False;
        if (TextLine[Index] = BeginComment) then begin
          CommentLine := True;
        end; {if}
      end; {if}
      if TextLine[Index] = Quote
        then begin
          Quoting := not Quoting;
        end {then}
        else begin
          if TextLine[Index] = BeginComment then begin
            if not Quoting then begin
              FoundComment := True;
            end; {if}
          end; {if}
        end; {else}
    end; {while}

    if FoundComment then begin
      TestString := TextLine;
      Delete( TestString, 1, Index );

      (*--------------------------------------------------------------*)
      (* all case statements must end in: end; {case}                 *)
      (* set a global flag for the benefit of the complexity procedure *)
      (* when we reach the end of a case construct                    *)
      (*--------------------------------------------------------------*)
      if ((Pos( 'case', TestString ) = 1) and WithinCase) then begin
        WithinCase := False;
      end; {if}

      Endex := Pos( EndComment, TestString );
      if Endex > 0
        then begin
          Delete( TextLine, Index, Endex + 1 );
        end {then}
        else begin {comment runs over end of line}
          TextLine := Copy( TextLine, 1, Index-1 );
          repeat
            Readln( Source, NextLine );
          until Pos( EndComment, NextLine ) > 0;
          Delete( NextLine, 1, Pos( EndComment, NextLine ));
          TextLine := Concat( TextLine, NextLine );
        end; {else}
    end; {if}
{  until not FoundComment;}

  RemoveComments := CommentLine;
```

```
end; {RemoveComments}

{---------------------------------------------------------------------------}

function GetNextToken( var Token : Shortstring; var Line : MaxString ) : Boolean;

{-----------------------------------------------------}
{Module to isolate and deliver next operand or operator.}
{-----------------------------------------------------}

var
  SpacePointer : Integer;
  Phrase       : MaxString;

begin
  while Pos( ' ', Line ) = 1 do begin
    Delete( Line, 1, 1);
  end; {while}

  SpacePointer := Pos( ' ', Line );

  if Pos( Quote, Line ) = 1
    then begin
      Delete( Line, 1, 1 );
      if Pos( Quote, Line ) > 0
        then begin
          Phrase := Copy( Line, 1, Pos( Quote, Line) - 1 );
          Delete( Line, 1, Length( Phrase) + 1);
        end {then}
        else begin
          Phrase := Line;
          Line := ' ';
        end; {else}
    end {then}
    else begin
      if SpacePointer > 0 then begin
        Phrase := Copy( Line, 1, spacePointer - 1 );
        Delete( Line, 1, SpacePointer );
      end; {if}
    end; {else}

  GetNextToken := (Length( Phrase ) > 0);
  if Length( Phrase ) > TokenSize
    then begin
      Token := Copy( Phrase, 1, TokenSize );
    end {then}
    else begin
      Token := Phrase;
    end; {else}
end; {GetNextToken}

{---------------------------------------------------------------------------}

procedure LowerCase( var TextLine : MaxString );

{-------------------------------------------------------}
{Convert to lowercase -- beware: implementation dependent.}
{-------------------------------------------------------}

var
  Index : Integer;

begin
  for Index := 1 to Length( TextLine ) do begin
    if ((TextLine[Index] >= 'A') and (TextLine[Index] <= 'Z')) then begin
      TextLine[Index] := Chr(Ord(TextLine[Index]) - (Ord('A') - Ord('a')));
    end; {then}
  end; {for}
end; {LowerCase}

{---------------------------------------------------------------------------}

procedure Strip( var ProgLine : MaxString );

{--------------------------------}
```

```
{Module to strip away punctuation.}
{--------------------------------}

var
  Index : Integer;

begin
  while Pos( ':=', Line ) > 0 do begin
    Line[Pos( ':=', Line )] := '=';
  end; {while}
  for Index := 1 to Length( ProgLine ) do begin
    {-----------------------------------------------------------------}
    {each branch of a case statement contributes to the cyclomatic complexity}
    {-----------------------------------------------------------------}
    if (WithinCase and (ProgLine[Index] = ':')) then begin
      Complexity := Complexity + 1;
    end; {if}

    if ProgLine[Index] in [':', ';', ',', '(', ')', '[', ']'] then begin
      ProgLine[Index] := ' ';
    end; {if}
  end; {for}
end; {Strip}

{---------------------------------------------------------------------------}

function Unique( NewWord : ShortString; WordList : List ) : Boolean;

{-----------------------------------------------------------------}
{module to determine if a given word has ever been encountered before}
{-----------------------------------------------------------------}

var
  Index : Integer;

begin
  Unique := True;
  for Index := 1 to WordList.Count do begin
    if NewWord = WordList.Word[Index] then begin
      Unique := False;
    end; {if}
  end; {for}
end; {Unique}

{---------------------------------------------------------------------------}

procedure Link( NewWord : ShortString; var WordList : List );

{-----------------------------------------------}
{module to add a token to a list of known tokens}
{-----------------------------------------------}

begin
  WordList.Count := WordList.Count + 1;
  WordList.Word[WordList.Count] := NewWord;
end; {Unique}

{---------------------------------------------------------------------------}

function log2( Number : Integer ) : Real;

