MEASUREMENT AND VALIDATION OF WATERLINES AND SURFACE CURRENTS USING SURF-ZONE VIDEO IMAGING

By

Jaysen Naicker

Submitted in fulfilment of the academic
requirements for the degree of
Master of Science in the
School of Pure and Applied Physics,
University of Natal

2001

Abstract

The continuous monitoring of beaches and structures near, or in the surf-zone, has become a vital task from an environmental and economic standpoint. Specifically, the Durban beaches are continually nourished with sand that is dredged south of the Durban Harbour mouth. Constant monitoring is essential to control the erosion and accretion of these beaches. Currently, infrequent and labour-intensive manual surveys are being used to fulfil this task.

This dissertation describes the techniques used to process and validate surf-zone video images to extract information, which is equivalent but more frequent and cost effective to that obtained using manual surveying methods. The emphasis is on the extraction of accurate hourly waterlines by analysing video images obtained from Coastal Imaging stations located at Addington Beach and North Beach in Durban, South Africa and the measurement of rip currents in the surf-zone from video images together with the validation of these results.

The implementation of a neural network incorporating non-local image pixel data is more reliable than previously used methods, for example, grayscale thresholding or the delta-discriminator. Grayscale thresholding relies on the difference between the water and the sand pixel intensities. The delta-discriminator incorporates hue-saturation-lightness (HSL) values to improve the discrimination between water and sand. These methods could not, however, overcome the problem of varying seasonal and diurnal light intensities and require the prior choice of threshold values.

The use of non-local data, such as the average RGB-values, in addition to individual pixel values as inputs to a neural network is shown to give better results in changing ambient lighting conditions. Wavelet and Fourier analysis of the temporal nature of the breaking waves in the surf-zone is also used to calculate more accurate waterlines. These extracted waterlines are validated by data collected from manual surveys.

The use of Digital Correlation Image Velocimetry (DCIV) has been proven as a successful method in tracing velocity flow fields in a fluid medium. Typically, the medium is populated with distinct seeds that are tracked through successive video frames using a cross-correlation technique. It is shown that DCIV can be used in surf-zone images to measure surface rip currents by tracking the structure of the advected foam. This technique was validated by simultaneously tracking a large number of yellow corks scattered on the water surface.

Preface

This dissertation describes the work undertaken by the author in the Applied Physics Group, School of Pure and Applied Physics, University of Natal, Durban, from January 2000 until December 2001 under the supervision of Prof. M.J. Alport. This dissertation is a continuation of work that the author has participated in since October 1997.

The author declares this dissertation to be his own original work except where other work has been used and duly acknowledged. This thesis has not been submitted in any form for any other degree or diploma purposes to any other tertiary institution.

Table of Contents

LIST OF FIGURES

Number *Page*

ACKNOWLEDGMENTS

The author wishes to thank his supervisor Prof. M.J. Alport for his guidance and support that has resulted in a number of interesting avenues of thought and research being pursed. The author wishes to thank the University of Natal and the National Research Fund (NRF) for funding this research. The Council for Scientific and Industrial Research (CSIR) needs to be thanked for funding much of the equipment used in the research on the Video Coastal Monitoring project. Christoph Soltau from the CSIR, Stellenbosch must be thanked for his collaboration on the Video Coastal Monitoring project. Mr Jonathan Jackson from the Department of Survey must be thanked for performing the manual surveys necessary for this research and for the discussions on the topic of surveying.

The author also wishes to thank his fellow colleagues at the Applied Physics Group (APG) for their help and support. The members of the Physics workshop, especially Derek Dixon, Robert Piasecki, Lloyd Kolb and Stanley Perumal, need to be greatly thanked for installation and maintenance of the coastal imaging stations and for their help on a number of other technical issues. Keith Barnett and Andrew Mather of the Coastal and Drainage Division, Durban Metro must also be thanked for their collaboration with matters related to Durban Metro and the beach nourishment scheme. The author also wishes to thank Dr Kessie Govender for the discussions on Particle Image Velocimetry (PIV).

NOTATIONS

Symbols

*	Complex conjugate operator
ϕ	Angle of rotation so the beach is parallel to the image x-axis
θ	Angle of roll
τ	Angle of tilt
ω	Angle of rotation
σ	Standard deviation
$\Im$	Forward Fourier Transform
$\wp$	Perspective matrix
$\otimes$	Convolution operator
o	Correlation operator
$\mathbf{P}$	Principal line

Abbreviations

ANN	Artificial Neural Network
AVG	Average image
CCS	Camera Co-ordinate System
DCIV	Digital Correlation Image Velocimetry
DFT	Discrete Fourier Transform
DWT	Discrete Wavelet Transform
FT	Fourier Transform
FFT	Fast Fourier Transform
fps	Frames per second
GCP	Ground Control Point
HSL	Hue-Saturation-Lightness colour space
MLP	Multi-Layer Perceptron

MSE	Mean Square Error
PIV	Particle Image Velocimetry
$\mathbf{R_x}$	Rotation matrix about the x-axis
$\mathbf{R_y}$	Rotation matrix about the y-axis
$\mathbf{R_z}$	Rotation matrix about the z-axis
RGB	Red-Green-Blue colour space
SCS	Scene Co-ordinate System
SNAP	Instantaneous image
STD	Standard deviation
WT	Wavelet Transform

Chapter 1

INTRODUCTION

Advanced analysis techniques and instrumentation have lead to more efficient and cost effective solutions in various disciplines. The use of video coastal monitoring follows this trend in providing a better and more frequent means of observing the changes in beach morphology.

The use of imaging to measure various processes related to the sea is not a new concept. Oblique single and stereo images were used to determine wave heights, water level and the current in the surf-zone in the 1970's (Maresca and Seibel [28]). Digitised video images were used to study tracer dispersion in the nearshore region (Rodriquez et al. [38]). Coastal imaging is not limited to terrestrial based methods only. Satellites, aeroplanes and hot air balloons have been used as platforms from which images of the surf-zone were captured (Horikawa [21]).

Immediate qualitative information is available from the video images for certain coastal processes like sandbar formation and evolution. The term waterline as used in this thesis is related to the *low water mark* used by surveyors. The *low water mark* is defined as "*the lowest line to which the water of the sea recedes during periods of ordinary spring tides*"[1]. The position of a waterline may be determined by an analysis of one or more consecutive video images captured by a video camera overlooking the surf-zone. This waterline will trace a contour, which will lie somewhat landward of the *low water mark* depending on local conditions such as tide height, wind speed, wind direction, temperature and pressure.

