The Development of a Behavioural Video Analysis System and a Toolbox Prototype

by

Throshni Naidoo

Submitted in fulfilment of the
requirements for the degree of
Master of Science
in the
Department of Computer Science
and
Information Systems

University of Natal
Pietermaritzburg
1996

ABSTRACT

Behavioural research studies often include a data acquisition process, during which subjects are observed, and the displayed behaviours recorded, and a subsequent analysis process, during which the observed behaviours are analysed. In addition the data acquisition process could occur in real-time, or at a later stage, by referring to a recording of the original session.

This dissertation addresses the latter approach through the use of computer based digital video technology. A windows based video analysis system, called VAS, that was developed to assist with the acquisition of observed behaviours from the video, and the analysis thereof is discussed. This is followed by a discussion of the implementation of VAS. Finally the directions for further research are identified, and discussed.

PREFACE

The experimental work described in this dissertation was carried out in the Department of Computer Science and Information Systems, University of Natal Pietermaritzburg from February 1995 to May 1996, under the supervision of Mr. R. Dempster.

These studies represent original work by the author and have not otherwise been submitted in any form for any degree or diploma to any University. Where use has been made of the work of others it is duly acknowledged in the text.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

Many people have contributed suggestions and support to this research. In particular I would like to thank my supervisor, Mr. R. Dempster who was always only to willing to assist me. A big thank to the following people for their assistance, suggestions, and or support : Dr. Dempster, Dr. Zaverdinos, Mr. M.C. Clarke, Prof Velinov, the Usenet users who provided answers to my many questions, and all the behavioural researchers who willingly gave up their time to assist me.

I would also like to thank my family and my boyfriend, whose love assistance and support has made this possible.

1. INTRODUCTION

1.1 GENERAL OVERVIEW

Systematic observation is a key aspect in many research areas, the behavioural sciences being one of them.

Behavioural researchers often engage in sessions during which subjects are observed, and the observed behaviours recorded. The researcher may choose to conduct such sessions in real-time or alternatively record the session on videotape, and conduct these activities at a later stage.

There exists computer based behavioural analysis systems that could assist the researcher with the real time recording of observed behaviours, and thereafter further assist with the analysis of the information acquired. There are however certain limitations with some of the current systems, one of which is that errors in the recording of observed behaviours, are not accommodated for, i.e. there is no facility for backtracking and editing. In addition some systems do not provide the facility to record several simultaneous behaviours. Also, the recording of behaviours of more than one subject is not accommodated by certain systems.

There are other problems associated with the real time recording of observed behaviours, the most significant being, that it is often difficult for a researcher to simultaneously observe and record behaviours. The problem is further magnified by instances where the subjects under observation display several behaviours in quick succession.

Videotape recording of experiments is an affective means of overcoming some of these problems. The major advantage of this approach is that while the researcher need not be present during the conducting of the experiment, he/she would still have a perfect account of the actual experiment. Also, the recorded experiment may be replayed as many times as is necessary. In addition the information recording process need not be done in a

laboratory or the field and also does not have to be done in real time. It can be done at a time most convenient to the researcher, and in an environment that is more suitable.

With this approach however, the researcher is still faced with task of extraction of observational information from the videotape, and the consequent analysis thereof.

This task could be made easier still if there existed specific computer systems to assist with the extraction of observational data from the video (in analogue or digital form), and subsequently with the analysis of the observed behaviours.

The notion of building computer-based digital video systems is particularly appealing at present because of the considerable advances in digital video technology. Also, with the price of video capture boards and video editing tools decreasing [OZER95], it is now more affordable to build applications to exploit this technology.

1.2 STATEMENT OF OBJECTIVES

1. To examine the domain of computerised behavioural analysis systems, paying special attention to the need for computer based video analysis systems.

2. To develop a computer based video analysis system, called VAS.

3. To investigate the feasibility of producing a generic toolbox that could later be used to build applications, firstly in this domain, and secondly in other areas that would benefit from computer based video systems, such as teaching.

1.3 OVERVIEW OF DISSERTATION

Chapter 1 : This chapter provides an introduction to the intended area of research, and also outlines the objectives of the study.

Chapter 2 : Is a survey of related research. This includes a discussion of computerised behavioural analysis systems, as well as video technology. While all the referenced material will not be cited in this thesis it has all been included in the bibliography for completeness.

Chapter 3 : Details the specification and logical design of the intended system, based on interviews conducted with researchers in the field.

Chapter 4 : Discusses the implementation of VAS.

Chapter 5 : Contains a case study indicating the use of VAS. The current state of the system is discussed, and directions for future research indicated.

Chapter 6 : Concludes the dissertation by assessing the work done in terms of the objects stated in section 1.2

2 LITERATURE SURVEY

2.1 INTRODUCTION

A literature survey was conducted in two primary areas namely, computerised behavioural systems, and video technology. This chapter will deal with each of these areas in turn, in order to provide the required background for the work done.

2.2 COMPUTERISED BEHAVIOURAL SYSTEMS

An integral aspect of behavioural research, (and the focus of this study), is the recording and analysis of observational data. What follows is a chronological examination of the different methods used to perform these tasks.

Initially the behavioural researcher made observations using pencil and paper methods, (checksheets) i.e. the researcher would sit with a pencil and paper in hand, and note observations. The subsequent analysis was done manually. This method was inexpensive and allowed immediate access to the data. However it was very time consuming and tedious, hence increased the possibility of errors.

With the advent of computer technology, behavioural researchers quickly noticed the advantages of using the computer to perform the analysis of the observed data. Observational data was recorded using pencil and paper methods. Thereafter, this information was input into the computer, and analysed using the appropriate software.

The process of capturing the observed data to the computer was time consuming, laborious and error prone. It was soon realised that, if the observational data were recorded in a form that was more compatible with the computer software systems, this process of capturing observed data could be totally eliminated. This would also reduce the possibility of error between the point of data collection and analysis.

White [WHIT71] indicated that Tobach was the first to recognise this, in designing the ASTL system. This system was first described in 1962. Observations were indicated by the use of a computer keyboard. The system would record the state of the keyboard one, two or three times a second, directly onto paper tape. Some of the shortcomings of the ASTL system was that it was large, hence preventing the researcher from being mobile. It was also expensive, and noisy implying it may disturb the subjects under observation.

WRATS (White's Recording And Transcribing System) [WHIT71] was a computer oriented system, which was designed to overcome some of the problems associated with the ASTL system, and thus better assist the behavioural researcher.

This system used the keyboard to indicate observations. WRATS transmitted the states of a number of keys from a keyboard to the computer using intermediate storage on an audio tape. This involved an encoding process by which data from the keyboard was prepared for recording on magnetic tape, and a subsequent decoding process by which the data was recovered from the tape for use by the computer before being transcribed to paper tape. All detailed data analysis was performed on the paper tape transcriptions.

Some of the disadvantages of WRATS was that it required specialised hardware and software for the transcribing and subsequent analysis of data. Also observations of more than one subject could not be easily accommodated.

The Datamyte 900 [TORG77] was a hand-held solid state data collector that was designed to assist the behavioural researcher with the collection of observed data, and the transmission of that data directly to the computer for immediate analysis. The system was based on and transmitted ASCII code. Pressing the appropriate button on the collector resulted in the stored data being transmitted to the terminal (for printing, recording on cassette or paper tape) or to the computer for storage on file.

The major disadvantage with this system was that it could not accommodate the observations of more than one subject at a time. Hence multi-subject observations were done by multiple passes of a video tape containing the experiment, and in each pass a new

subject became the focus. Alternatively many observers, each equipped with a Datamyte 900 could be used.

The early 1980's saw many rapid developments in the computer industry, and in particular in large scale integration, thus resulting in powerful microcomputers becoming available at prices comparable with, if not less, than conventional activity recording equipment. (e.g. Datamyte 900)

It followed that the use of computers in this field could now be extended from purely analysis of information, to include the recording of observational data.

The literature records examples where computers were used for the recording of observational data (using dedicated software), and the subsequent analysis thereof. Some of these examples will now be described.

Flowers [FLOW82] illustrated the use of the Apple II in collecting and analysing observational data. In his paper two systems were discussed, the BCODE system, which was designed for monitoring mutually exclusive (temporally nonoverlapping) behaviours, and the TACT3 system, which was designed to allow the researcher to monitor up to three behavioural states that may occur simultaneously.

For both software packages, the programs for data acquisition were kept separate from the programs used in the data analysis and summary. This assisted in maintaining program simplicity whilst offering the researcher the opportunity to create their own analysis routines in addition to the ones provided.

A major disadvantage of these systems, was that they were designed to be used in laboratory settings, hence they could not be used in the field.

Deni *et. al.* [DENI83] illustrated the use of the TRS-80 Model 100 portable and Model 4 computers in behavioural research. Programs were written to allow the TRS-80 Model 100 portable computer to record data from behavioural observations and to transfer those data to a larger system, the Model 4, for storage and further analysis.

The RECORD program for the Model 100, was used during the actual observation session. This program was used for the acquisition of observational data. The data acquired was stored to a file that was read by the Model 4 utility program TAPE100. The REDUCE program for the Model 4 was used to summarise the observational data, and perform certain analyses.

Berry *et. al.* [BERR87] described the use of the Radio Shack Model 100 laptop computer for collecting data in the field. The GATHER program was used for the data collection. The processing and analysis of the observed data was done using the IBM-PC. Data was transferred from the Model 100 to the IBM-PC using the software TRANS.BA and RECV2X.COM via the RS-232 interface.

Unwin and Martin, [UNWI87] described a microcomputer based event recorder for recording behavioural observations. The system used the Epson PX-8 microcomputer, and included two different methods of making observations. It combined the advantages of the keyboard event recorder methods with the ease of use and flexibility of the pencil and paper methods. (checksheets). Using the Checksheet emulator program a time sampled record of complex behaviour patterns could be acquired. With the event recorder program frequencies, duration and latencies of many different categories of behaviour, could be recorded.

Whiten and Barton, [WHIT88] describes the use of miniature hand-held computers (the HP71, and HP41) in remote fieldwork applications. It was found that the HP71 that was small enough to fit in the hand or pocket was more than adequate when it came to storing an entire day's data collected observing baboons in the field.

The system described by Whiten and Barton [WHIT88] was most suitable to fieldwork hence was limited in its use. Also, this system was used primarily for the recording of observations, hence for the analysis, this information had to be transferred to larger computer systems.

The Observer described by Noldus *et. al.* [NOLD89], Noldus [NOLD91], and Visser [VISS93] is a general-purpose software package for event recording and data analysis that is widely used in behavioural research at present. It allows the IBM-type personal computer (or compatible) to serve as an event recorder. It also accommodates data collected with the following hand-held computers, the Psion Organizer II LZ/Z64, the Huskey Hunter 2/16, the Epson PX-4/PX-8, the TRS-80 Model 100, the Tandy 102, and the Olivetti M10. Data collected with the hand held computers can be uploaded into the PC for further analysis and or storage.

The Observer consists of three separated software modules, the Configuration Module, Event Recording Module, and Data Analysis Module. All of these are menu driven with help screens explaining the underlying tasks. These modules will now be described in more detail.

Configuration Module : This is where the research design is specified. In addition the independent variables, subject(s) under observation, behavioural events being monitored, and other set-up information is specified. This module is used to tailor the Event Recording session to specific experimental designs. This configuration is saved to a file on disk.

Event Recording Module : A configuration could be used directly for event recording (data collection) on the PC. For event recording on a hand-held computer, the configuration can either be used directly or passed to a program generator that creates a data collection program to be used on the hand-held computer. To permanently store and or, further analyse the data collected by a hand-held computer, the Observer uploads the data to the PC via the RS-232 interface.

Data Analysis Module : The analysis software calculates frequencies, mean duration and total duration for classes, events and concurrencies of events in different classes. Output files clearly indicate the information generated during the analysis. Further analyses could be carried out by transferring the observed data to a spreadsheet or statistical package.

The structure of The Observer was described by Noldus *et. al.* [NOLD89][NOLD91] [NOLD93], Hile [HILE91], as well as Albonetti *et. al.* [ALBO92], Tourtellot [TOUR92], and Wawra [WAWR94].

With the use of small computers in field applications, it was found that it was very often difficult to watch an ongoing interaction and simultaneously activate the different keys on the keyboard. Another issue of concern was that, during the data acquisition process there were occasional instances, where the subject(s) displayed many behaviours in quick succession. This posed a problem in that, not only did the event recorder have to be able to accommodate this, but the researcher (observer) as well. Also, many of the event recorders discussed so far, did not accommodate for errors being made by the researcher (observer), during the data recording process in that there was no facility for backtracking and editing.

In order to overcome these problems and produce more accurate data the experiment could be videotaped. The observational data could then be acquired from the video at a later stage.

To facilitate the acquisition of data, also known as scoring, from the videotape, Noldus developed a Video Interfacing Module. This module was described by Noldus [NOLD94A], Wawra [WAWR94], and Keene [KEEN94].

The hardware and associated software read time codes directly from the videotape, which allowed event timing at variable playback speeds. To prevent errors the system displayed observed behaviours on the screen. This system also included search and editing functions, and permitted the control of the video recorder from the keyboard.

The Noldus suite of software is constantly being upgraded, and extended in order to better assist the behavioural researcher. A recent addition, described by Noldus [NOLD94B] is the EthoVision module. This module, can be used for the automatic recording of activity, movement and interactions of animals. It is a computer vision system for behavioural research based on digital image processing techniques.

Fentress[ALBO92] stated that, The Observer, like other software packages did have certain limitations. Despite this however, Fentress referred to The Observer as being "simply the best program yet developed for doing what it intended to do", and The Observer was also highly recommended by several other behavioural researchers.

The major drawback of The Observer however, was the cost as indicated by Wawra [WAWR94]. Wawra stated that certain components (e.g. The Video Tape Analysis System) was considered too expensive for some members of universities.

The literature surveyed presents many different computerised systems for real time recording of observational data, and the subsequent analysis thereof. However the literature has indicated certain problems associated with real time recording of information. Two of the most significant being :

- It is often difficult for an observer to watch an ongoing interaction and simultaneously activate the different keys on the keyboard.
- Occasionally there are instances when the subjects display several behaviours in quick succession, which makes real-time recording of observed behaviours virtually impossible.

In order to attain more accurate results, videotape recordings of the experiments are used. The researcher is then faced with the task of extracting the observational information from the videotape, and thereafter performing the necessary analyses. Of the computerised systems surveyed, only the Noldus Videotape Analysis System could be used to assist the researcher with such tasks. Hence it can be seen that there is a very real need for computerised systems to handle behavioural information on video .

2.3 VIDEO TECHNOLOGY

Video whether in analogue or digital form, is an electronic representation of a sequence of images. These images are called frames.

2.3.1 ANALOG VIDEO REPRESENTATION

Analogue video is at present not typically stored and manipulated within computer systems, however a knowledge of analogue video is essential in order to appreciate the capabilities of digital video. Also, even though the computer and hardware may handle video in a digital form, it often enters as an analogue signal.

Analogue video is a continuous electrical signal, which is encoded by changes to signal amplitude. There are various video formats, each of which specify rules governing the structure of these signals. These rules determine the breakdown of the signal into frames, frames into scan lines, the way in which colour is represented, and the placement of synchronisation information. [GIBB95]

The key dimensions of any format include :

- **Frame Rate** : This is the number of frames per second produced by the video signal. Common rates are 25-75Hz (i.e. 25-75 frames per second, or fps). The disadvantage of low frame rate signals is that they flicker when displayed. Also at lower frame rates moving objects appear blurred within individual frames, and motion is jerky.
- **Number of Scan Lines** : Scanning is the process of converting an image to an electrical signal by moving a sensing point across the image usually in a pattern from left, to right and repeated from the top to the bottom as a series of horizontal lines. [LUTH89]. These lines are referred to as scan lines. Each frame consists of a set of scan lines. The number of scan lines per frame is constant.
- **Aspect Ratio** : The ratio of the width of a video image to its height.