{------------------------------}
{return log base two of argument}
{------------------------------}

begin
  log2 := ln(number) / ln(2);
end; {log2}

{---------------------------------------------------------------------------}

procedure CalcComplexity( Token : ShortString );

{------------------------------------------------------------}
{Calculates the McCabe Cyclomatic Complexity of a program - this }
```

```
{is a count of all the if-then-else decision branches in a program.}
{An N-way case construct adds n-1 to the Cyclomatic Complexity.    }
{------------------------------------------------------------------}

begin
  if (Token = 'accept') or (Token = 'else') or (Token = 'if') or
     (Token = 'until')  or (Token = 'when') or (Token = 'while')
    then begin
      Complexity := Complexity + 1;
    end {then}
    else begin
      if (Token = 'case') then begin
        {----------------------------------------------------------}
        { Set a global flag so identification of all ':' adds 1    }
        { to complexity until end of case encountered - else part }
        { of case will be trapped by normal else search above.     }
        { The -1 of the n-1 part of the formula is set below.      }
        {----------------------------------------------------------}
        WithinCase := True;
        Complexity := Complexity - 1;
      end; {if}
    end; {else}

  TokenHistory[2] := TokenHistory[1];
  TokenHistory[1] := Token;
end; {Complexity}

{--------------------------------------------------------------------------------}

procedure CountProcedures( Token : ShortString );

{---------------------------------------------------------------}
{Counts the number of procedures and functions in the program.}
{---------------------------------------------------------------}

begin
  if (Token = 'interface')
    then begin
      {-----------------------------------------------------------}
      {Ignore procedure specifications/forward references in a unit.}
      {-----------------------------------------------------------}
      StartCount := False;
    end {then}
    else begin
      if (Token = 'implementation')
        then begin
          {-----------------------------------------}
          {Start counting once within the unit body.}
          {-----------------------------------------}
          StartCount := True;
        end {then}
        else begin
          if (StartCount and ((Token = 'procedure') or (Token = 'function') or
             (Token = 'constructor')  or (Token = 'destructor')))
            then begin
              Procedures := Procedures + 1
            end; {if}
        end; {else}
    end; {else}
end; {CountProcedures}

{--------------------------------------------------------------------------------}

begin
  {-------------------}
  {Get the source file.}
  {-------------------}
  repeat
    Write( 'Source File = ' );
    Readln( Filename );
    Assign( Source, Filename );
    {$I-} Reset( Source ); {$I+}
  until IOResult = Ok;

  {-----------------------}
```

```
{Initialise the counters.}
{------------------------}
UniqueTokens      := 0;
TotalTokens       := 0;
Complexity        := 0;
Vocabulary.Count  := 0;
LOC               := 0;
CommentLines      := 0;
Procedures        := 1;
WithinCase        := False;
StartCount        := True;

for i := 1 to HistoryLength do begin
  TokenHistory[i] := '';
end; {for}

{------------------------}
{Process the source file.}
{------------------------}
while not Eof( Source ) do begin
  L := L + 1;
  Write('.');
  Readln( Source, Line );
  {---------------------------}
  {Pad with at least one space.}
  {---------------------------}
  Line := Line + ' ';
  if not BlankLine( Line ) then begin
    if RemoveComments( Line ) then begin
      CommentLines := CommentLines + 1
    end {then}
    else begin
      LOC := LOC + 1;
    end; {else}
    LowerCase( Line );
    Strip( Line );
    while GetNextToken( Token, Line ) do begin
      TotalTokens := TotalTokens + 1;
      if Unique( Token, Vocabulary ) then begin
        Link( Token, Vocabulary );
        UniqueTokens := UniqueTokens + 1;
      end; {if}
      CalcComplexity( Token );
      CountProcedures( Token );
    end; {while}
  end; {if}
end; {while}

{----------------------------------}
{Calculate some intermediate values.}
{----------------------------------}
Vol    := (TotalTokens * log2( UniqueTokens ));
Weight := Vol * Complexity;
Time   := ((TotalTokens*Vol) / (4*18))/3600;
Ratio  := LOC / CommentLines;
Rate   := LOC / Time;

{------------------}
{Output the metrics.}
{------------------}
Writeln( 'Metrics for ', Filename );
Writeln( 'Lines of Code                  : ', LOC );
Writeln( 'Lines of Comments              : ', CommentLines );
Writeln( 'Code/Comment ratio             : ', Ratio:5:2 );
Writeln( 'Procedures                     : ', Procedures );
Writeln( 'Average Procedure Size         : ', Round( LOC / Procedures ));
Writeln( 'Halstead Length                : ', TotalTokens );
Writeln( 'Halstead Vocabulary            : ', UniqueTokens);
Writeln( 'Halstead Volume                : ', Round( Vol ));
Writeln( 'McCabe Cyclomatic Complexity   : ', Complexity );
Writeln( 'Average Complexity/Procedure   : ', Round( Complexity / Procedures ));
Writeln( 'DeMarco Information Weight     : ', Round( Weight ));
Writeln( 'Software Science Stroud No.    : 18' );
Writeln( 'Software Science Coding Time   : ', Time:5:2, ' hrs');
Writeln( 'Coding Rate                    : ', Rate:5:2, ' lines/hr' );
```

```
end. {Metrics}

{----------------------------------------------------------------------}
```