[1] Simpson K.W., Sweeney G.M.T., The Land Surveyor and the Law, p 191, University of Natal Press, Pietermaritzburg, 1973

The hourly calculated waterlines during the tidal cycle can be used to measure the intertidal beach profile for a particular day (Janssen [23]). Comparison of these beach profiles over different periods (weeks, months, years) can then provide vital information on the trends of accretion and erosion of the beaches. The waterlines can also be used to report on the effective beach width, that is, the width of the strips of sand available for recreation or buffering of coastal building structures.

Knowledge of the beach width is of particular importance for the Durban beaches as they are continuously nourished by a dredging-pumping scheme. The littoral northward transport of sand round the Bluff onto the Durban beaches is interrupted by the piers at the harbour mouth that ensure safe passage for ships into the harbour. To overcome this problem the sand trap just south of the harbour entrance is regularly dredged using the Portnet dredger. This sand is then discharged into a 5000 m^3 concrete hopper where it is filtered and then pumped onto the beaches at several distribution points along a 3.5 km length of the beachfront. On average, 280 000 m^3 of sand is pumped onto the beaches per annum[2].

Durban Metro has expressed interest in utilising video coastal monitoring as a means of constantly monitoring the effective beach width especially with the advent of the proposed Jet-pumping scheme, which will replace the current dredging scheme. The Jet-pumping scheme entails the building of another pier on the south side of the harbour entrance. Jet pumps built into the pier structure will dredge up sand, which will be pumped under the harbour using pipes to the existing sand outlets. By utilising the effective beach width there will be an early indication of which beaches need immediate replenishing. The specific concern is that the jJet-pumping scheme is not as flexible as dredging where the relative volumes of sand dumped on the beaches and out to sea can be adjusted depending on the equilibrium beach widths.

[2] http://www.durban.gov.za/engineering/drainage/sandpump.htm

The surface velocity fields in the surf-zone can also be calculated by measuring the displacement of foam structures in consecutive video images using the Digital Correlation Image Velocimetry (DCIV) technique. These velocity fields are of particular interest as they can indicate the presence of rip current next to structures such as piers. These rip currents can cause severe scow, undermining and damage of these structures, as was the case with the Dairy Beach Pier, which was severely damaged during the storms in 1998.

Chapter 2 provides an overview of video coastal monitoring from its inception in 1989 till the present. Included is a description of the video coastal monitoring project at the University of Natal. Chapter 3 provides a discussion and comparison of the various techniques used to estimate the waterline from video images. Chapter 4 presents the result of applying the Digital Correlation Image Velocimetry (DCIV) in the surf-zone.

Many of the results presented in this thesis use a formalism and software, which although important, have not been included in the body of the thesis. However, for completeness and future reference, this material has been included in a number of Appendices. The formulation of the geometric transforms used in relating image co-ordinates to ground co-ordinates can be found in APPENDIX A. A discussion of the different colour representations for images is provided in APPENDIX B.

An introduction to Artificial Neural Networks (ANNs) and their implementation is presented in APPENDIX C. The theory of the Fourier Transform (FT) is provided in APPENDIX D and the theory of the Wavelet Transform is given in APPENDIX E. APPENDIX F describes the DCIV technique.

A listing of the software routines used can be found in APPENDIX G. Most of the routines were written using IDL[3]. IDL provides a number of image processing tools including an implementation of the Fast Fourier Transform (FFT) and Continuous Wavelet Transform (CWT) that facilitate the development and debugging of the algorithms.

[3] Interactive Data Language. http://www.rsinc.com

Chapter 2

VIDEO COASTAL MONITORING

2.1. Introduction

Scientists express an active curiosity and need to understand the natural phenomena that surround them. The sea, and the processes related to it, have prompted a major part of this scientific endeavour. The region that most people can relate to is the nearshore. This chapter first reviews concepts and then gives a brief survey of the various video coastal monitoring research activities that has taken place during the past 13 years. This discussion then describes the Cyclops[4] coastal imaging stations developed at the University of Natal.

2.2. Concepts used in Video Coastal Monitoring

To facilitate a better understanding of the material and discussion presented in this thesis it is necessary to first explain some of the key concepts used in video coastal imaging. Other concepts will be explained later in the thesis, as they are needed.

2.2.1. SNAP, AVG and STD Images

In this thesis a number of different images are used. Each image is composed of a number of discrete points, which are called pixels (picture elements). For a colour image, each pixel is defined by a RGB (red, green, blue) triplet, while in a grayscale image each pixel is represented by a single intensity value. The grayscale image is usually calculated by averaging the RGB triplet values for each pixel in an image. More details of this and other colour models can be found in APPENDIX B.

[4] Cyclops is the one-eyed giant from Greek mythology.

The instantaneous image captured from a CCD camera is referred to as a SNAP (snapshot) image. Once a sequence of SNAP images has been captured an AVG (average) image can be calculated by taking the average pixel intensity value for each pixel in each colour plane.

After the AVG image has been calculated, the STD (standard deviation) image was obtained by calculating the standard deviation of each pixel's intensity values for the sequence of SNAP images. The standard deviation σ is defined as[5]

$$\sigma = \sqrt{\frac{\sum_{i=0}^{N-1}\left(x_i - \bar{x}\right)^2}{N-1}} \qquad [2.1]$$

where x_i is the ith pixel's grayscale value, $\bar{x}$ is the pixel's average grayscale value and N is the number of images in the sequence. A large value of standard deviation in any of the three colour planes of the STD image corresponds to a region in the image where a large variation in the pixel intensity has occurred over time.

2.2.2. Keogram

A useful construction that gives a compressed representation of the wave motion in a sequence of images is the keogram. A keogram is created by extracting a single line from successive images and stacking them on top of each other to create a time-displacement image as shown in the example in Figure 2.1. The temporal evolution of individual features such as waves can be tracked by a visual inspection of the keogram. Other authors also referred to the keogram as a timestack.