- **Interlacing** : Low frame rates cause flicker. The flickering can be reduced by interlacing. Interlacing refers to a concept where more than one vertical scan is used to produce a frame. With a 2:1 interlace, one vertical scan displays all the odd-numbered scan lines of the frame, and a second scan puts in all the even numbered scan lines. Thus the display appears to refresh at twice the original frame rate.
- **Quality** : Different grades of video equipment are produced for the different classes of purchasers, namely the *consumer*, *professional*(or industrial), and *broadcast*. The main difference between the product lines, besides the cost, is the quality of the video produced. Quality is measured in objective terms including signal-to-noise ratio and image resolution.
- **Composite video versus component video** : With composite video the luminance signal (intensity information) and chrominance signal (colour information) are combined into a single electrical signal. With component video there are multiple signals. Component video generally provides better quality. Its disadvantages are more elaborate cabling, and the need to maintain synchronisation between the different components. [GIBB95]

The major analogue video formats are :

- NTSL (National Television Systems Committee) : A television standard developed in the USA, and used throughout North America, Central America and Japan, and some parts of the South Pacific and South America.
- PAL (Phase Alternation Line) : A European television standard used in much of Western Europe, India, China, Australia, and parts of Asia and South America.
- Secam (Séquentiel Couleur avec Mémoire) : The French television standard, used in Eastern Europe, Russia and parts of Africa and the Middle East.
- RGB : A component video format with separate red, green, and blue signals. RGB is commonly used for computer displays.

The three standards NTSL, PAL, SECAM, have both composite and component representations.

2.3.2 DIGITAL VIDEO REPRESENTATION

A digital video system is any system where the information for images is represented by a series of digital bits.

Analogue video is a sequence of frames, encoded as a continuous electrical signal that occurs at a constant rate, the frame rate. With digital video there is once again a sequence of frames, which are now digital images. This implies all information representing images is in some kind of computer data form, which can be manipulated and displayed by the computer [LUTH89]. This information may be compressed in some manner. The essential feature of digital video is that, it is a sequence of images with accompanying synchronisation information, because frames are meant to occur at a particular rate. [GIBB95].

Some key issues that characterise digital video include :

- **Analogue formats sampled :** Digital video frames are generally obtained in two ways, synthesis (usually by a computer program) and sampling (of an analogue video signal). In the latter case, the characteristics of the underlying analogue signal (i.e. the frame and scan rates, method of colour coding, etc.) will influence the form of the digital representation.
- **Sampling and quantization** : The analogue signal is continuous in both value (amplitude) and in time. In order to convert the analogue signal to a digital signal, both these dimensions must be changed to non-continuous values. The amplitude is represented by an integer that is represented by a specified number of bits. The time is represented by a series of those integer amplitude values taken at equal steps in time. The process of converting time into discrete values is called sampling, and the analogous process of converting amplitude into discrete values is called quantizing. These two processes together is referred to as analogue-to-digital conversion (A/D conversion) or digitising.

The choice of sampling rate is crucial in specifying a digital format for sampled analogue video, since its value determines the storage requirements and data transfer rates. Both these processes are described by Gibbs [GIBB95], and Luther [LUT89]. Sample size is determined by the number of bits used to represent sample values. Factors that influence the choice of sample size include the signal-to-noise ratio of the sampled signal, and the sensitivity of the medium used to display frames. If the sample size was carefully chosen taking these factors into account, the effects of pixellation and contouring can be reduced. The pixellation effect refers to the blocky appearance of a digital image. The visibility of the pixellation effect is dependent on the medium used for the display. Contouring occurs because all analogue levels that fall between two thresholds of the quantizer are replaced with the same digital value, and will thus be reproduced at the output of the digital system with the same value also. For noise patterns that are coherent or correlated with the image, specific sample sizes must be used to minimise the contouring effect. [LUTH89]

Digital video commonly use linear quantization. This form of quantization is the simplest and is convenient for the arithmetic processing of sample values.

We can divide digital video representations very broadly, into two categories, namely high data rate formats and low data rate formats. The high data rate formats are primarily used in professional video production and post-production. There is little or no compression, and picture quality and ease of processing are of primary importance. The low data rate formats are intended for both interactive computer-based applications and transmission over general-purpose data networks. In order to achieve a low data rate (with acceptable picture quality) compression is necessary.

HIGH DATA RATE FORMATS

Examples of high data rate formats include :

- Digital component video (CCIR 601)
- Digital composite video

- Common intermediate format (CIF)
- Quarter-Common intermediate format (QCIF)

LOW DATA RATE FORMATS

Dimensions for comparing these formats include :

- **Data rate** : The high data rate formats can be converted to lower rate formats by a combination of data compression, (reducing the horizontal and vertical resolution), and reducing the frame rate.
- **Frame rate** : The basic issue is whether the format is intended for playback at analogue video frame rates i.e. 25 or 30 frames per second or not. This is also referred to as full-motion video. When the video frame rate is reduced to 15-20 frames per second, motion is less accurately depicted and the playback image flickers. The data rate is also much reduced.
- **Compression** : Compression is the key to low data rate digital video. Many video compression techniques have been developed. These include :

  - simple compression techniques (e.g. truncation, CLUT, run-length)
  - interpolative techniques (e.g. colour subsampling)
  - predictive techniques (e.g.. DPCM)
  - transform coding techniques (e.g.. Fourier, Hadamard)
  - statistical coding techniques (e.g.. Huffman coding).

  Gibbs [GIBB95] and Luther [LUTH89] have described three dimensions according to which video compression can be compared. These are:

  - **Lossy versus lossless** : With lossless compression, the image is preserved by the compression/decompression process, however with lossy compression, some information is lost during this process. Generally video compression can afford to be lossy, since it is often not critical that the reconstituted data exactly match the data prior to compression.

- **Real-time compression** : With motion video / live video compression systems, speed is a critical factor. With live video, the frame rates are between 25-30 frames per second, which implies that the digital video system must deliver a new image every 30-40 milliseconds. If this is not done then motion will be slow and jerky. In addition to requiring greater speed for real-time compression, greater compression is also required. This is so because of the data rate limitations of storage media.
  If, however, an application uses only pre-recorded, as apposed to live video, then real-time compression is not necessary. Thus a digital video format might choose not to record in real time in favour of better image quality, or a lower data rate.
  Compression schemes that have near equal compression and decompression times are said to be symmetric.
- **Interframe versus intraframe (relative versus absolute) compression** : Interframe (also called frame-to-frame) compression schemes assumes that not a lot of the image changes from frame to frame. This scheme will take a single full frame of the video and mark it as the key frame, and thereafter analyse each subsequent frame recording only the changes between that frame and the previous frame. A key frame is one that differs significantly from preceding (or following) frames. With intraframe schemes, all frames are independently encoded.

The video quality at a given data rate depends upon the techniques used i.e. the low data rate formats may be compared by comparing their quality (resolution and frame rate) at a particular data rate. The goal of the compression algorithm is to avoid sacrificing too much video quality in attaining a particular data rate.

Support for interactivity : Requirements for interactive applications include random access to video frames, the ability to playback at different rates and in reverse. Support for these features depend on the compression scheme used. For example with the inter-frame schemes, where frames are derived from previous frames, reverse play is difficult.

Scalability : Scaleable video allows control over video quality. Transmit scalabilty refers to the case where the encoded data rate is chosen at compression time. Thus this encoded

rate can accommodate transmission and processing constraints as well as storage constraints of the resulting data. Receive scalability refers to the case where the decoded data rate is chosen at decompression time in order to match playback requirements.

Transmit scalability is present in current approaches to low data rate digital video since there is flexibility in choosing frame resolution, frame rate, and compression factors. Receive scalability is not currently supported by video coding standards. [GIBB95]

Some approaches to low data rate digital video include :

- Digital Video Interactive (DVI)
- The Moving Pictures Expert Group (MPEG)
- px64

The literature surveyed provided a greater insight into digital video representation, as well as analogue video representation, and hence provided the background knowledge for the work using digital video, which was to be performed.

3 SYSTEM SPECIFICATION

3.1 INTERVIEWS

In order to gain a more comprehensive understanding of the problem various researchers using video tape recording as an observational tool, were interviewed.

Dr. Dempster, when in the Department of Zoology at the University of Natal video taped experiments conducted on pairs of gerbils. The gerbils were placed in a glass box with a piece of wire mesh separating them. At the start of the experiment, the wire mesh was removed and the recording thus began. The video recorder used did not have the facility to include the time, so a kitchen clock was placed in front of the glass box such that it was visible at all times. Using the kitchen clock as the timing mechanism, the rodents were taped for fifteen minutes. Dr. Dempster had constructed a list of target behaviours that she was looking for. When extracting data from the video tape (scoring the video tape), she would sit in front of the television with the remote control, pen and paper. She would then pause the video as and when required to encode all the behaviours that were displayed by each rodent in terms of ten second intervals. After the information extraction process, she would calculate all behaviour and behaviour/rodent frequencies, as well as the duration each rodent displayed a particular behaviour. Thereafter a sequencing analysis was conducted. Dr. Dempster did all her information extraction by hand. This was a time consuming process. Due to the facilities at her disposal it was difficult for her to acquire exact time measurements.

Dr. Mike Lawes, an ethologist in the Department of Zoology at the University of Natal frequently observes birds. He identified three instances when he would use a video tape as an observational tool. These were :

- To observe sequences of behaviours displayed by the subject
- To attain a detailed summary of the subjects behaviour
- To observe nested behaviours

Dr. Lawes used a real time event recorder to extract information from a video. This mechanism however, had the disadvantage of not accommodating errors. That is, if one had incorrectly encoded any information at any point, then all work up to that point had to be discarded and the entire process restarted. Basically there was no facility for backtracking and editing. The event recorders used were unable to accommodate nested behaviours. This is a concept that refers to the case where one behaviour induces another behaviour and this new behaviour lasts for the duration of the former behaviour. e.g. : Feeding behaviour may induce the rubbing of the tummy and this rubbing of the tummy lasts throughout the duration of the feeding behaviour. A roaming behaviour may induce a growling, aggressive behaviour. Another problem associated with the event recorder was that it did not allow for the possibility of more than one behaviour occurring at any particular time. E.g. A mouse may exhibit a sniffing behaviour whilst simultaneously scratching its ear. Measurement of distances was another issue that posed a problem to Dr. Lawes. He often had to measure the distance between two birds or the length of the birds tail, but he did not have the facility to do this, directly from the videotape.

Dr. Jane Koalsvig at the Medical School in the University of Natal Durban is a behavioural researcher who observes people. She used a video recorder to observe a section of a river that was known to be contaminated with a particular disease. She would then observe the different people using the river, the activities they engaged in and the percentage of their body exposed to the water. Dr. Koalsvig thus needed a method of calculating the surface area of a persons body that was exposed to the water. After acquiring all the data she calculated an exposure index which indicated which activities were more likely to cause the particular illness. At the moment Dr. Koalsvig does her data extraction and analysis by hand. She often has to rely on approximations which thus reduces the accuracy of her findings.

Dr. Sue Marriner at the University of Durban Westville is an ethologist who often observes horses. She also uses video in her research. She often looked at foal/mare relationships and how well a foal actually learnt grazing behaviour from its parent. Measures of speed were important in determining the intensity of certain behaviours. If Dr. Marriner had a

means of calculating the distance traversed between two points, she could then easily calculate the speed. At present she has no means of attaining such measures.

Prof. Lachenicht in the Department of Psychology at the University of Natal specialised in observing the gestures of people. He often had more than one person viewing and scoring a particular video with a predefined set of target behaviours. Very often it is found that a particular observation was interpreted differently by different people. When this disagreement went beyond an accepted norm there was said to be no inter-rater reliability and the points of disagreement needed to be carefully examined. Currently the comparisons of the data extracted by different individuals are conducted manually. This task is an extremely time consuming one.

Prof. Basson is another researcher in the Department of Psychology at the University of Natal, who makes use of video recording in his work. His focus was slightly different, though. He used video tapes as a training aid. He recorded counselling sessions and by referring to sections in the video asked questions and taught skills. At the moment he does this by editing the video tape and recording his comments and questions on the tape. He would like a facility whereby the video tape could be played and he could ask his questions in the form of text in a window that could appear in a corner of the screen. This would enable one to look at a particular video from more than one perspective.

The issues that need to be addressed, as identified by the interview conducted, are :

- The scoring of a video tape. This must also include tools to perform certain calculations such as distances, speeds and areas.
- The analysis of the observed data.

The interviews conducted served to reinforce the need for the development of the proposed system. They also assisted with the ultimate formulation of the system objectives.

3.2 ANALYSIS OF THE PROBLEM

The interviews formed the basis for defining the scope of the proposed video analysis system.

3.2.1 STATEMENT OF PURPOSE

The purpose of the **V**ideo **A**nalysis **S**ystem (**VAS**) is to facilitate the acquisition and recording of information from observations on video, and the subsequent analysis thereof

3.2.2. DATA FLOW DIAGRAMS

The processes that would need to be performed by VAS, as identified during the interviews are illustrated in the following data flow diagrams.

3.2.2.1 OVERVIEW OF VAS

An overview of VAS in terms of the processes required is illustrated in Figure 3.1. These are :

- **Design an Experiment** : Specify a collection of information that defines a particular investigation.
- **Edit an Experiment** : Modify information that defines an experiment.
- **Copy an Experiment** : Create a duplicate of an experiment.
- **Create a Project** : Create an instance of an experiment (trial).
- **Score a New Project** : Extract observational information from a project (trial).
- **Edit a Scored Project** : Edit observed information.
- **View a Project** : View a scored project (trial).

- **Analyse a Project** : Perform analyses on the observed information.
- **Compare Projects** : Perform a comparison of two scored projects(trials).