[5] Press W. H., Teukolsky S. A., Vetterling W. T., Flannery B.P., Numerical Recipes in C: The Art of Scientific Computing (2nd Ed), p 611, Cambridge University Press, USA, 1992

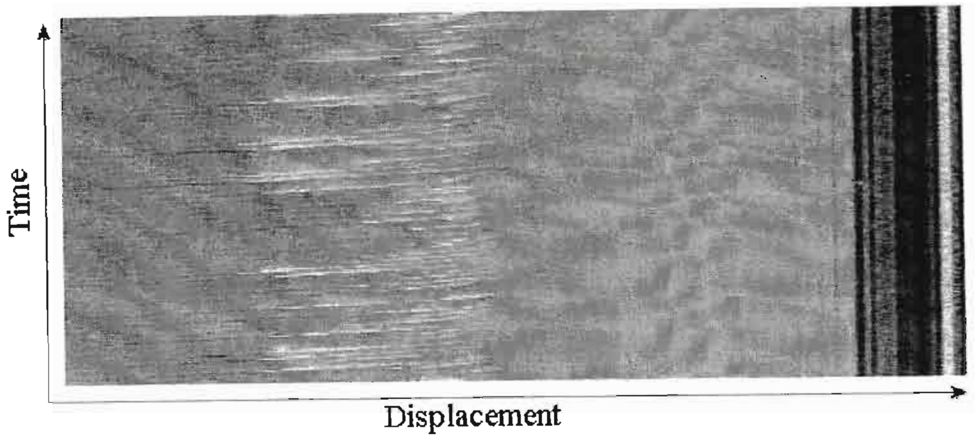


Figure 2.1 An example of a keogram (time-displacement image). The white streaks are individual waves that propagate towards the shore.

2.2.3. Linear Wave Theory

Linear wave theory (LeBlond and Mysak [25]) defines the relationship between the angular frequency ω of the waves, and water depth *h* as

$$\omega^2 = gk \tanh(kh) \qquad [2.2]$$

where the wavenumber *k*, gravitational acceleration *g*.

Using the relationship $\omega = ck$, equation [2.2] is rewritten as

$$c^2 = \frac{g}{k} \tanh(kh) \qquad [2.3]$$

where *c* is the phase speed of the wave. For shallow water $\tanh(kh) \approx kh$ and equation [2.3] simplifies to

$$c \approx \sqrt{gh} \qquad [2.4]$$

This indicates that the wave phase speed of a wave is approximately equal to the square root of the water depth. It should be noted that the dispersion (equation [2.2]) is strictly only applicable for small amplitude (linear) waves, that is, not rollers.

2.3. Historical Review of Video Coastal Monitoring

2.3.1. 1989

Lippmann and Holman (Lippmann et al. [26]) presented a video technique to determine sandbar morphology, which required capturing time-averaged images instead of the traditional instantaneous image. Oblique images were captured over a 10-minute period at a rate of one frame per second (fps) and averaged. The sandbars in the nearshore stood out as white (high pixel intensity) bands. This white band is the result of waves breaking over the shallower regions of the sandbar.

The fluctuations due to transient incident waves are averaged out in the time-averaged image, however the problem of residual foam remained. To overcome this, a differencing time exposure image was created. This image was obtained by taking the difference between two sequential video images (captured typically 0.5-1.0 seconds apart) and then averaging the difference images over a 10-minute period. This difference image gave better correlation with the bathymetry results.

2.3.2. 1992

The ARGUS[6] research program, which was started in 1992 at the Coastal Imaging Laboratory, Oregon State University, has resulted in the installation of a number of ARGUS coastal imaging stations. Each station consisted of one or more video cameras, connected to a personal computer, watching a beach obliquely. During the daylight hours, an hourly instantaneous image and time-averaged image were captured and stored locally.

[6] Argus is the giant with a hundred eyes, from Greek mythology.

ARGUS stations[7] can now be found at various worldwide locations including Agate Beach (Oregon), Duck (North Carolina), Egmond Aan Zee (the Netherlands), Lake Erie (Ohio), La Jolla (Scripps Institution), Muriwai (New Zealand), Noordwijk (the Netherlands), Palm Beach (Australia), Perranporth (England), Teignmouth (England) and Waimea Bay (Hawaii).

2.3.3. 1996

Aarninkhof [1] compared the output of the numerical wave decay submodel UNIBEST-TC to the sandbar position obtained from time averaged images. The UNIBEST model is an upgraded version on the parametric wave dissipation model proposed by J.A. Battjes and J.P.F.M. Janssen (Aarninkhof [1], Battjes and Janssen [8], Battjes and Stive [9], Janssen [23]).

An inverse model MONIMORPH (MONItoring MORPHology) was developed, which estimated the bar bathymetry using time averaged images and wave characteristics. The model was tested at the ARGUS stations at Duck and Noordwijk. The results at Duck showed the estimation of the top of the sand bar to be accurate to within 10 to 20 cm while the position of the sand bar was shown to be 30 to 40 m shoreward of the actual position.

Noordwijk bathymetry data was not available for the period of the investigation and fictitious beach profiles were used hence rendering the comparison between the modelled results and actual results unreliable. The scaling parameters derived at Duck for the model could not be confirmed to be applicable at Noordwijk or any other ARGUS site.

[7] http://www.vms-argus.com/

2.3.4. 1997

Janssen [23] presented a new method to determine the intertidal beach bathymetry. The ARGUS station at Noordwijk was used to validate the method. Hourly time averaged images and standard deviation images were used to estimate the horizontal co-ordinates of the waterline. Hydrodynamic modelling was used to calculate the vertical co-ordinate of the waterline. By calculating the horizontal and vertical co-ordinates of the waterline in all the images in a tide cycle, the intertidal beach bathymetry was estimated. The mean difference in the horizontal co-ordinates was in the order of 4-8m while the variation in the vertical co-ordinate was in the order of 5-20cm when compared to actual field data.

Aarninkhof et al. [3] presented a paper reporting the results of an inverse model, MONIMORPH (Aarninkhof [1]). Also presented was the method to determine the intertidal beach bathymetry using the average and standard deviation images to calculate the horizontal co-ordinates of the waterline and hydrodynamic modelling to calculate the vertical co-ordinates (Janssen [23]). The method of extracting the horizontal co-ordinates of the waterline was named the Pixel Variance Method (PVM). The PVM determines the optimal shift to best correlate the cross-shore profiles from the average and standard deviation images hence give a better estimate of the position of the sand bars in the image.

Wijnberg and Holman [44] reported on the cyclic nature of the sandbar systems along the Dutch coast. A 2-year data set of hourly images from the ARGUS station at Noordwijk was used to analyse this phenomenon. The sandbar of interest was the outer sandbar, which was projected to have a four-year cycle.

Davidson et al. [13] used data from the ARGUS station at Perranporth to evaluate large scale (km) intertidal beach morphological changes. The calculated difference between the estimated beach morphology and field measurements was less than 11cm. The area of interest was 1.0km by 0.5km.

2.3.5. 1998

Holland [19] used a nine-year data set of images from the ARGUS coastal station at Duck to determine the temporal evolution and spacing of the cusp formations in the nearshore region. It was reported that the cusps formed 2-4 days after a storm event and approximately 50% of the cusps formations could be predicted using an empirical formula.

Plant and Holman [31] used the bathymetry derived from oblique video images and the directly surveyed bathymetry from the ARGUS station at Duck to create a morphological model. All the parameters necessary for the model were estimated from the survey data. The video images could only provide an estimation of the bar crest position. This model was not tested on any other ARGUS sites so the full applicability of the model to any beach was unknown.

Worley et al. [45] developed an aerial video system for monitoring km long (up to 36 kms) stretches of North Carolina beach. Small aircraft were used to fly along the beach and time-averaged images were captured. Complex sandbar morphology was recorded during these flights.

Alport et al. [5] presented results of the work done by Govender [15] in successfully measuring velocity flow fields and turbulence levels in a wave flume using particle image velocimetry (PIV - APPENDIX F). Also presented were the results of the first coastal imaging station in South Africa. The intertidal beach profile was calculated from the video images obtained from the coastal imaging station.

2.3.6. 1999

Aarninkhof and Roelvink [4] presented a new technique of estimating the waterline using colour images. The new technique utilises the HSV (hue, saturation, value) colour space. Observations had revealed the clustering of *wet* (in the water) and *dry* (on land) pixels in Hue-Saturation (HS) space. A discriminator function was proposed that classified pixels as either *wet* or *dry*. The interface between the *wet* and *dry* pixels gave the waterline. Also presented was a new model for predicting wave dissipation in the inner surf-zone although no measure of accuracy was given.

Van Enckevort and Wijnberg [42] monitored a triple barred system at the Noordwijk ARGUS station. Images captured in 1995 were used for analysis and focus was directed to the daily and monthly evolution of the middle and inner sand bars. It was observed that rip currents interrupted the middle bar for long periods when it was linear in shape.

Ranasinghe et al. [36] utilised time averaged images at Palm Beach, Sydney to observe rip dynamics in the nearshore. The dark (low pixel intensity) regions in the time-averaged images were used as an indicator of the position of the rip currents. Daily observations of the images showed the longshore movement of the rip channels.

Alport et al. [6] reported on the improvements made by Basson [7] in calculating the waterline using the proposed δ colour discriminator (details of this technique are described in section 3.2.2).

2.3.7. 2000

Basson [7] completed his MSc thesis on imaging techniques applied to the surf-zone. The δ colour discriminator (section 3.2.2) and Artificial Neural Network (ANN) (section 3.2.3) techniques were presented as improvements to the grayscale thresholding technique (section 3.2.1) for estimating the position of the waterline in oblique video images. The accuracy of theses techniques was quantified by fitting a Gaussian to the derivative of a single cross-shore transect taken from the results of each technique. The full width at half maximum (FWHM) value of the Gaussian for the ANN, δ colour discriminator and grayscale thresholding were 2.87, 16.7 and 20.6 respectively. This indicated that the ANN had the best discrimination of the transition from water to land, followed by the δ colour discriminator.

Also as part of his research Basson applied DCIV (APPENDIX F) to the surf-zone images to quantify the velocity vector field. This was done by averaging the displacement vectors calculated using DCIV, over a number of consecutive images. Using equation [2.4] the phase speed of waves was calculated using data from a manual bathymetry survey and compared to the calculated velocity vector field. Good correlation could not be found between the data and it was concluded that the DCIV algorithm used at that time could not be used to correctly track features between wave fronts.

Konicki and Holman [24] presented a study on observing the transverse sand bars at Duck using a 10-year data set of time averaged images captured at the ARGUS station. It was concluded that longshore currents affected the direction and speed of the sand bars. No conclusive evidence could be found to prove that the formation of these bars was a result of strong longshore currents.

Stockdon and Holman [39] report on the development of a new method to estimate nearshore bathymetry from video images. This method first calculates the cross-shore wavenumber by applying the Fourier Transform (FT) to a timestack. This calculated wavenumber was then substituted in equation [2.2], which was rewritten as

$$h = \frac{\tanh^{-1}\left[\frac{\omega^2}{gk}\right]}{k} \qquad [2.5]$$

so the local water depth could be calculated. Bathymetry data and video images from the ARGUS station at DUCK for October 1997 were used to do a comparison. It was reported that the mean difference between the video derived and manual bathymetries was –35 cm.

Aarninkhof et al [2] performed a comparison of the accuracy of video derived waterlines to a number of GPS-surveyed waterlines. The Hue-Saturation discriminator was used to estimate the waterlines from the time-averaged video images. The error was expressed as the deviation in distance between the video derived waterlines and the surveyed waterlines. It was reported that the average horizontal deviation was ~ 5m while the vertical deviation was ~ 0.1m.

The focus of this thesis is the development and comparison of the different waterline extraction techniques and the application DCIV to the surf-zone. In the literature, a number of waterline techniques have been reported on:

(1) The waterline was obtained by using grayscale thresholding on the average image and the differencing time exposure image by Lippmann et al. [26],

(2) Greyscale thresholding was used on standard deviation images by Janssen [23],

(3) A discriminator function was proposed to classify *wet* and *dry* pixels in Hue-Saturation space by Aarninkhof et al. [4] and

(4) Basson [7] used two techniques, namely an Artificial Neural Network and the δ colour discriminator. However, a comprehensive comparison of the accuracy and robustness of these techniques has not yet been completed.

Basson [7] also applied DCIV in an attempt to calculate the velocity flow field in the surf-zone using consecutive images. The same DCIV technique will be applied in Chapter 4 to validate the surface current in a rip channel.