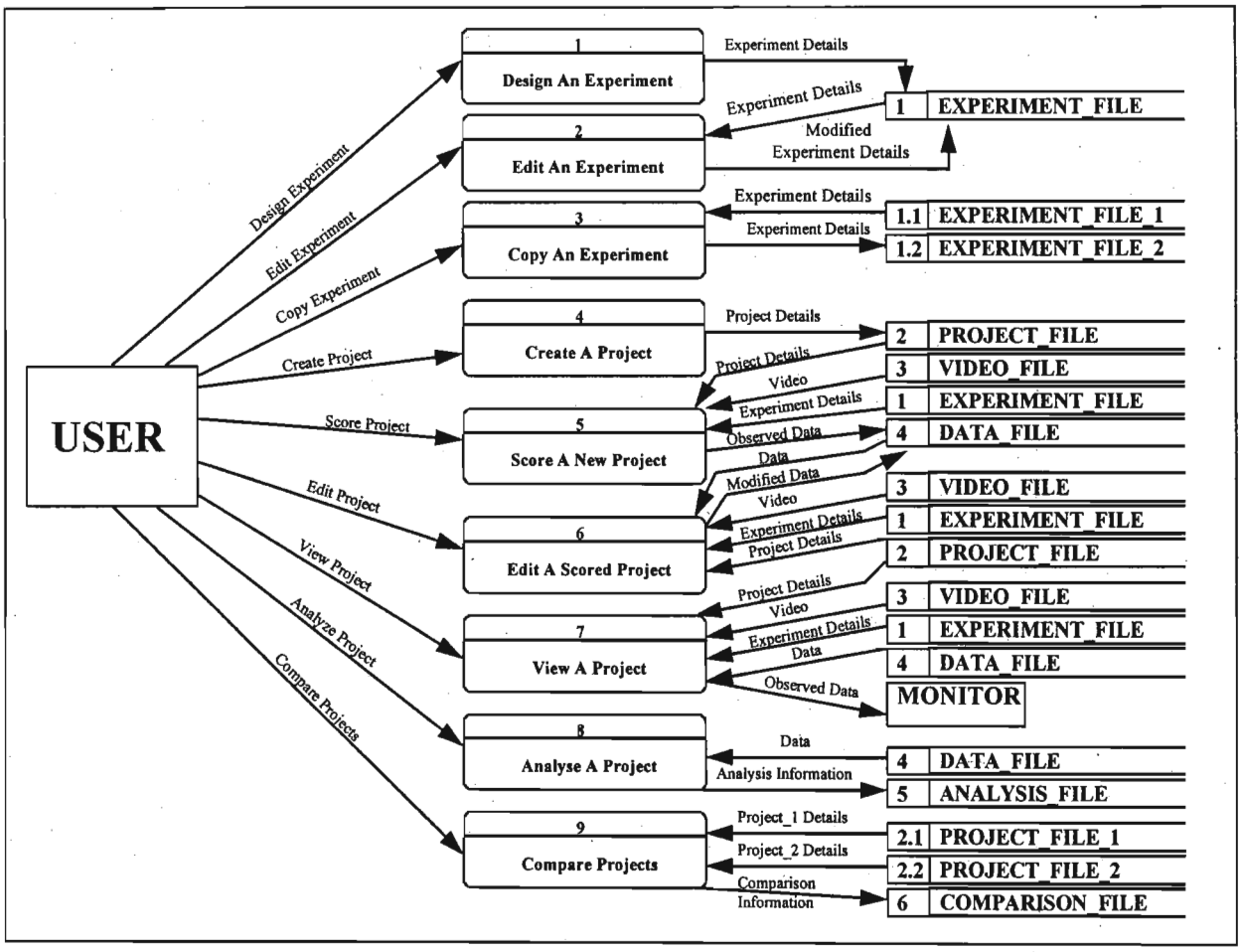

Figure 3.1 : Overview of VAS

3.2.2.2 EXPERIMENTS

Design An Experiment

The processes involved in designing an experiment are :

- **Specify Filename** : The experiment would be saved under the specified filename.
- **Describe Experiment** : Include a brief description of the experiment.

- **Specify Target Behaviours** : Specify a list of behaviours to be monitored by the researcher (target behaviours).
- **Specify Observational Interval** : The frequency of recording the observed behaviours. Recording of observed behaviours could either occur at fixed time intervals, or at random.
- **Specify Length of Experiment**
- **Specify Number of Interacting Subjects**

These processes are illustrated in Figure 3.2.

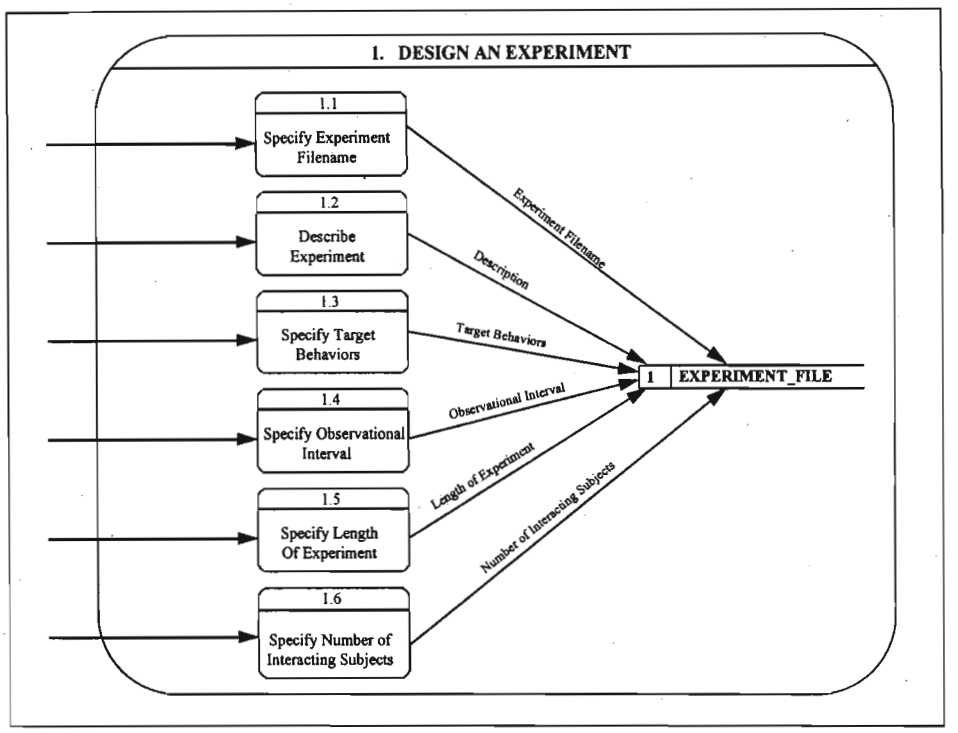

Figure 3.2 : Process 1 Exploded

Edit An Experiment

Figure 3.3 illustrates the processes associated with the modifying of an experiment. The Description, Target Behaviours, Observational Interval, Length of Experiment, and or the Number of Interacting Subjects may be modified, after the experiment to be modified has been opened.

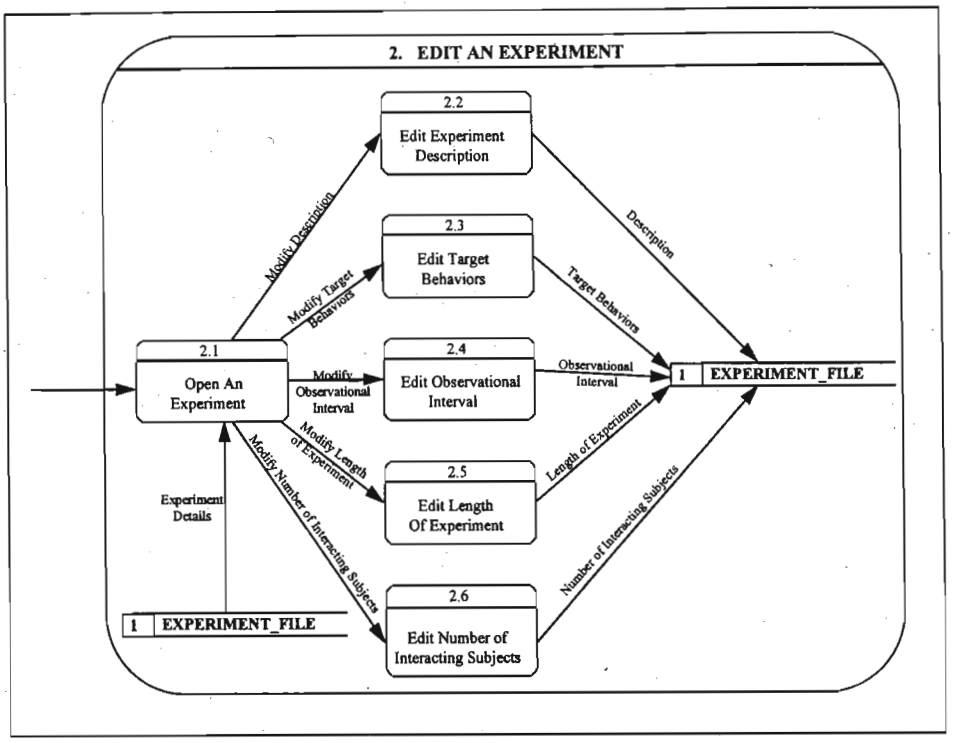

Figure 3.3 : Process 2 Exploded

Copy An Experiment

The processes associated with duplicating an experiment are :

- **Specify From_Experiment** : Specify the file from which to the experiment is to be copied.
- **Specify To_Experiment** : Specify the name of the file, to which the experiment is to be copied.
- **Copy Experiment** : Process responsible for the copying of the experiment.

These processes are depicted in Figure 3.4.

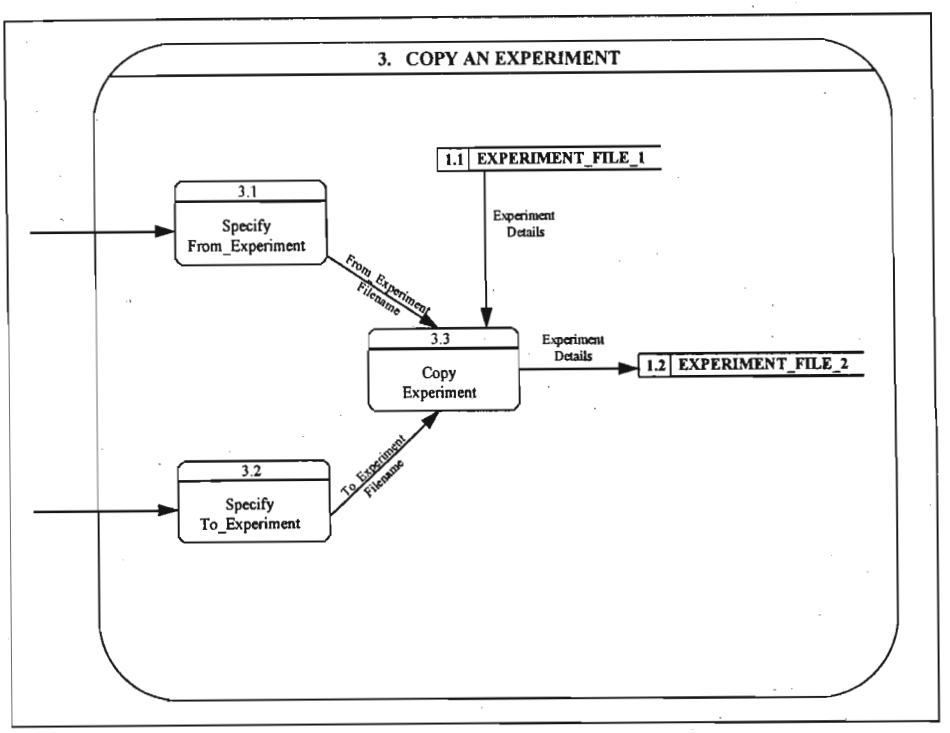

Figure 3.4 : Process 3 Exploded

3.2.2.3 PROJECTS

Create a Project

The processes involved in creating a project are depicted in Figure 3.5. These are :

- **Specify Project File** : The project (trial) would be saved under the specified filename.
- **Specify Experiment File** : Every project (trial) is based upon a previously defined experiment.
- **Specify Video File** : The name of the file containing the video that is to be analysed.
- **Data File** : The name of the file to which all observed information is to be saved.

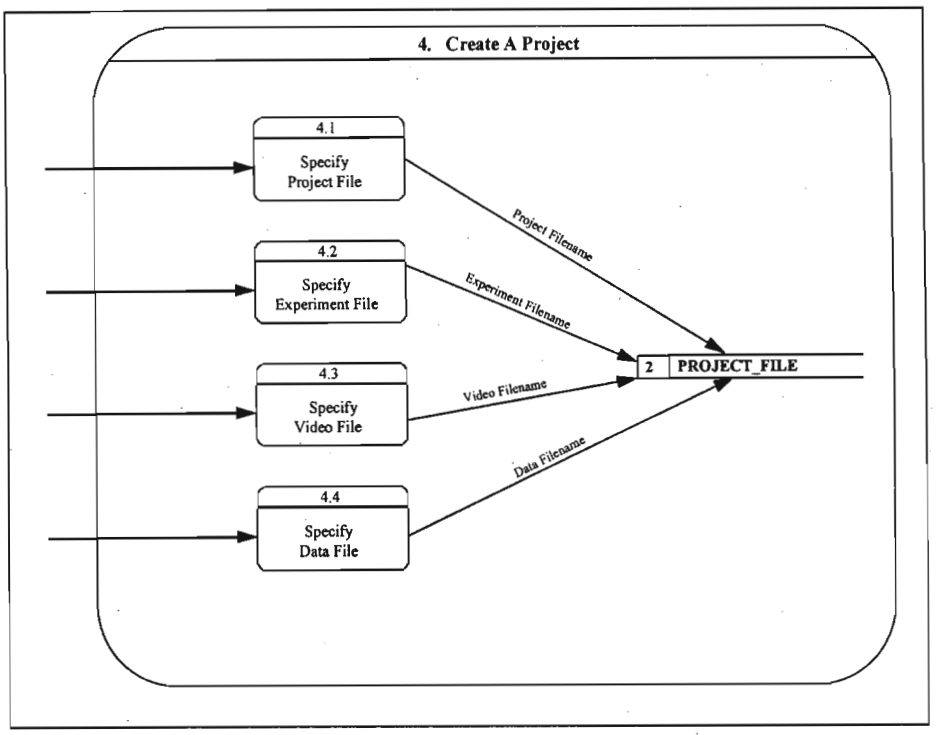

Figure 3.5 : Process 4 Exploded

Score a New Project

The processes that need to be performed when scoring a project are illustrated in Figure 3.6. These are :

- **Inserting Annotations** : Insert a list of target behaviours that have been observed at a particular point in the video.
- **Inserting Comments** : Insert additional information at a particular point in the video.
- **Time Events** : The duration of certain events may be timed.
- **Control Video** : Control the playback of the video e.g. Stop, Step forward a frame, play, Goto the beginning of the video etc.
- **Set Scale** : Scale to be used when measuring distances, calculating speeds and areas, may be set.
- **Calculate Length**
- **Calculate Speed**
- **Calculate Area**

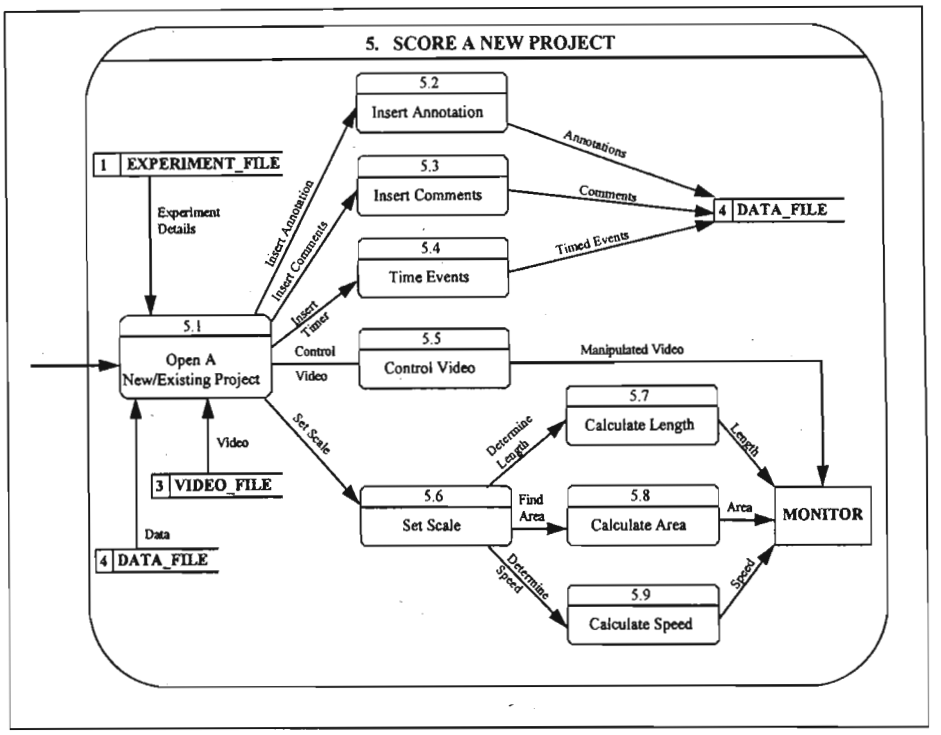

Figure 3.6 : Process 5 Exploded

Edit a Scored Project

Figure 3.7 illustrates the processes associated with the modifying of an experiment. These included processes to modify annotations and comments, insert additional timers, manipulate the video, and redefine the scale to be used in measurements. All subsequent calculations of area, speed and length would be based on the new scale.

View A Project

The processes associated with viewing a project, as illustrate by Figure 3.8, are :

- **Open a Project** : The project to be viewed must first be opened. Ideally this project would have previously been scored.
- **View a project** : A project displaying the video with all observed information.

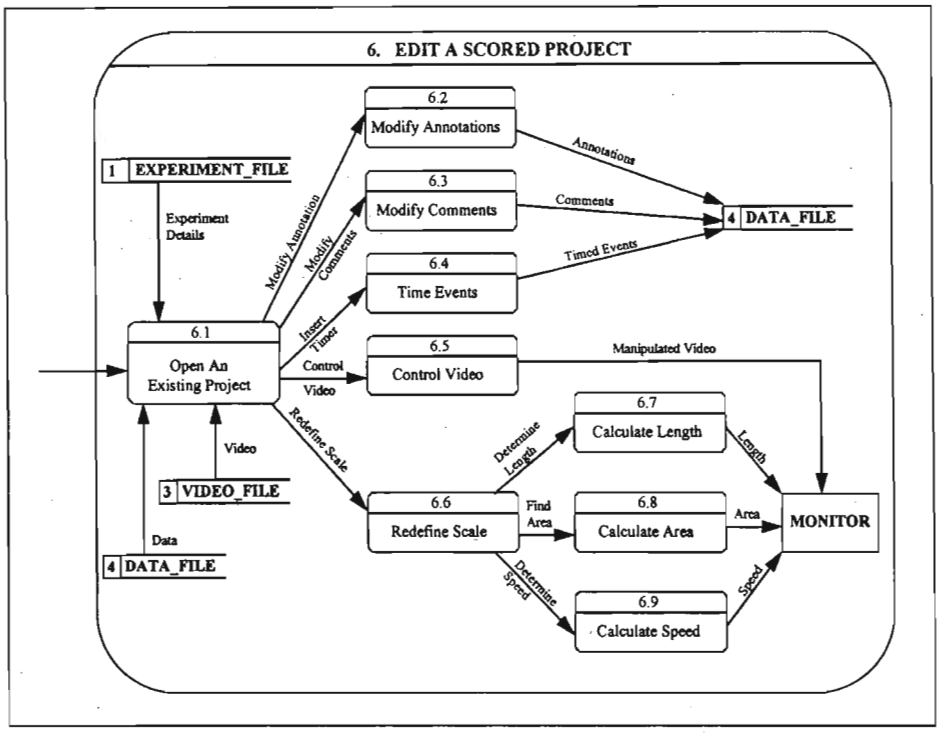

Figure 3.7 : Process 6 Exploded

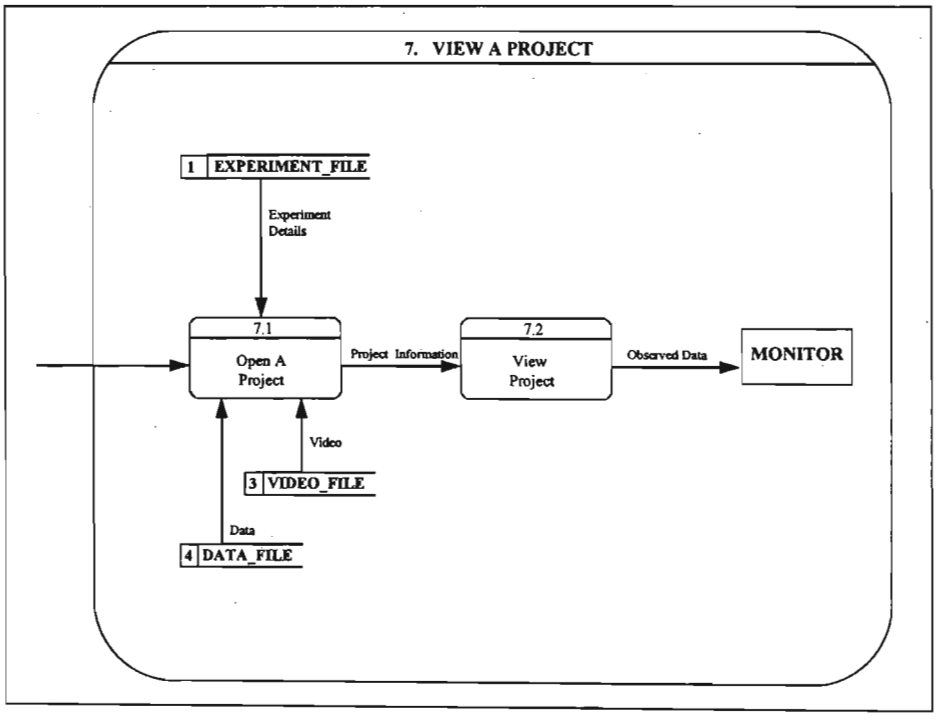

Figure 3.8 : Process 7 Exploded

Analyse A Project

The processes associated with analysing a project (shown in Figure 3.9) are :

- **Open a Project** : Ideally the project opened would have been previously scored.
- **Perform a Sequencing Analysis** : A calculation of the frequency of all (preceding behaviour: succeeding behaviour) pairs where the preceding and succeeding behaviours are elements of the set of target behaviours.
- **Perform a Frequency Analysis** : A calculation of the total number of times each target behaviour was observed.

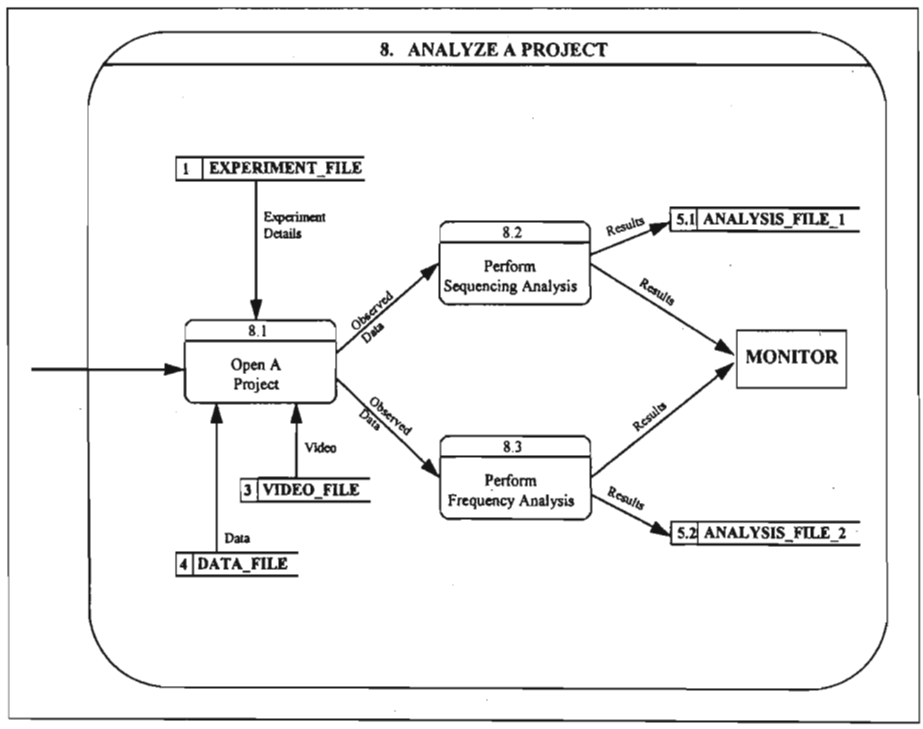

Figure 3.9 : Process 8 Exploded

Compare Projects

The processes that need to be performed during the comparison of projects are :

- **Open Projects**
- **Initial Processing** : Information from both projects is initially processed to ascertain whether or not a comparison can be made.
- **Perform Comparison** : The actual comparison of the projects is conducted.

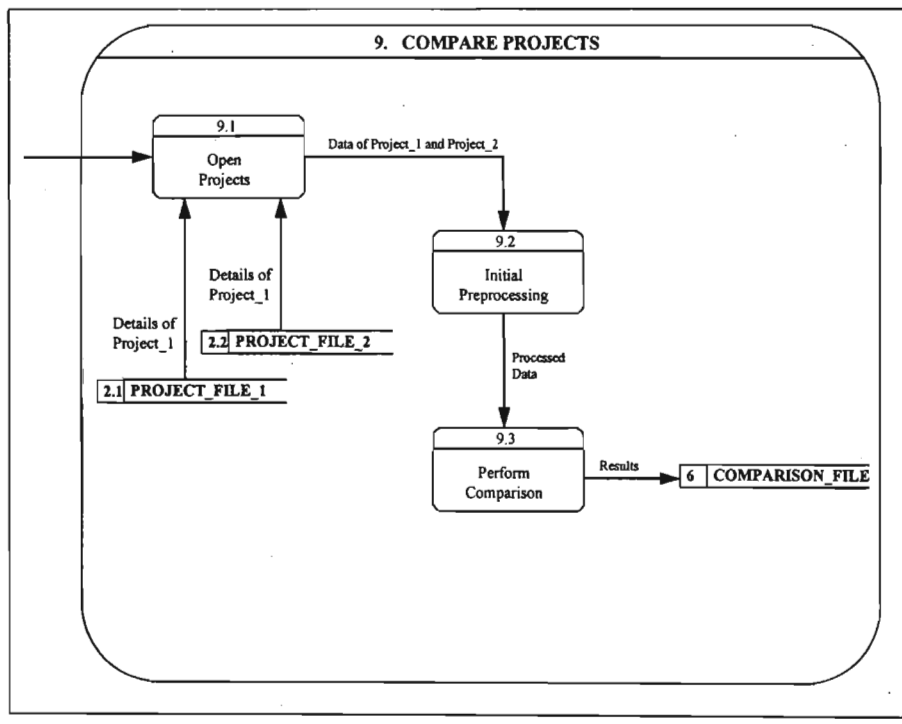

Figure 3.10 : Process 9 Exploded

3.3 CHAPTER SUMMARY

The interviews with researchers, using video tape recording as an observational tool played a key role in determining the scope of the intended system.

The purpose of the Video Analysis System (VAS) is to :

- facilitate the extraction of information from observations on video.
- assist with the analysis of the observed information.

The main processes that need to be performed by VAS are :

- Design an Experiment
- Edit an Experiment
- Copy an Experiment
- Create a Project
- Score a New Project
- Edit a Scored Project
- View a Project
- Analyse a Project
- Compare Projects

4 SYSTEM IMPLEMENTATION

4.1 OVERVIEW

VAS can be divided into three major modules, namely, the set-up module, the functional module, and the analysis module, illustrated in Figure 4.1. All these modules will be expanded on later in the chapter.

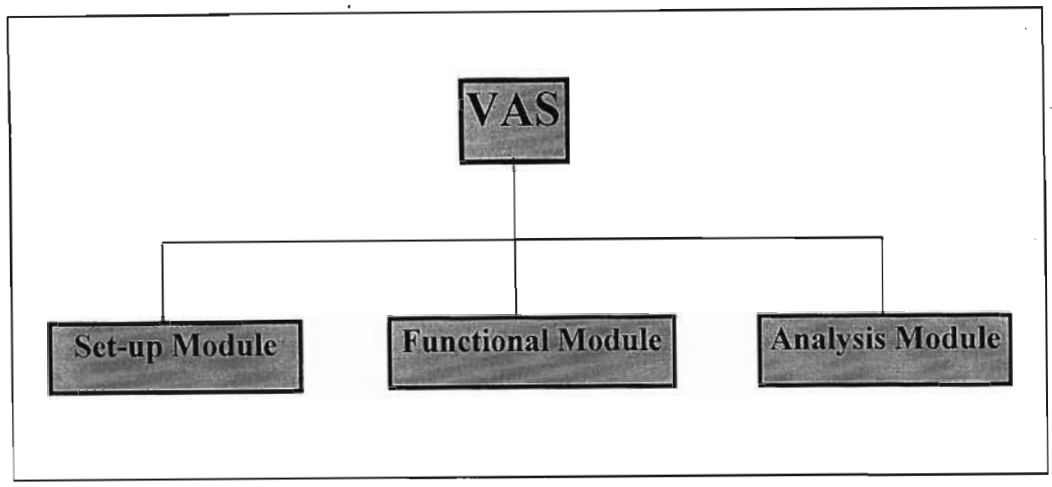

Figure 4.1 : Overview of VAS

4.2 ENVIRONMENT AND LANGUAGE CONSIDERATIONS

4.2.1 ENVIRONMENT

The author decided on a MS-Windows based system for the following reasons :

- The nature of the problem required an environment conducive to displaying several windows simultaneously.
- MS-Windows was favoured over other environments, for example OS/2, owing to its availability to the author and intended users, alike. Note, the intended users of the

system are behavioural researchers primarily in an academic environment. Most academic institutions in the country have MS-Windows based systems.

- Windows provides a consistent, easy to use graphical user interface.
- The most common environment for multimedia delivery using the PC, is Microsoft Windows 3.1 or higher. [BOTT95]
- MS-Windows provides control over numerous media devices through the inclusion of the Media Control Interface (MCI) drivers. [BOTT95]

4.2.2. LANGUAGE

When developing a multimedia application under windows, the choice of programming language dictates the level of abstraction afforded to the programmer.

The author chose Borland C++ version 4.5 as the implementation language for the following reasons :

- With C++, the programmer has total control, and hence can control how windows interfaces with the multimedia drivers. [WODA94]
- C++ provides a fairly rich set of data types and data structures. These include arrays and structures.
- The three basic concepts that underlie object oriented programming are data abstraction, inheritance, and polymorphism. C++ includes features that support each of these concepts. Thus new classes could easily be derived from existing ones.
- Borland C++ includes the ObjectWindows classes, which facilitates the quick and easy development of windows applications.

Borland C++ was chosen over Microsoft Visual C++, as the most recent version of Borland C++, namely version 4.5, was available at the time.

C++ was favoured over other software development platforms such as :

- **Delphi** : because at the time, Delphi was a fairly new windows programming environment, therefore it was not totally clear how much of flexibility it would afford the programmer. Also, there was not a lot of documentation available on Delphi.
- **Visual Basic** : because the multimedia support of Visual Basic lacks the callback capability. [WODA94]
- **Toolbook** : because Toolbook does not provide adequate access to the Windows API. [WODA94]

4.3 IMPLEMENTATION CONSIDERATIONS

4.3.1 THE APPLICATION

The application was developed using the ObjectWindows classes. The multiple document interface was implemented by deriving classes from the ObjectWindows `TMDI` classes. This implementation permitted the user to simultaneously work with more than one project. However, it also permitted the user to simultaneously edit multiple instances of a particular project, and thus have multiple inconsistent versions of the project displayed on the screen. In an effort to eliminate the occurrence of such an inconsistency the user was restricted to working with only one project at any given time. This approach maintained simplicity and consistency.