2.4. Cyclops Coastal Video Monitoring

In 1997 Prof M.J. Alport, Applied Physics Group (APG), University of Natal, Durban in collaboration with Dr G. Mocke, Environmentek, Centre for Industrial and Scientific Research (CSIR), Stellenbosch began a research program which included the setting up of a series of coastal monitoring stations similar to the ARGUS stations in South Africa. The first of these stations was erected at Melkbosstrand on the South African southwest coast in December 1997.

The station shown in Figure 2.2 consisted of a colour CCD camera mounted on a 60m Telkom tower. This camera was connected to a personal computer (PC) containing a Miro DC 30 frame grabber. Bitmaps (768 x 576 pixels in size) were captured hourly at 1fps for 10 minutes, making a total of 600 frames. From these 600 images, an instantaneous (SNAP) image, an average (AVG) image and a standard deviation (STD) image were extracted and stored locally in JPEG[8] (Joint Photographic Expert Group) format. These images were then sent via FTP (File Transfer Protocol) to a UNIX PC from which they were periodically transferred to the CSIR for archiving via a Diginet line.

[8] http://www.jpeg.org/

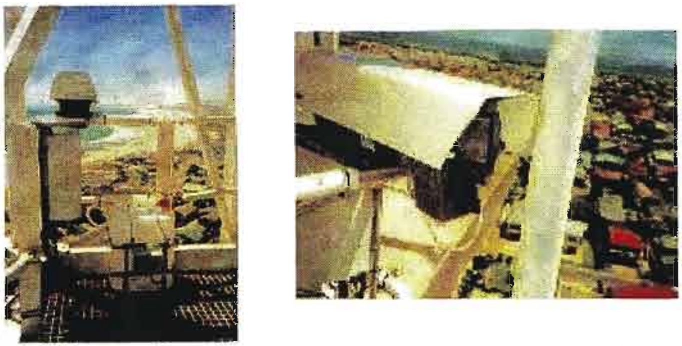


Figure 2.2 Camera installation on the Telkom tower at Melkbosstrand

The AVG image was created by averaging each pixel's intensity value for each colour plane (red, green, blue) across the 600 images. Calculating the standard deviation of each pixel in the image created the STD image. These calculations were done in a C dynamic link library (dll) that was called by the capture software written in Visual Basic. This station was later decommissioned in November 1999.

The focus for the South African coastal imaging stations then shifted to Durban where the initial field trials had taken place. Two imaging stations were installed, overlooking the North and Addington beaches. Data from the Addington and North beach cameras were transferred to the Applied Physics Group and hourly images placed on the website http://www.nu.ac.za/physics/coastal/. The locations of these cameras are shown in Figure 2.3.

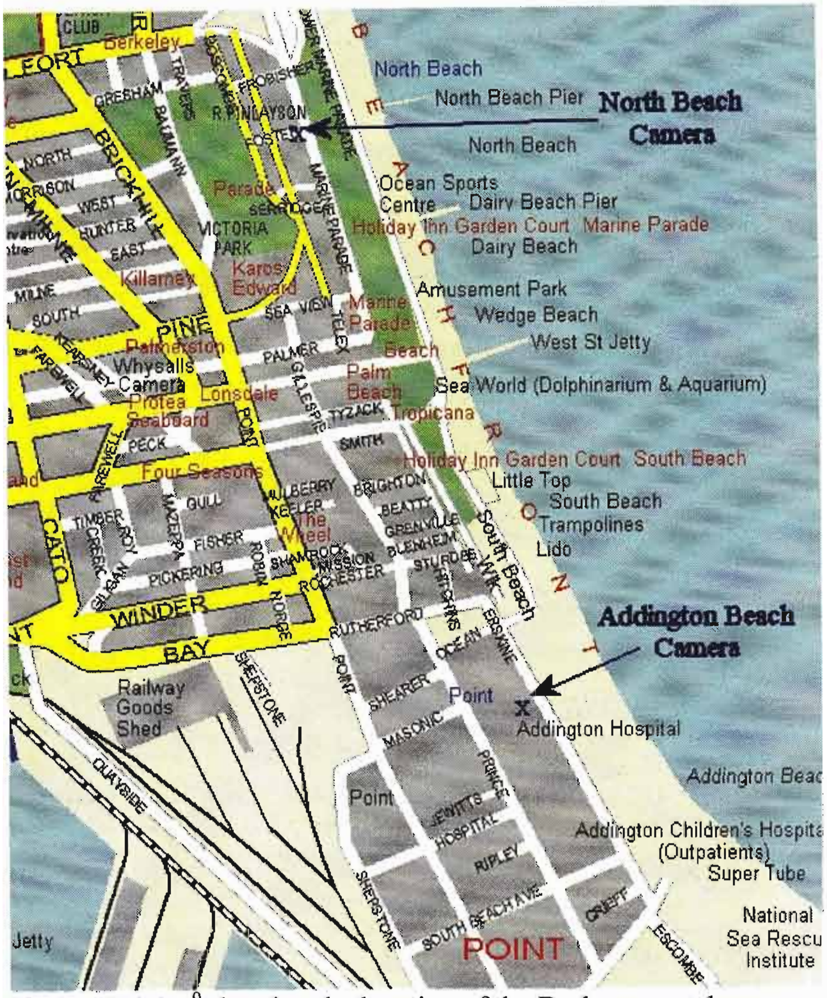


Figure 2.3 Map[9] showing the location of the Durban coastal cameras.

In September 1998 a fixed colour CCD camera and microwave link were installed on Addington Hospital, Addington Beach (Figure 2.4). A live video signal was transmitted back via a 1.2 GHz microwave link to the Desmond Clarence Physics Building at the University of Natal (a distance of ~ 7 km). Images (760 x 576 pixels in size) were then captured on a PC with a Flashbus Pro frame grabber using a similar version of the software that had been previously used at the Melkbosstrand station.