An alternative method would be to permit the user to simultaneously work with several projects, but prompt the user to rename a project and its data file component, if that project is already in use.

4.3.2 SET-UP MODULE

This module defines the fundamental operational element, the **project**.

Projects are defined in terms of **experiments**, and hence represent an instance of an experiment (trial). An **experiment** is a set of specifications that defines a particular investigation. The concept of an experiment was particularly appealing for it avoided duplication of information in that, several instances of an experiment (i.e. **projects**) could be defined using just one specification of an experiment.

4.3.2.1 EXPERIMENTS

This provided the means for users to supply the initial set-up information. That being, a brief description of the experiment, number of interacting subjects, a list of behaviours that would be monitored by the user (i.e. target behaviours), the observational interval (i.e. a specification of how frequently the observed target behaviours would be recorded) and the duration of the experiment.

DATA STRUCTURES USED

An array of pointers to characters was used to store the target behaviours. The upper limit of the array was fifty, hence implying at most fifty target behaviours could be specified. At the time, this was taken to be a reasonably large enough value, however, with hindsight it would have been more feasible if there were no such restriction. Hence a linked list would have been a more appropriate data structure.

The description was stored in a string, and the length of experiment, number of interacting subjects and observational interval were saved to variables of type integer. The number of interacting subjects were encoded as 1, 2, or 3, depending on whether there was one, two or a group of interacting subjects.

All information pertaining to an experiment was saved to a text file with a .EXP extension.

4.3.2.2 PROJECTS

A project maintains a record of :

- the experiment upon which the project was based (a text file with a .EXP extension)
- the video to be scored/analysed (a .AVI file)
- the file to which the data generated for the project was to be saved. (a text file with a .DAT extension)

All information pertaining to the project was saved to a text file with a .PJT extension.

4.3.3 THE FUNCTIONAL MODULE

This module provides the facilities needed to perform the primary task of VAS, that being the extraction of observational information from a video. In addition, it also provides a means for viewing the extracted observational information. The essential features of this module include :

- Video Manipulation
- Annotations
- Comments
- Timers
- Calculations

4.3.3.1 VIDEO MANIPULATION

The ability to view and manipulate a video is a key feature of VAS, as would be the case in any other system using video.

THE MEDIA CONTROL INTERFACE

The author chose to use the Media Control Interface (MCI) driver, MCIAVI, to play the video for the following reasons :

- From the perspective of the programmer, the MCI provides a command string interface, which is easy to use.
- When invoking a device operation with MCI, the application may request a notification event be generated upon completion of that operation.
- MCI performs certain system operations such as `sysinfo`, which given a `device type`, allows applications to determine the number of instances of that type, their names and which of those instances are currently open.
- The windows environment provides video cassette recorder (VCR) services through a device driver that is based on the MCI command set for VCRs. At the moment the system accommodates video in the form of .AVI files stored on disk. If the scope of the system where to be extended in order to accommodate video tapes, it could easily be done using the MCI drivers.
- The availability of the MCI drivers and documentation, also served to make it a more favourable choice.

VIEWING WINDOW

This window was used to display the video, video controls and the video runtime clock. This window was first displayed upon the opening of a project and remained visible until the project was closed. This was considered appropriate as the video was considered to be the primary focus in almost all tasks that were to be performed.

The viewing window was derived from the ObjectWindows `TDialog` class. The size of this window was determined by the size of the video as stored in the .AVI file. In order to ensure that the resolution was at its optimum it was decided to use the default window size and not allow the window to be resized.

The video runtime clock was implemented using a Windows timer. When a Windows timer is started it sends `WM_TIMER` messages to the window that owns it (the viewing window) at whatever interval was specified. The `EvTimer()` function then handles this message. The video runtime clock was controlled by overriding this function. Each time the `EvTimer()` function was called, the current time of the video was obtained and used to update the runtime clock on the screen.

The major drawback of this approach is that the `WM_TIMER` messages can be late or even skipped, as the `WM_TIMER` message has the lowest priority of any windows message. It will only be returned by GetMessage if there are no other messages in the message queue. [RICH95]

This did not, however prove to be a major problem with VAS, as it was manifested rather infrequently by virtue of the fact, that the **display** of the runtime clock was not updated and hence went off by a second at most. What this really implied, was that the system did not receive a `WM_TIMER` message and hence did not update the video runtime clock with the current time from the video. It must be stressed that even though the display of the runtime clock went off, the actual synchronisation of the video remained in tact, as this information was stored in the .AVI file.

However, for areas where accurate timing is an absolute necessity, and such discrepancies prove to be a major problem, one of several software timers with millisecond resolution could be used to generate a timer event. (see [PERO85], [COYO87], [GRAV88]).

The video controls (play, pause, etc.) were implemented by sending the appropriate commands to the MCI driver, using the command string interface. A feature of all the video controls was that they maintained the synchronisation of all comments and annotations previously made. For example if the user chose to go to the beginning of the video clip, in addition to moving to the beginning of the clip, the `GOTO START` control, would also ensure that any annotation or comment previously made at the start of the clip, would be displayed.

Synchronisation of annotations and comments were also maintained during the playback of the video. This was achieved by automatically pausing the video, and thus the video runtime clock, at all points at which annotations and comments were previously made.

4.3.3.2 ANNOTATIONS

Annotations are encoded textual recordings of observed target behaviours, using previously defined descriptors of target behaviours.

DATA STRUCTURE USED

An implementation of a linked list was used to store annotations. Each node in the list consisted of a field for the time (taken from the video runtime clock) as well as a field for the annotation itself. This data structure was particularly suitable because the number of annotations and the size of each annotation in any project could not be predetermined. It also readily accommodated the subsequent editing of an annotation, which involved the insertion and or deletion of annotational elements, i.e. target behaviours.

REPRESENTATION OF TARGET BEHAVIOURS

The notation used for the recording of observed target behaviours varies depending on the number of interacting subjects present, which may be one, two or a group.

One Subject

Target behaviours are encoded by simply noting the behaviour, and separating consecutive behaviours by use of a semi-colon. Optional white space between the separator (semi-colon) and target behaviours was permitted. This notation ensured that the annotation could be easily read by the user.

Example :

Target Behaviours = walk, sit, talk, laugh

Number of interacting subjects = 1

Typical Annotation for time interval 0-3 : walk; sit ; laugh;talk

Explanation : The subject displayed the target behaviour *walk*, then the behaviour *sit*, followed by the behaviour *laugh*, and finally *talk*.

Two Subjects

In a scenario where there are two interacting subjects under observation, the user may choose, if he/she so desires, to make annotations about each subject separately, and or about both subjects jointly. This permits the user to record non-interacting behaviours, as well as interacting behaviours. For example, if there are 2 subjects under observation and they are both sleeping, the user would want to note these behaviours for each subject, separately, as subject1 is not affecting, nor is affected by, subject2. The same holds true for subject2. However if subject1 was attacking subject2 and subject2 was defending this attack, then the user would want to note these interacting behaviours as such.

Even though, at any particular time, the user may note annotations of subject1 alone, subject2 alone, and or subject1 and subject2, all these annotations cannot be noted simultaneously, as only one window for noting annotations is present. At any particular time, the user could toggle among the different annotations, for the different subjects by changing the subject under observation to `Subject1`, `Subject2`, or `Sub1 and Sub2`.

Annotations that are made for each subject separately, are made in exactly the same manner as for the scenario where there is only one subject (described above). However when noting annotations for subject1 and subject2 jointly, (i.e. interacting behaviours), the user must note the displayed behaviour and in addition, indicate the subject that displayed the behaviour. This is done by preceding each displayed behaviour by a subject indicator, i.e.. a '(1)' if subject 1 displayed the behaviour or a '(2)' if subject2 displayed the behaviour.

This notation was used because it kept the annotations simple and readable. The use of subject indicators were favoured over the use of identifiers or names because subject indicators would more blatantly indicate the different subjects.

Example

Target Behaviours = sleep, sit, walk, attack, defend, shuffle

Number of Interacting Subjects = 2

Typical annotation for subject1 alone at time interval 0-4 : sleep, sit

Explanation : Subject1 displayed the behaviour *sleep*, followed by the behaviour *sit*

Typical annotation for subject2 alone at time interval 0-4 : sleep

Explanation : Subject2 displayed the target behaviour *sleep*.

Typical annotation for subject1 and subject2 jointly, at time interval 4-8

(1) walk; (2) sit ; (1) attack ; (2) defend;

Explanation : Subject1 displayed the behaviour *walk*, subject2 then displayed the behaviour *sit*, followed by subject1 displaying the behaviour *attack*, and finally subject2 displaying the behaviour *defend*.

With the scenario where there are two interacting subjects being observed, the user could effectively have, up to three different annotations for any given time. In the previous

example the user made annotations for subject1 as well as, subject2, during the time interval 0-4 seconds.

As annotations were saved to a linked list, in which nodes were uniquely identified by their time field, a hidden coding scheme had to be devised to accommodate this special case, which may include more than one annotation for any given time. This was done using special identifier strings, "$$$" and "###". Annotations of subject2 alone were preceded by "$$$", and annotations of subject1 and subject2 jointly were preceded by "###". Having prepended the appropriate identifier string, annotations for the specific time were concatenated, and saved to a single node in the list.

A Group

In the scenario where there was a group of interacting subjects under observation, the user was permitted to make annotations about the group as a whole. For example when observing group activities of a troop of monkeys.

Example

Target Behaviours = play, fight, run

Number of interacting subjects = A Group

Typical Annotation at time interval 0-9 : play ; fight ; play ; run;

Explanation : The subjects within the group, displayed the behaviour *play*, followed by the behaviour *fight*, then the behaviour *play*, and finally the subjects displayed the behaviour *run*.

A shortcoming of this option is that it limits the user to recording only group activities. A method of overcoming this limitation, would be to extend the scope of this option, to include annotations about individual subjects within the group, as well as annotations

indicating interactions among subjects within the group. The former could be implemented through the use of subject indicators, and the latter through the use of subject indicators and modifiers. Modifiers would indicate the subject(s) that was/were affected by the displayed behaviour.

NESTED BEHAVIOURS

Nested behaviours refer to the concept where one behaviour induces another, and the latter lasts for the duration of the former.

Nested behaviours were represented by the use of the open and close square brackets (i.e.. '[', ']').

Example 1

Target behaviours = growl, roam, sit, attack

Number of interacting subjects = 1

Typical annotation : growl[roam];

Example : This indicates that the behaviour *growl* induced the behaviour *roam*, and this roaming behaviour lasted for the duration of the growling behaviour.

Example 2

Target behaviours = growl, roam, sit, attack

Number of interacting subjects = 2

Annotation for subject1 and subject2 jointly : (2) growl[(2) roam]; (1) sit;

Explanation : This indicates that the behaviour *growl* of subject2 induced the behaviour *roam* of subject2, and this roaming behaviour lasted for the duration of the growling behaviour. Subject1 then displayed the behaviour *sit*.

The use of square brackets was particularly appropriate in this instance, because they intuitively imply that the outer behaviour encapsulates the inner one, which in a sense hints at nested behaviours.

SIMULTANEOUS BEHAVIOURS

Simultaneous behaviours refer to those behaviours occurring at the same time. Simultaneous behaviours are indicated by the use of a simultaneous behaviour index (i.e. a number) that precedes each target behaviour in the set of simultaneous behaviour. This notation was chosen because it resulted in effective and simple annotations.

Lets consider an example :

Example

Target Behaviours = talk, wave, smile, walk, sit

Number of interacting subjects = 2

Typical annotations for subject1 alone at interval 0-9 :

1talk; 1sit; walk; 2wave[2smile]; 2sit; walk;

Explanation : Subject1 displayed the behaviour *talk* whilst simultaneously displaying the behaviour *sit*. Thereafter, the behaviour *walk* was displayed followed by the simultaneous behaviours *wave[smile]* and *sit*. This implies that the subject displayed the nested behaviours *wave[smile]* and simultaneously displayed the behaviour *sit*. Thereafter the subject displayed the behaviour *walk*.

Note, within each new annotation, the simultaneous behaviour index, starts off being the lowest possible value (i.e. 1). Thereafter for each successive set of simultaneous behaviours, within this annotation, the index is incremented by one.

Typical annotations for subject1 and subject2 jointly :

(1)1talk; (2)1sit; (1)walk; (2)2wave[(2)2smile];(1)2sit; (2)3walk; (2)3sit;

Explanation : Subject1 displayed the behaviour *talk* whilst subject2 simultaneously displayed the behaviour *sit*. Thereafter, subject1 displayed the behaviour *walk*. This was followed by subject2 displaying the nested behaviour *wave[smile]* while subject1 simultaneously displayed the behaviour *sit*. Subject2 then displayed the behaviour *walk*, whilst simultaneously displaying the behaviour *sit*. Note, this is an impossible set of simultaneous behaviours, i.e.. a subject cannot simultaneously walk and sit. VAS does not perform any validation to ensure that annotations entered do in fact make sense. The onus is upon the user to ensure that annotations entered do make sense.
Note also, the simultaneous behaviour index starts off being 1. Each annotation has its own instance of the simultaneous behaviour index.

ANNOTATION WINDOW

This window, derived from the ObjectWindows TDialog class, is used for the recording and display of annotations. A feature of this window is that it maintains the synchronisation of the displayed annotations with the video runtime clock. For example, if the user chose to view the annotation made at time 3, (by selecting this time in the appropriate combo box in the annotation window), the video in the viewing window will also be affected. The video will automatically be stepped forward or backward so as to ensure that the video runtime clock is synchronous with the annotations displayed in this window.

4.3.3.3 COMMENTS

Comments refer to any additional information that the user may want to include. Unlike annotations, comments need not necessarily be in terms of the observed target behaviours, and hence afford the user greater flexibility when noting information.
The data structure used for the implementation of comments was the same as that, used for annotations, for the reasons already specified.

COMMENT WINDOW

This window was used for the recording and display of comments. The synchronisation of this window with the video runtime clock was maintained at all times, in a manner similar to that used for the synchronisation of the annotation window.

4.3.3.4 TIMERS

Timers allow the user to time certain displayed behaviours or even a sequence of displayed behaviours.

The user was permitted to activate as many timers as required, at any point in the video. Each timer was uniquely identified by its name, which had to be given by the user. A linked list was used to store all activated timers, since the number of timers that would be used could not be predetermined. Each node in the list consisted of a field for the name of the timer (a pointer to a character), a field indicating the **frame** at which the timer was started (a pointer to a character) and a field indicating the **time** at which the timer was started (a pointer to a character).

As soon as a timer was stopped, it was removed from the list of activated timers and placed in a list of timed events. This was done to enable the user to display the information of a timed event, at any later point. Timed events were also stored in a linked list. Each node

in the list consisted of a field for the name of the timer, the start frame, the start time, the stop frame, the stop time, and the duration of the timed event.

All information pertaining to timed events, as well as all comments and annotations were saved to the data file of the project.

4.3.3.5 MEASUREMENT TOOLS

The user was permitted to perform the following calculations :

- length
- speed
- area

Before performing any calculations the user would need to set up a scale that would be used in all subsequent calculations. This would be done by specifying two points on the video as well as, the actual distance between these points.

A feature of this, and all calculation tools in VAS is that, as a sequence of points is being specified, (by right clicking on the video displayed in the viewing window), lines connecting these points are immediately drawn. This enables the user to keep a track of the path or object, defined by the sequence of points. If the video was taped from a static location (i.e. the video camera was not moved during the recording), and not zoomed in or out, then the user may specify points from different video frames. However if the either of the previous conditions is not met, then all points have to be specified in a single frame. In addition the scale has to be redefined, each time the camera is moved or the image is zoomed in or out.

FORMULAE

Lengths were calculated using the following formula :

$$\text{length} = ((x_1-x_2)^2 + (y_1-y_2)^2)^{½}$$

where y_i and x_i refer to the (x,y) coordinates of point i.

Speeds were calculated using the formula :

$$\text{Speed} = \sum_{i=1}^{n-1} ((x_i-x_{i+1})^2 + (y_i-y_{i+1})^2)^{½} \ / \text{time\_taken}$$

where y_i and x_i refer to the (x,y) coordinates of point i, and time_taken is specified by the user.

Areas were calculated using the formula :

$$\text{Area} = \sum_{i=1}^{n} (x_{i+1} * y_i - x_i * y_{i+1}) \ / 2 \ + \ (x_1 * y_n - x_n * y_1) \ / 2$$

where y_i and x_i refer to the (x,y) coordinates of point i.

4.4 ANALYSIS MODULE

This module permits the user to perform the following analyses :

- frequency analysis
- sequencing analysis
- comparison of projects

4.4.1 FREQUENCY ANALYSIS

The user is permitted to perform a calculation to determine the number of times each target behaviour was recorded.

The results of this analysis was saved to a text file and also displayed on screen using a dialog box derived from the ObjectWindows TDialog class.

4.4.2 SEQUENCING ANALYSIS

Sequencing analysis is a frequency count of all (preceding behaviour:succeeding behaviour) pairs, where the preceding behaviour, and succeeding behaviour are elements of the set of target behaviours.

The results of this analysis was saved to a text file.

4.4.3 COMPARISON OF PROJECTS

A comparison of projects, or more specifically a comparison of annotations, may be conducted on two projects that are based on the same experiment. This is done by firstly

determining the number of annotations that were made at the same time, in both projects. And thereafter performing a detailed examination of these annotations, in order to ascertain the number of target behaviours that coincide in each of them.

4.5 CHAPTER SUMMARY

A windows-based video analysis system, called VAS has been developed. VAS has been intended primarily for the use of behavioural researchers.

VAS begins by permitting the user to design an experiment, and thereafter projects. A project consists of an experiment file component, a video file component as well as a data file component.

Having defined a project, the user was able to extract the relevant observational information from the video by recording annotations, comments, timed events and or measurements.

Having completed the process of information extraction, the user may choose to either view, or edit this information, or alternatively perform certain analyses.

5. DISCUSSION AND FUTURE RESEARCH

5.1 INTRODUCTION

This chapter will begin with a walk through of a case study. The case used is one typical of the work done by one of the interviewees. Thereafter VAS will be discussed, and the strengths and weaknesses of the system highlighted, and consequently areas for further research identified.

5.2 CASE STUDY

VAS was used to analyse observational data which was originally on videotape. The video illustrated the interaction of two subjects in an enclosed area.

A video to be scored and analysed with VAS has to be the form of a .AVI file. Hence the first step was to capture the video from videotape to a .AVI file. This was done using a video capture board and dedicated software. The RT300 VideoBlaster card together with the Adobe Premiere software was used.

5.2.1 USING VAS

5.2.1.1 SET-UP

The initial process consists of designing an experiment and thereafter a project.

This experiment included two interacting subjects. The observational interval specified was a fixed time interval of ten seconds. Thus observed behaviours would be saved by the system, not at the exact time of making the observations, but rather to the corresponding

interval, e.g. if a target behaviour was noted at time three, this behaviour would be saved to the time interval 0-10.

The following target behaviours were decided upon :

- shuffle
- kick
- cling
- attack
- defend
- threaten

Figure 5.1 and figure 5.2 shows the state of the **Design An Experiment** and **New Project** dialog boxes at the completion of this process. Figure 5.2 also reflects the files associated with this project, i.e. an instance of the experiment.

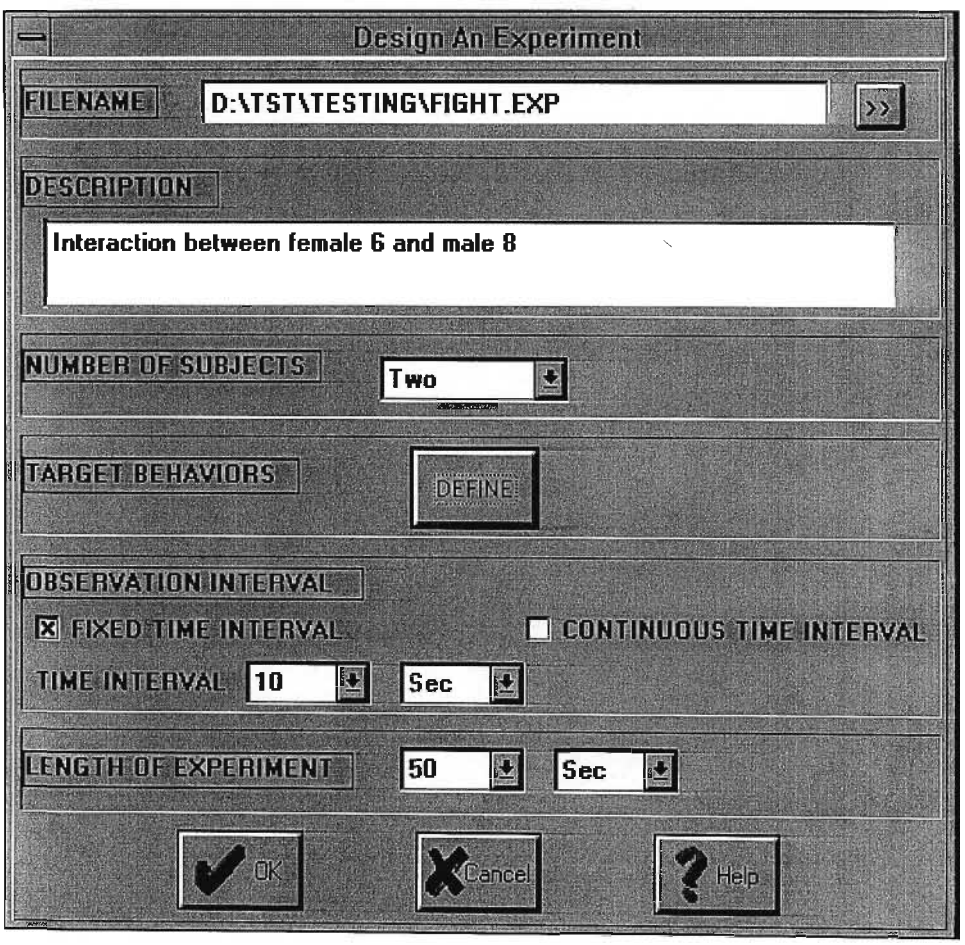

Figure 5.1 : Design an Experiment

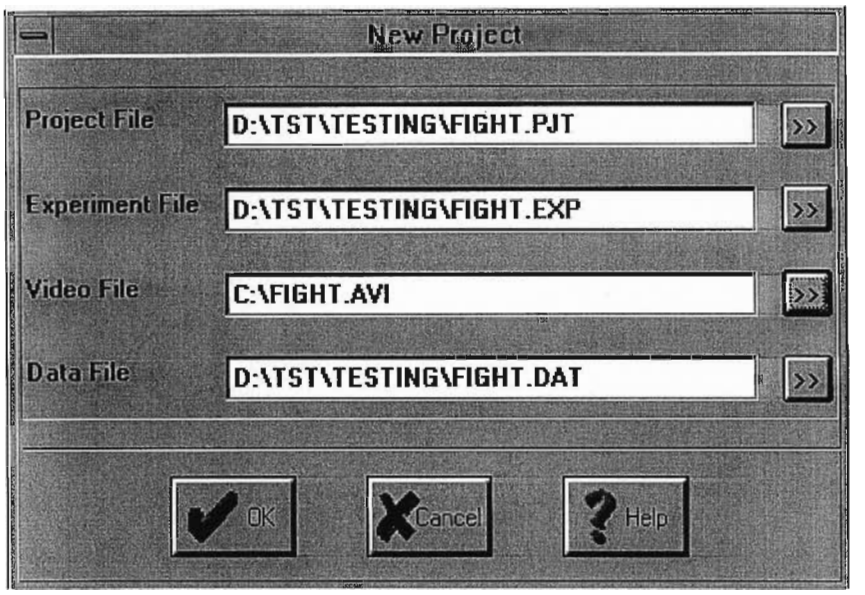

Figure 5.2 : Create a New Project

5.2.1.2 INFORMATION EXTRACTION

Having designed an experiment and set-up a project, the next step in the process is the extraction of observed information. This would entail the inclusion of annotations, comments, timed events and certain measurements.

Annotations

Annotations were made in the annotation window.

Referring to figure 5.3, Annotation at time indicated the start time of the current interval. For example, during the interval 0-10, the value in the corresponding combo box would be zero.

Since there were two interacting subjects the Subject under observation could be **Subject1**, **Subject2**, or **Sub1 & Sub2**. The focus of the case study was interacting behaviours, hence the annotations currently displayed are observations for **Sub1 & Sub2**.

The annotation window currently displays the recorded annotations, namely : **Subject1** displayed the nested behaviour *shuffle[kick],* whilst **Subject2** simultaneously displayed the behaviour *cling*. **Subject1** then displayed the behaviour *attack*, and **Subject2** then displayed the behaviour *defend.*

Comments

Also shown in figure 5.3 are the Comment Window, and the Viewing Window.

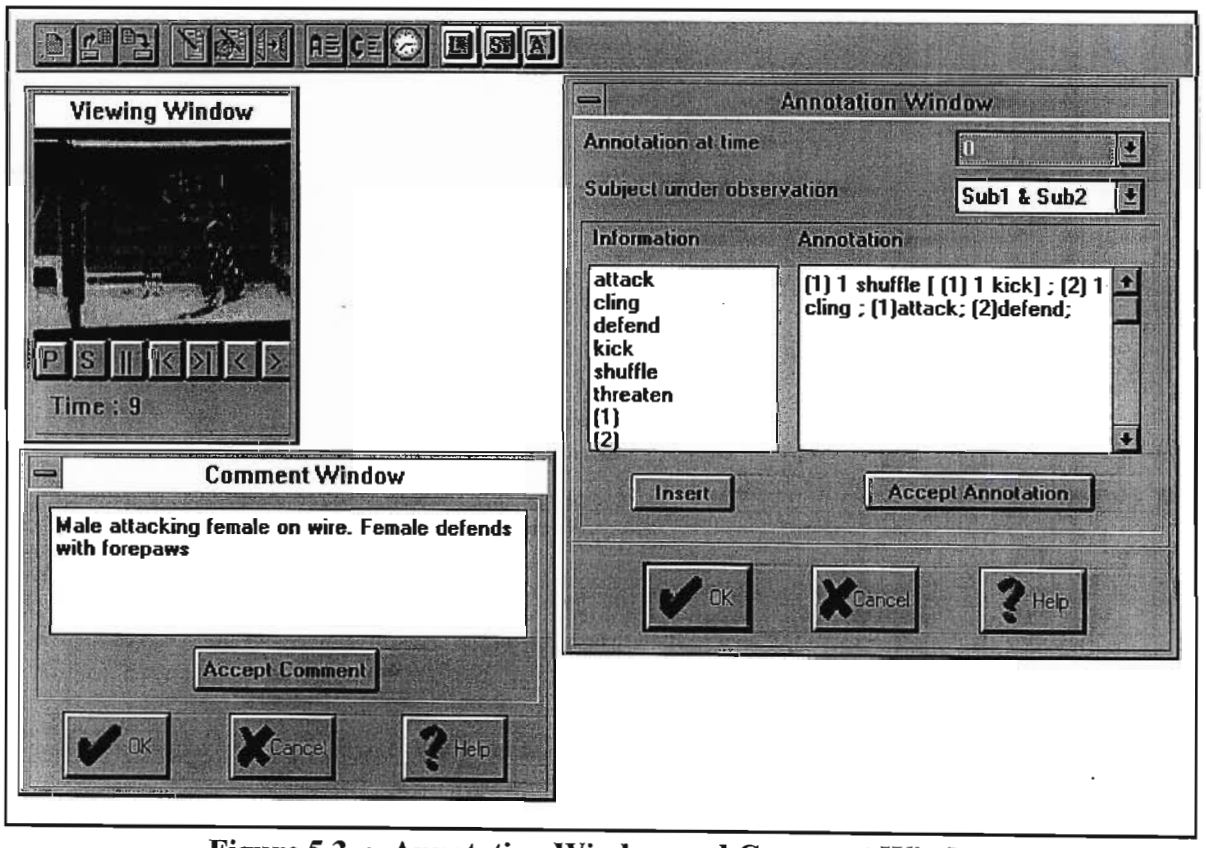

Figure 5.3 : Annotation Window and Comment Window

Timed Events

During the observation, certain events were timed, e.g. the duration of a fight between the two subjects. Information pertaining to these timed events (i.e. the start time stop time, and duration represented in milliseconds, as well as the start frame and stop frame) could be viewed in the **Timed Events** dialog box. Figure 5.4 currently reflects the duration of **FIGHT1** as 3200 milliseconds, and illustrates the use of a comment to provide additional information about the timed event **FIGHT1**

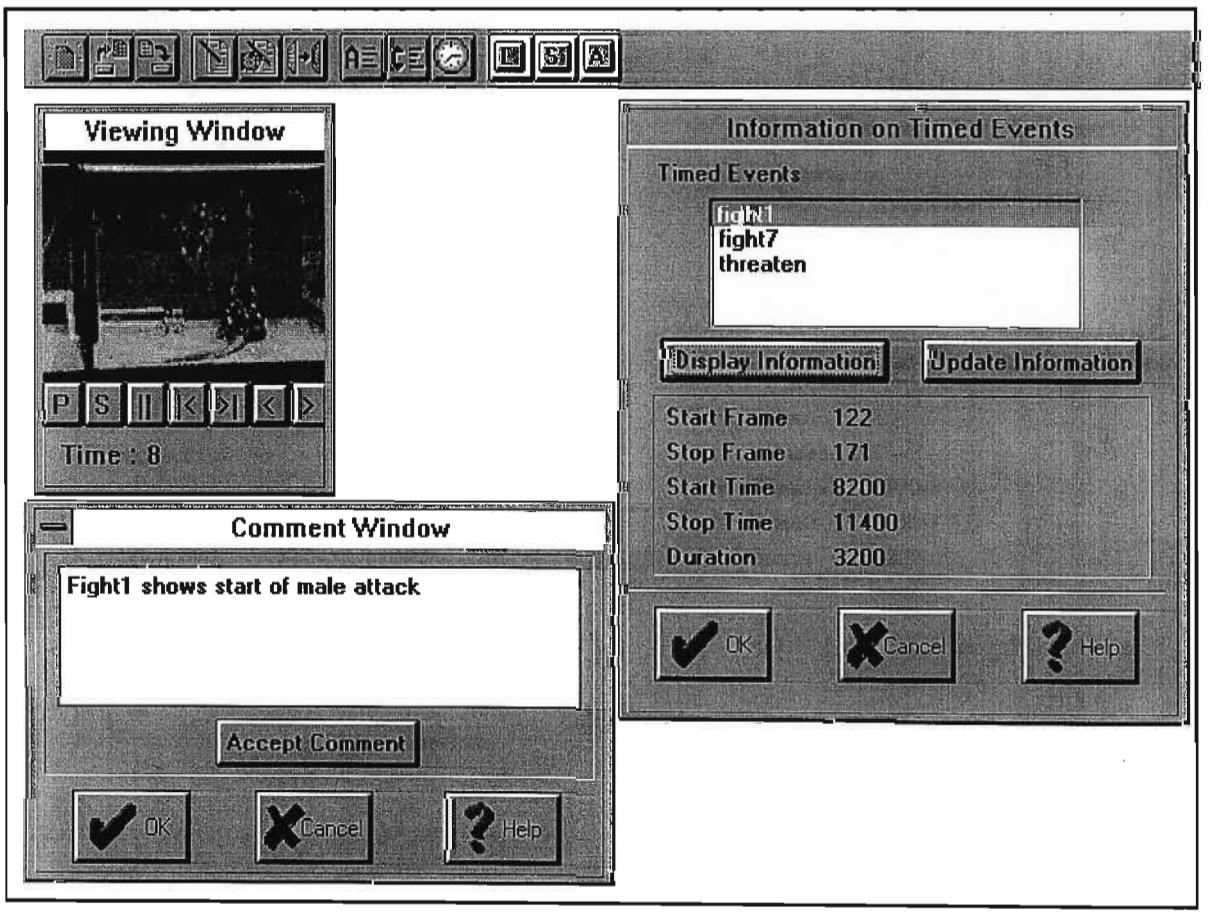

Figure 5.4 : Timed Event Dialog

At the completion of the information extraction process, all annotations, comments and timed events were stored to the data file component of the project. All files generated during the case study are included in Appendix 1.