[9] Map courtesy of http://www.easymap.co.za/

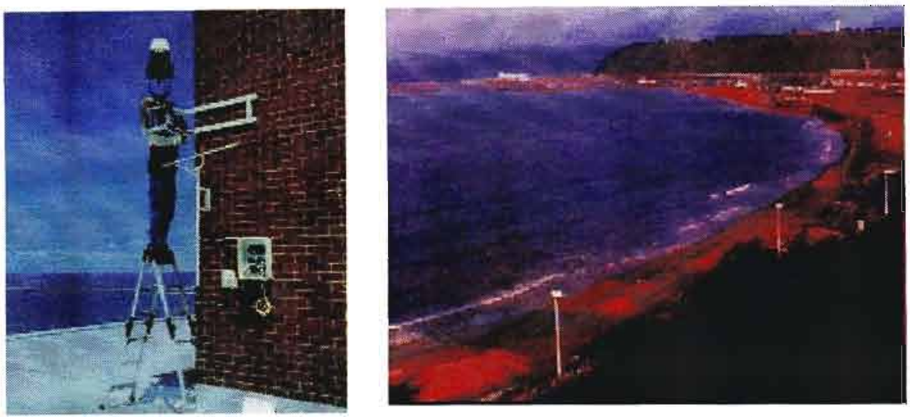


Figure 2.4 Camera installation at Addington Beach and a SNAP image captured using the camera.

In February 1999 a colour CCD camera with pan, tilt and zoom (PTZ) capabilities was installed on the Palace, North Beach (Figure 2.5). This also used the same 1.2 GHz microwave band as the Addington link to send back a live video signal. The two microwave transmitters were time multiplexed using AC time switches so that each would transmit in alternate 30-minute periods, to a single receiver.

The software on the capture PC was modified so that the capture time was reduced to 5 minutes. This allowed ample time for the signals from both transmitters to be captured and the creation and storage of the SNAP, AVG and STD images to be completed within 30 minutes for each station. In addition the hourly time-stamped SNAP images are made available on the University of Natal website (http://www.nu.ac.za/physics/coastal/).

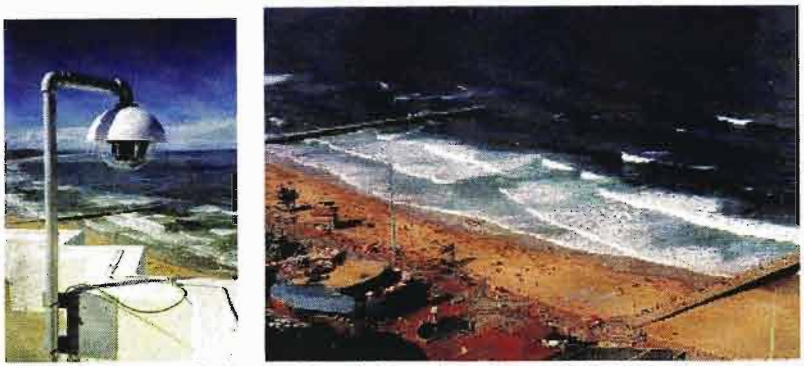


Figure 2.5 Camera installation at North Beach and an a SNAP image captured using the camera

Current research with the Durban coastal imaging stations has involved developing a robust, efficient and accurate technique for extracting waterlines from the captured images. A discussion and comparison of the various techniques developed follows in Chapter 3. The application of the DCIV technique to the surf-zone has also been validated. A discussion with the results of this experiment can be found in Chapter 4.

Chapter 3

WATERLINE EXTRACTION

3.1. Introduction

In this chapter the operational definition of the waterline is given, together with a comparison of the different techniques for waterline extraction, that have been developed. The problems of extracting a reliable waterline is outlined and the strengths and weaknesses of each waterline extraction technique are explained.

The waterline can be simply defined as the interface between the water and the dry sand on the beach. However this interface between the water and sand is a dynamic function of time on the time scale of the wave action, tides and seasons. For this dissertation, the waterline is defined as the average position of the interface, averaged over a number of wave periods, that is, over a time scale, which is larger than the wave period (~5 s) but shorter than the tidal period (~6 hrs). The actual location of the waterline will depend somewhat on the specific technique that is used, and will generally differ from the surveying *low water mark* by an offset.

It is necessary to be able to extract an accurate waterline so that the intertidal beach morphology as well as the effective beach width can then be determined. The effective beach width could be defined as the distance from a fixed reference point on land to the waterline. This is of interest to the Coastal and Drainage Division, Durban Metro, for example, who are responsible for monitoring the accretion and erosion of the Durban beaches and management of the beach nourishment involved in the sand pumping scheme. An example of the measured effective beach width is shown in Figure 3.1. The beach width can only be calculated during the daylight hours hence the plot of the beach width (bottom right of Figure 3.1) exhibits missing data during the hours of darkness.

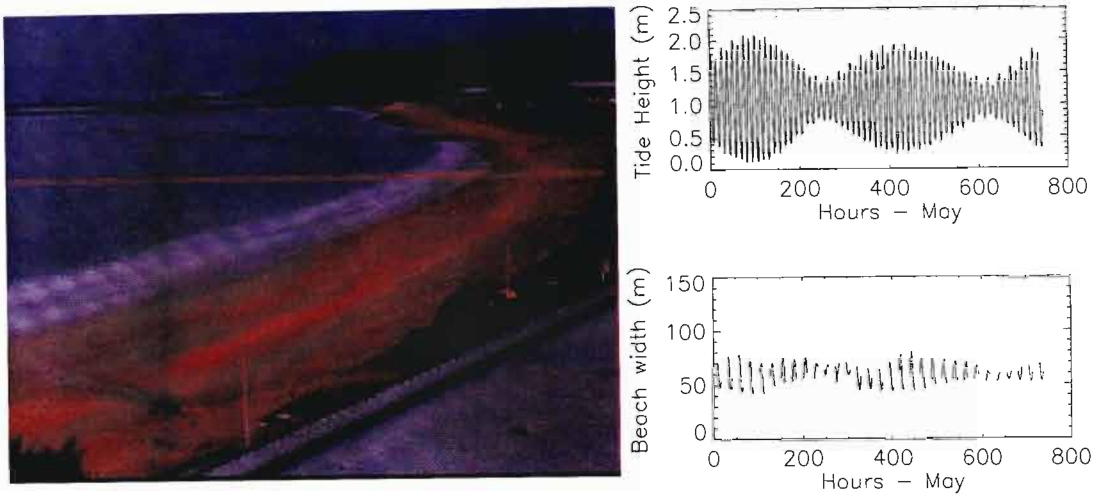


Figure 3.1 An AVG image of Addington Beach (left) with the transect for which the effective beach width was calculated (red). The effective beach width is shown (right bottom) with the tidal influence (right top).