Calculations

During the information extraction process, calculations of speed, length and area were conducted. The scale was set using the bottle which could be viewed in the video. This bottle was 35 centimetres in length. For each calculation a sequence of points was specified, which was then outlined on the video by lines connecting each of these points. The speed with which the male (subject on the floor) shuffled from one end of the box to the other was calculated. This is depicted in Figure 5.5. Figure 5.6 illustrates the distance from one end of the box to the other. Figure 5.7 illustrates a calculation of the area of the body of the male, when in a crouched position.

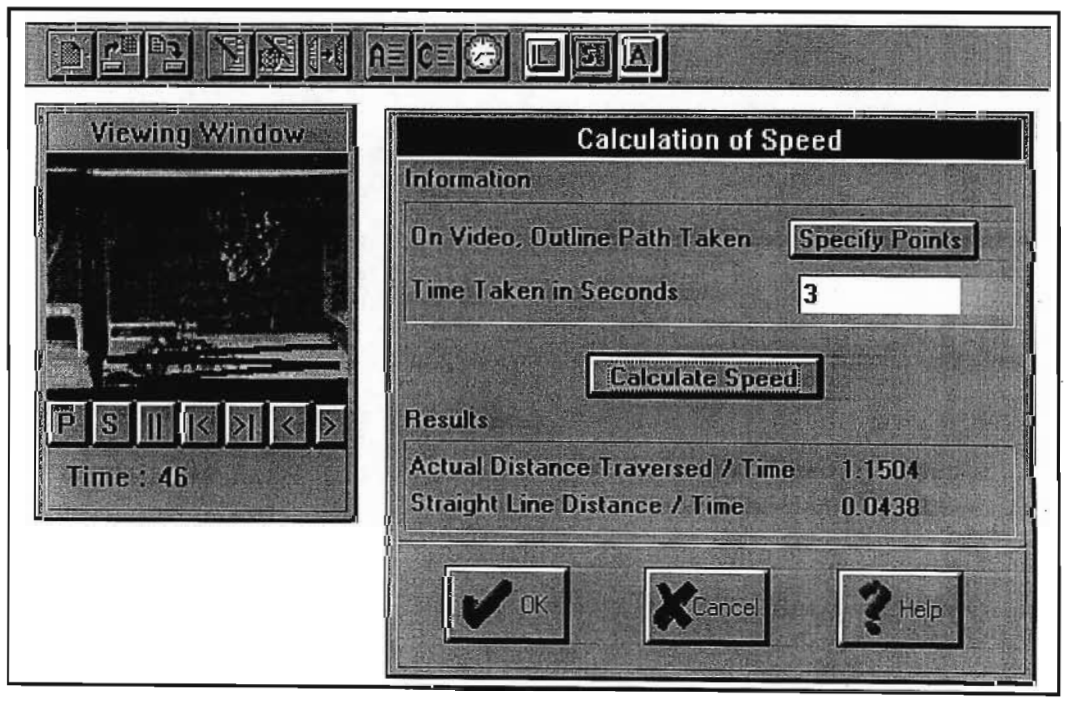

Figure 5.5 : Calculation of Speed

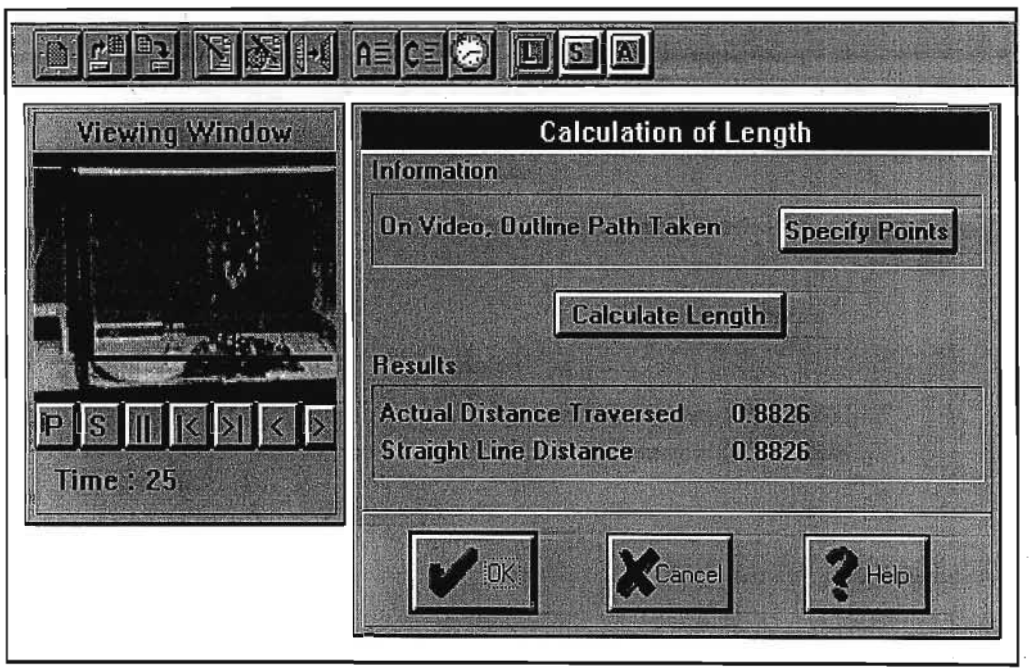

Figure 5.6 : Calculation of Length

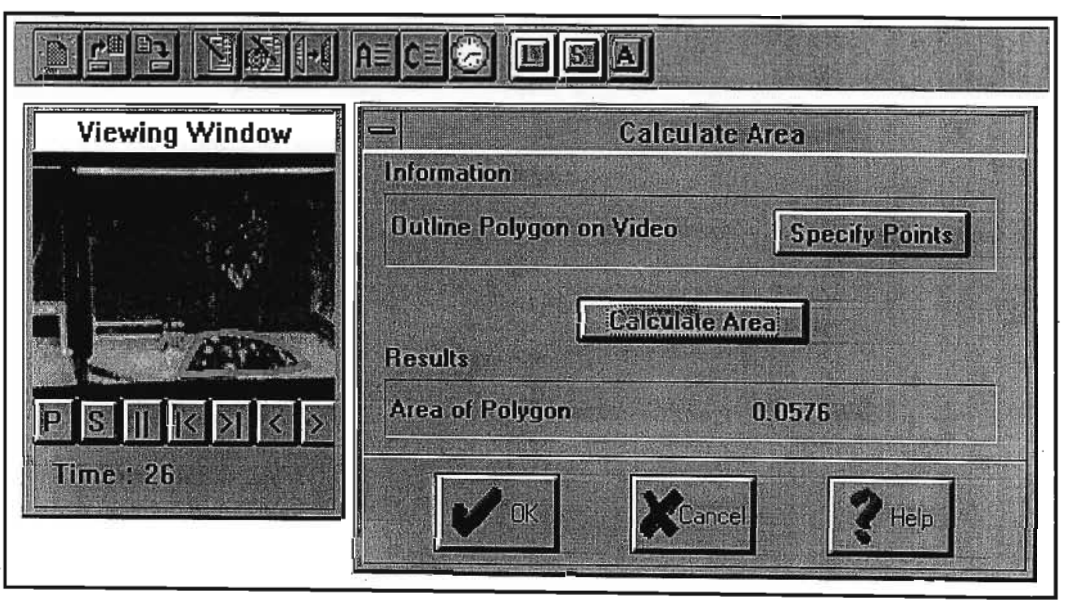

Figure 5.7 : Calculation of Area

5.2.1.3 ANALYSIS

In order to gain a better understanding of the subjects, in terms of the behaviours they displayed most often, a frequency analysis was conducted. In an attempt to investigate the possibility of the existence of behaviour patterns, a sequencing analysis was conducted. The files generated have been included in Appendix 1.

5.3 DISCUSSION

Behavioural researchers often record experiments on videotape, however the literature survey on computerised behavioural systems, presented in Chapter 2, indicates only one system that currently assists the researcher with the extraction of information from the video and the analysis thereof. This system is The Video Tape Analysis System by Noldus. However Wawra [WAWR94] stated that this component of the Noldus software is considered too expensive for some members of universities.

These circumstances suggest that there is a need for a more readily available computer-based video analysis system. VAS is, in some respects an attempt to meet this need. VAS will now be discussed in terms of the features implemented to meet this need.

5.3.1 PRELIMINARY REQUIREMENTS

A prerequisite of VAS, is that the video to be used must be in the AVI format, as the video is manipulated through the media control interface driver, MCIAVI.

The Audio/Video Interleaved (AVI) standard does not define how the video will be captured, compressed or played back, it merely defines how the video and audio will be stored together on the hard disk. [WODA94]

Wodaski [WODA94], stated that ever since the introduction of the AVI format, it has become a standard that "the industry can rally around". The fact that several companies have introduced hardware and software based on the AVI standard, seems to indicate that it could become a de facto standard.

The reasons for using the media control interface, as previously outlined in Chapter 5, are :

- From the perspective of the programmer, the MCI provides a command string interface, which is easy to use.
- When invoking a device operation with MCI, the application may request a notification event be generated upon completion of that operation. This is useful to verify whether or not a command was successfully completed.
- MCI performs certain system operations such as `sysinfo`, which given a `device type`, allows applications to determine the number of instances of that type, their names and which of those instances are currently open.
- The windows environment provides video cassette recorder (VCR) services through a device driver that is based on the MCI command set for VCRs. At the moment the system accommodates video in the form of .AVI files stored on disk. If the scope of the system where to be extended in order to accommodate video tapes, it could easily be done using the MCI drivers.
- The availability of the MCI drivers and documentation, also served to make it a more favourable choice.

The fact that VAS requires the video to be worked with in the form of a .AVI file, does not preclude it from being used with video on videotape, or video in some other computer based video file format.

In both these cases, the video would first have to be converted to a .AVI file. Each of these scenarios will now be discussed.

5.3.1.1 VIDEOTAPES

Video on a videotape has to be captured from the videotape to computer disk. This process requires specialised hardware, and software.

Hardware

Video capture boards/cards are specially developed interfaces which are used for the capture of video from videotape to a file. There are various boards available from commercial vendors.

Wodaski [WODA94] recommends the following video capture boards, for the following reasons:

- **Intel Smart Video Recorder** : This board allows real time compression. It permits the capture of raw (uncompressed) video, which can be edited and compressed at a later stage.
- **VideoLogic Captivator** : Captures both single frames, and motion video. This board provides good colour accuracy, sharpness and contrast.
- **Creative Labs Video Spigot** : Captures up to thirty frames per second at 160x120 pixels
- **Media Space / DVA-400** : This board works reliably and predictably. It offers fullscreen, full motion video capture. However it does not operate with software playback. The same hardware used for recording is needed for the playback.
- **Bravado** : This board provides video overlay, still image capture, motion video capture, real time compression with hardware assisted playback, VGA output to a tape, and Super VGA windows resolution. This Bravado, however is expensive.
- **Super Video Windows** : Permits overlay, and video capture, both still and motion capture.
- **Video Blaster** : This board is the most reasonably priced. Provides overlay and video capture.

All of these boards are compatible with the .AVI standards. In addition Ozer [OZER95] reviewed 11 other video capture boards, that could also be used. The development of video capture boards is an area which is currently experiencing tremendous growth.

Software

There are various video capture software products available. These include, among others:

- **Video For Windows** : Video for Windows described by Botto, [BOTT95] comprises a suite of programs providing the following capabilities :
  - Editing of the colour pallets of 8-bit video and bitmap using PALEDIT
  - Capture of video using VIDCAP
  - Compress a video using one of several different algorithms using VIDEDIT
  - Edit video sequences using VIDEDIT
- **Adobe Premiere** : Video can be captured using ADOBE VIDEO CAPTURE Program. In addition extensive video editing facilities are included. These are described by Wodaski [WODA94]. Two of the most significant are :
  - Filters : Provide the facility to modify each frame of the video in some way e.g. Blurring, sharpening, embossing, inverting, and cropping
  - Transitions : Are used to move from one video clip to another
- **MergeMedia** : Supports video capture, and in addition supports editing. The editing capabilities highlighted by Wodaski [WODA94] are :
  - Inclusion of features for creating and moving text into, and out of video clips.
  - Inclusion of the facility that permits the overlay of several video clips

All of these packages support the capturing of video directly from a videotape to a .AVI file, which can then be used by VAS.

5.3.1.2 VIDEO FORMATS

Digital video may exist in many different formats. The dimensions according to which these formats differ have been outlined in Chapter 2. Some of the more commonly used formats (excluding AVI) include :

- MPEG
- QuickTime

If the video file to be worked with is in any of these alternative formats, it has to be converted to the AVI format. This can be achieved by the use of special software that convert video from one format to another.

Example :

- **SmartVid** is a program that converts video files between the AVI and QuickTime formats
- **DMPEG and RawRip** are programs that are collectively used to convert between the MPEG and AVI formats.

5.3.2 MODULES IN VAS

VAS consists of three distinct modules, the set-up module, the functional module and the analysis module.

The set-up module defines the basic operational element, that is the project. The functional module is used for the extraction of observational information. The analysis module, is used to perform certain analyses. All information generated in VAS are saved to text files.

The functional module was kept separate from the analysis module, because it maintained program simplicity, whilst permitting the user to analyse the observed data with other analysis packages as well.

5.3.3 CALCULATION TOOLS

VAS includes tools to calculate distances/lengths, speeds and areas.

The literature examined in Chapter 2, presented several computerised behavioural analysis systems, however none indicated the inclusion of facilities for such calculations. Even the most recent, and highly recommended Noldus suite of software does not mention the existence of such tools.

Prior to the performance of any distance, speed or area calculation, the scale to be used must be set. This is achieved by specifying two points on the video and the actual distance between these points. Provided that the video is taped from a static location, and not zoomed in or out, this scale needs only be set once. However if either of these conditions are not met, then the scale has to be redefined prior to each successive calculation.