3.2. Technique Description

In this section the criteria for a good waterline technique are discussed together with a description of the various techniques. Waterlines were generated for each technique using the same sequence of images. The comparison of the waterlines calculated using these techniques is given in section 3.3.

The criteria for a good waterline extraction technique are:

1. Robustness: The technique must be resilient to changes in ambient lighting, sea and land colours.
2. Accuracy: Parallel to the coast, the waterline should be displaced from the *low water mark* by a constant offset.
3. Minimal supervision: The application of the technique should be as automatic as possible. It should require a minimal manual choice of parameters, for example threshold values, for each waterline. It would be acceptable to require one set of initial parameters for each imaging station's field of view set-up.

4. Efficiency: The algorithm should be computationally efficient.
5. Well defined: The computed function, which is used to find the waterline, should have a large cross-shore spatial gradient and waterline position should not be ambiguous.

3.2.1. Grayscale Thresholding

This was the first technique to be used, as it was the simplest to understand and implement. It entailed an observation of the variation in the pixel intensity values in the grayscale STD image (Figure 3.2). The grayscale image was calculated by taking the average RGB intensity values of each pixel in the colour image.

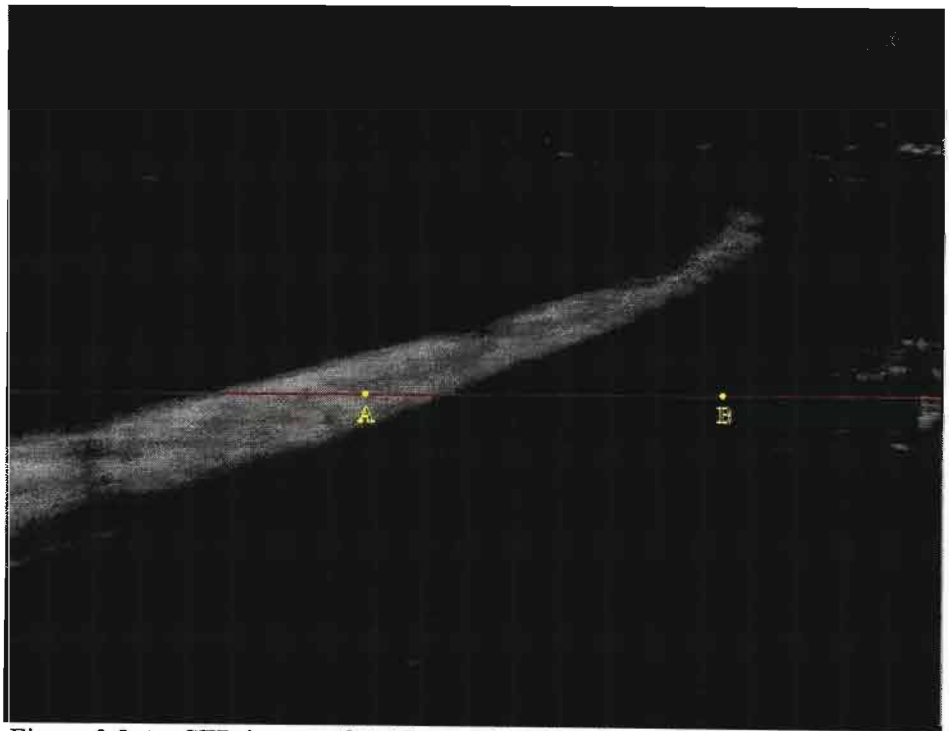


Figure 3.2 An STD image of Addington Beach with a single horizontal line of interest (transect) marked in red. A time series of the pixel intensities at points A and B are shown in Figure 3.3.

The STD images were calculated from the time series of each pixel's grayscale intensity over 300 sequential images. Figure 3.3 shows the temporal evolution of a pixel's grayscale intensity in the surf-zone (black) (position A in Figure 3.2) and on land (red) (position B in Figure 3.2).

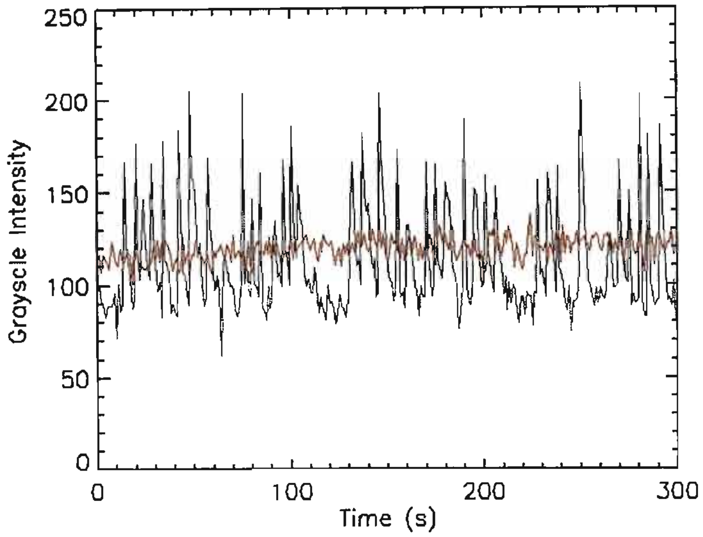


Figure 3.3 Temporal evolution of grayscale intensity in the surf-zone (black) (position A in Figure 3.2) and on land (red) (position B in Figure 3.2).

The grayscale intensity values vary over a larger range (62→209) in the surf-zone than on land (101→138). The more the pixel's grayscale intensity value varies over time, the higher the standard deviation is for that pixel, hence the pixels in the surf-zone will have a larger standard deviation. In Figure 3.3 the pixel in the surf-zone (black) and on land (red) have a standard deviation of 28.13 and 6.07 respectively.

The breaking waves in the surf-zone make the transition between the variable white water and stationary sand stand out noticeably in the STD image as a white band (high standard deviation) in the surf-zone. This is clearly illustrated in Figure 3.4 where the region of breaking waves in the surf-zone (pixels 150→380) exhibits a greater pixel intensity value in the STD image when compared to the regions on land. Other smaller patches on the land also stand out due to moving objects such as ships, cars and people.