A feature of all the calculation tools is that as a sequence of points is being specified, lines connecting these points are immediately drawn in the viewing window. This was achieved by playing the video in a window created by the author as apposed to using the default window provided by the media control interface. By doing this it was possible to monitor the viewing window for the click of the right mouse button.

5.3.4 ANNOTATIONS

VAS accommodates nested behaviours as well as simultaneous behaviours. Recall, nested behaviours refers to a concept where one behaviour induces another, and the latter lasts for the duration of the former. Simultaneous behaviours refer to those behaviours that occur at

the same time. Nested behaviours are indicated by the use of the open and close square brackets (i.e. '[', ']'), and simultaneous behaviours by the use of a simultaneous behaviour index (i.e. a number).

When noting annotations indicating the interaction of two subjects, then subject indicators are used. These are "(1)" to indicate subject1, and "(2)" to indicate subject2.

The notation used, was chosen in order to maintain simplicity, whilst intuitively convey the purpose of its use.

While VAS does not perform any validation in terms of the semantics of the annotations, it does ensure that the annotations are syntactically correct. This is achieved by parsing the annotations entered by the user, and ensuring that the notation used is correct.

5.3.5 ANALYSES

VAS provides a frequency analysis, and sequencing analysis. In addition a comparison of projects may also be conducted.

A sequencing analysis is a frequency count of all (preceding behaviour : succeeding behaviour) pairs where the preceding behaviours, and succeeding behaviours are elements of the set of target behaviours.

When there are two interacting subjects, the sequencing analysis is conducted separately for all annotations of subject1 alone, subject2 alone, and subject1 and subject2 jointly. So there are in effect three sequencing analyses performed, which are then saved to a file, under the headings *Subject1*, *Subject2*, *Subject1 and Subject2*.

Since the data file generated in VAS, is text based, it could easily be formatted for use in other analysis packages, hence the user is not restricted to only the analyses offered in VAS.

5.4 LIMITATIONS AND FUTURE RESEARCH DIRECTIONS

A number of areas have been identified as possible candidates for improvement, each of which will now be discussed.

5.4.1 TARGET BEHAVIORS

The system requires that the user define the set of target behaviours prior to the commencement of the information extraction process. This set of behaviours cannot be edited during the information extraction process. A more feasible approach would have been to allow the user to include more target behaviours to the list during the information extraction process, as there are instances when all target behaviours are not known in priori. This approach was not previously considered, as the researchers previously interviewed indicated that they generally have a predefined list of target behaviours, that they would be looking for. This feature, however, could be included with the addition of the appropriate interface, and a routine to include the newly specified target behaviour in the list of target behaviours.

5.4.2 TIMERS

The system permits the user to start as many timers as required, at any point. Each timer has a unique name, and may be used only once. Another useful extension, would be one that would permit the user to use a particular timer more than once. For example if one timer were required to keep a total of, the duration of a particular displayed behaviour which occurred more than once, at various points in the video, then the user would want to stop and start a timer more than once. At the moment, this could be achieved by using several timers, and thereafter summing the duration of each.

5.4.3 A GROUP OF SUBJECTS

When making annotations of a group of subjects the user is limited to recording only group activities. A method of overcoming this limitation, would be to extend the scope of this option, to include annotations about individual subjects within the group, as well as annotations indicating interactions among subjects within the group. The former could be implemented through the use of subject indicators, and the latter through the use of subject indicators and modifiers. Modifiers would indicate the subject(s) that was/were affected by the displayed behaviour.

5.4.4 ZOOMING THE FIELD OF VIEW

Calculations in VAS are conducted in terms of a previously defined scale. This scale, however has to be redefined each time the video is zoomed in or out.

An alternative approach to accommodating the zooming of the video, would be to use a pattern recognition in the following manner. Initially two points (objects) on the video, and the actual distance between them would need to be specified. These points (objects) must thereafter always be in the viewing plane, when measurements are taken. Each time a calculation is conducted, the system would automatically locate these points on the video, and redefine the scale accordingly. Hence the system would automatically detect and accommodate the zooming and or movement of the video camera.

5.4.5 VIDEOTAPES

Video in VAS, is handled in the form of a .AVI file. Many of the researchers however have the video to be worked with, on videotapes. A useful extension of the system, would be to accommodate video on videotapes.

In order to maintain a runtime clock of the video, the video camera used for the recording must have had the facility to include a timecode. If this was the case, then a routine could be written to access this timecode, and hence maintain the video runtime clock. However, if it was not the case, then specific hardware would be required. A time code generator would be needed to write a time code to the video, and a time code reader would be required to retrieve the time code from the videotape during playback, and forward it to the computer.

In addition, in order to control the video cassette recorder from the computer, additional hardware would be required. This would provide an interface to the VCR that would allow the computer to control certain functions of the VCR, such as play, stop, pause etc.

When deciding about the feasibility of this approach, one needs to take into account the overheads in terms of cost that would be incurred, and weigh these against the overheads of using VAS as it currently is.

In order to use VAS to score and annotate a video on a videotape, this video must be captured to a .AVI file. This requires a video capture card, and specific software, which is generally provided when the card is purchased. In addition, one would need a fairly large amount of space on the hard drive. A fifty second long video clip was captured to a file. The size of this file was 4.97 Megabytes. While this would vary slightly depending on the compressor/decompressor that is used, it never the less provides a good indication of the space that would be required. Hence the cost of a larger hard drive must also be considered.

5.4.6 TOOLBOX

The interviews conducted revealed that a system that manipulates a video, and allows information to be inserted and thus synchronised with the video runtime clock, could be useful in areas other than behavioural research. Prof. Basson, from the Department of Psychology at the University of Natal, indicated that a he would use such a system as a

teaching tool. Since similar systems could be used in more than one different environment, the idea of a toolbox became particularly appealing. The toolbox would ideally contain classes, much the same as the ObjectWindows classes, that could be used by a programmer, to design other similar computerised systems that manipulate video. This toolbox could take the form of a dynamic linked library. A prototype dynamic linked library was developed which contained routines similar to the ones used in VAS, to calculate lengths, speeds and areas. This dynamic linked library could be further extended to include all classes and features implemented in VAS.

5.5 CONCLUSION

VAS has been successfully implemented as a system that can be used to extract observational data from a video. It also supports certain analyses as was illustrated by the case study conducted.

6. CONCLUSION

The objectives of the study will first be restated, in order to draw conclusions based upon them.

Restatement of Objectives :

1. To examine the domain of computerised behavioural analysis systems, paying special attention to the need for computer based video analysis systems.

2. To develop a computer based video analysis system, called VAS.

3. To investigate the feasibility of producing a generic toolbox that could later be used to build applications, firstly in this domain, and secondly in other areas that would benefit from computer based video systems, such as teaching.

The literature surveyed provided an overview of the use of computer based systems in behavioural research, and indicated the need for computer-based video analysis systems. A distinct set of requirements for a video analysis system was obtained from interviews conducted with a group of local researchers.

A windows-based video analysis software system, called VAS, has been developed. The system was designed to primarily meet the immediate needs of the local behavioural researchers. VAS manipulates video in the form of a .AVI file which is on disk, through the use of the media control interface. Using VAS, observational data can be extracted from the video in the form of annotations, comments, timed events, as well as measures of distance/length, speed and area. In addition VAS provides certain analyses, these being, frequency analysis, sequencing analysis, and comparison of projects.

Although it has been shown that VAS could be improved in certain respects, VAS is a useable video analysis system, as was illustrated by the case study, and thus fulfils the objectives of the study.

7. LITERATURE CITED

[ALBO92] Albonetti M.E., Lazarus J., Dickins D.W. & Whybrow A., Schonheiter R., Prenter J. & Elwood R., Newman J.D., Odberg F.O., Kaiser L., Pham-Delegue M.H., Kerguelen V., Fentress J.C., Boccia M.L. : Software Multiple Review. The Observer - Software for Behavioural Research Version 2.0, **Ethology, Ecology and Evolution**, 1992, 4, 401-414.

[BERR87] Berry J.S., Holtzer O., Norman J.M. : Computers in the Field, **Bulletin of the Entomological Society of America**, 1987, 33, 209.

[BORL94] Borland ObjectWindows Programmer's Guide, **Borland International**, 1994

[BOTT95] Botto F. : PC Multimedia - An Introduction to Authoring Applications, **Butterworth-Heinemann**, Oxford, 1995, 290-505

[COYO87] Coyne A.C. : A Software-Based Millisecond Timer for the CP/M Operating System, **Behaviour Research Methods and Instrumentation**, 1987, 19, 47-48.

[DENI83] Deni R., Szijarto A.E., Fantauzzo C. : Basic Programs for Observational Research Using the TRS-80 Model 100 Portable and Model 4 Computers, **Behaviour Research Methods and Instrumentation**, 1983, 15, 616.

[FLOW82] Flowers, J.H. : Some Simple Apple II Software for the Collection and Analysis of Observational Data, **Behaviour Research Methods and Instrumentation**, 1982, 14, 241-249.

[GIBB95] Gibbs S.J. : Multimedia Programming Objects, Environments and Frameworks, **ACM Press**, New York, 1995, 27-55

[GRAV88] Graves R., Bradley R. : More on the Millisecond Timing and Tachistoscope Applications For the IBM PC, **Behaviour Research Methods and Instrumentation**, 1988, 20, 408-412.

[GURE94] Gurewich O., Gurewich N. : Borland C++ Multimedia Programming, USA, **Sybex Inc.**, 1994

[HILE91] Hile M.G. : Hand-held Behavioural Observations : The Observer, **Behavioural Assessment**, 1991, 13, 187-196.

[KEEN94] Keene J. : Software Review - The Observer, **Behavioural and Cognitive Psychotherapy**, 1994, 22, 181-183

[LUTH89] Luther A.C. : Digital Video in the PC Environment, **McGraw-Hill**, 1989 11-63 187-204

[MCI] Online Information about the MCI provided with Video For Windows

[NOLD89] Noldus L.P.J.J., van de Loo E.L.H. M., Timers P.H.A. : Computers in Behavioural Research, **Nature**, 1989, 431, 767-768.

[NOLD91] Noldus L.P.J.J. : The Observer : A Software System for Collection and Analysis of Observational Data, **Behaviour Research Methods and Instrumentation**, 1991, 23, 415-429.

[NOLD93] Noldus Information Technology : The Observer Version 3.0, **Technical Review**, 1993.

[NOLD94A] Noldus Information Technology : The Observer - Video Tape Analysis System, **Pamphlet**, 1994.

[NOLD94B] Noldus Information Technology : EthoVision, **Pamphlet**, 1994

[OZER95] Ozer J. : Digital Video Shot-by-Shot, **Byte,** 1995, 14 No. 7, 104-152

[PERO85] Perone M. : A Software System for the Real-Time use of TRS-80 Microcomputers, **Behaviour Research Methods and Instrumentation**, 1985, 17, 119-121.

[RICH95] Richter J. : Q&A Win32, **Microsoft Systems Journal**, Feb 1995, 97-102

[TORG77] Torgerson, L. : Datamyte 900, **Behaviour Research Methods and Instrumentation**, 1977, 9, 405-406.

[TOUR92] Tourtellot M.K. : Software Review - The Observer, **Journal of Insect Behaviour**, 1992, 5, 415-416.

[UNWI87] Unwin D.M., Martin P. : Recording Behaviour Using a Portable Microcomputer, **Behaviour**, 1987, 101, 87-100.

[VISS93] Visser M.E. : The Observer - A Software Package for Behavioural Observations, **Animal Behaviour**, 1993, 45, 1045-1048.

[WALN94] Walnum C. : Borland C++ 4.x Tips, Tricks and Traps, **Que Corp.**, USA, 1994

[WAWR94] Wawra M. : The Observer 3.0 - A Software Package for Behavioural Observations, **Ethology**, 1994, 95-96.

[WHIT71] White, R.E.C. : WRATS : A Computer Compatible System For Automatically Recording and Transcribing Behavioural Data, **Behaviour**, 1971, 40, 135-161.

[WHIT88] Whiten A., Barton R.A. : Demise of the Checksheet : Using Off-the-shelf Miniature Hand-held Event Recorders for remote Fieldwork Applications, **Trends in Ecology and Evolution**, 1988, 40, 135-161.

[WODA94] Wodaski R. : Multimedia Madness, **SAMS Pub.**, USA, 1994

[YAOP94] Yao, P. : Borland C++ 4.0 Programming for Windows, **Random House Inc.**, New York, 1994

8 APPENDIX 1 - OUTPUT FILES GENERATED DURING THE CASE STUDY

Curly brackets i.e. "{" and "}" have been used to provide additional information, about the data in the files.

8.1 EXPERIMENT FILE

EXPERIMENT FILE CREATED BY THE VIDEO ANALYSIS SYSTEM
DESCRIPTION
Interaction between female 6 and male 8 *{the description of the experiment}*
NOS
Two *{there are two interacting subjects in the experiment}*
LOE
50 Sec *{specification of the length of the experiment}*
FTI
10 Sec *{observations are to be recorded by the system at a fti of 10 seconds}*
TB *{list of target behaviours}*
attack
cling
defend
kick
shuffle
threaten
END OF EXPERIMENT FILE

8.2 PROJECT FILE

PROJECT FILE CREATED BY THE VIDEO ANALYSIS SYSTEM

D:\TST\TESTING\FIGHT.EXP *{the experiment file}*

C:\FIGHT.AVI *{the video file}*

D:\TST\TESTING\FIGHT2.DAT *{the data file}*

8.3 DATA FILE

COMMENTS *{*** is used to separate the time from the actual comment}*

0***Male climbing wire to attack female

15***Male kicking up sawdust and shuffling with hindfeet

30***Male approaching with body very low

34***Male sniffing sawdust below female

8***Fight1 shows start of male attack

8***Male attacking female on wire. Female defends with forepaws

ANNOTATIONS *{### preceding an annotation indicates the annotation is for subject1 and subject2 i.e. interacting behaviours}*

10***###(1) 1 shuffle [(1) 1 kick] ; (2) 1 cling ; (1)attack; (2)defend;

20***###(1)attack; (2)defend; (1)attack; (2)defend;

30***###(1) 1 shuffle;(2) 1 cling; (1)shuffle [(1)kick];

40***###(1)threaten; (2)cling; (1)shuffle; (2)cling;

50***###(1) 1 threaten; (1) 1 kick; (2) 1 cling;

TIMED EVENTS *{manner in which timed events are recorded is : name of timed event;start frame; stop frame; start time; stop time; duration }*

fight1;122;171;8200;11400;3200

threaten;171;446;11400;28800;18400

fight7;182;243;12200;16200;4000

8.4 ANALYSES

8.4.1 FREQUENCY ANALYSIS FILE

Frequency of Target Behaviours

attack	3
cling	5
defend	3
kick	3
shuffle	4
threaten	2

8.4.2 SEQUENCING ANALYSIS

Subject 1 *{the sequencing analysis for subject 1 alone and subject 2 alone was not performed as there were no annotations for subject1 alone, and subject2 alone. Recall in the case study, the focus was interacting behaviours only.}*

Subject 2

Subject 1 & Subject 2 *{indicates the sequencing analysis for the both subjects jointly}*

(1)attack (2)defend	3
(1)kick (1)threaten	1
(1)kick (2)defend	1
(1)shuffle (1)kick	1
(1)shuffle (2)cling	1
(1)shuffle (2)defend	1

(1)threaten (2)cling	1
(2)cling (1)kick	2
(2)cling (1)shuffle	1
(2)cling (1)threaten	1
(2)cling (2)cling	1
(2)cling (2)defend	1
(2)defend (1)attack	2
(2)defend (1)shuffle	1
(2)defend (2)cling	1