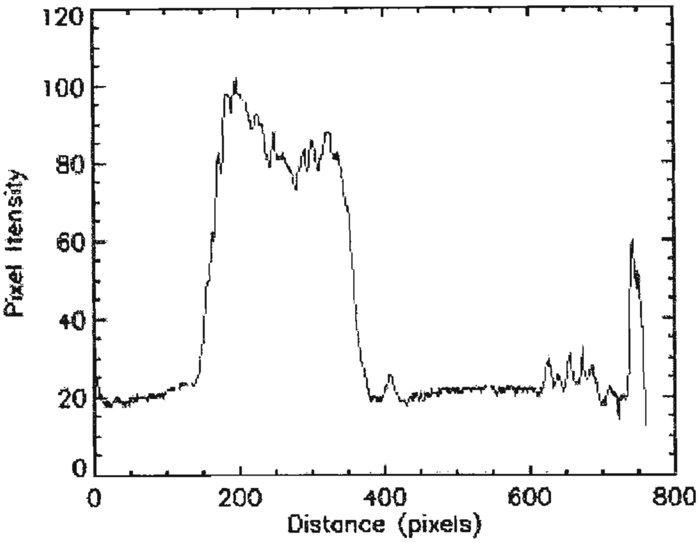


Figure 3.4 A plot of the transect extracted from the image in Figure 3.2. The transition between the water and sand is ~ 400.

The IDL *Contour* routine was used to extract the waterline from the STD image. The *Contour* routine was passed a value (threshold value) and the contour with the largest number of elements for that value was extracted. The threshold value had to be chosen very carefully to work properly. For the STD image in Figure 3.2 a value of 23 was chosen as the plot in Figure 3.4 indicates that the grayscale intensity values for the region of breaking waves all had a value ≥ 23. The results of the contouring can be seen in Figure 3.5.

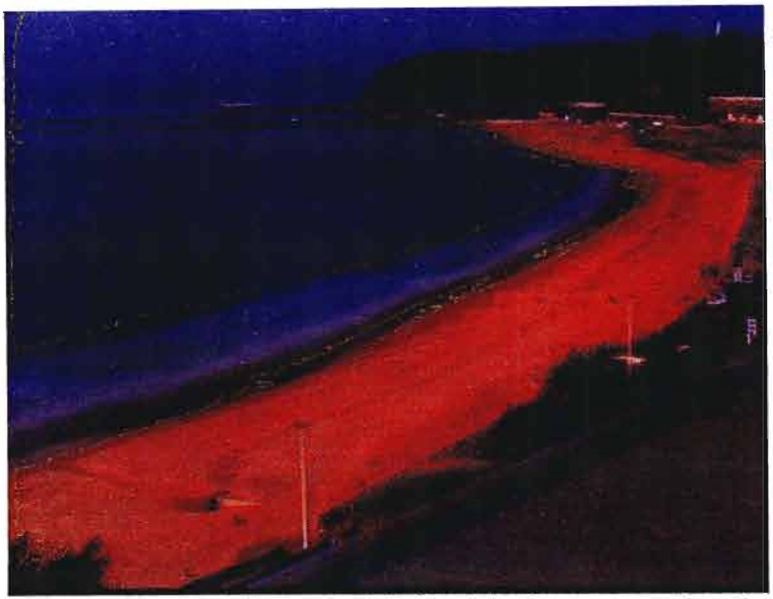


Figure 3.5 AVG image with a contour (yellow) having a grayscale value of 23.

The varying ambient light intensity during the day also made it necessary for a new threshold value to be chosen for each hour. In addition the changing lighting conditions due to cloud movement has made automation of this technique difficult. Patchy clouds moving over the recorded scene change the ambient lighting conditions hence reducing the contrast between the white water and the rest of the image. This produces images that are difficult to analyse as structures out of the surf-zone are also enhanced making contouring difficult. This can be clearly seen in Figure 3.6 where the beach grayscale intensity is also significantly enhanced.

This phenomenon was also observed by Janssen [23]. A plot of the same transect used in Figure 3.4 but for the STD image shown in Figure 3.7 does not give a clear distinction of the interface between the water and sand although visually it can be inferred to be at a pixel distance of ~ 470. The contouring routine has problems with images such as the one shown in Figure 3.6.

Figure 3.6 STD image of Addington Beach (09/10/2000) exhibiting the problems associated with irregular ambient lighting conditions. The beach is significantly enhanced.

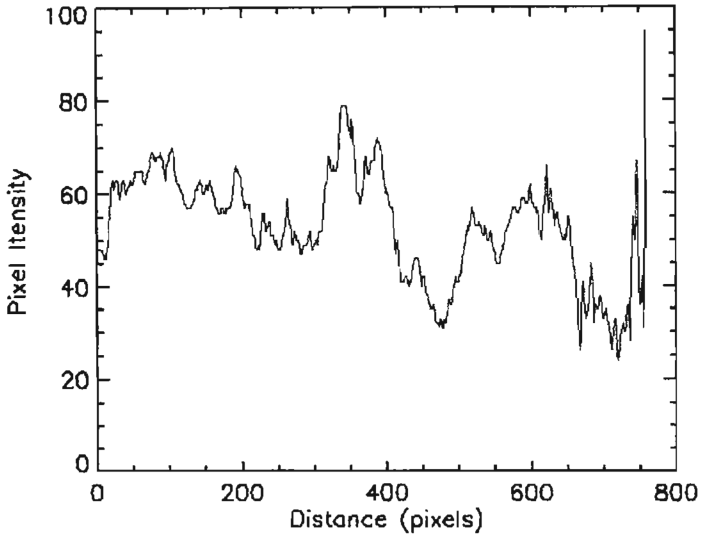


Figure 3.7 An example of the grayscale intensity plot along the same transect used in Figure 3.4 but for the STD image shown in Figure 3.6. Although the interface of the water and sand is ~470 the contouring routine has problems since the beach (located at 480→650) is also enhanced.

3.2.2. δ Colour Discriminator

The δ colour discriminator proposed by Basson (Alport et al.[6], Basson [7]) is another thresholding method that uses the colour information of a pixel instead of just its grayscale value. In order to obtain a single parameter that describes a pixel's colour information the RGB (red, green, blue) colour space is converted to the HLS (hue, lightness, saturation) colour space. A discussion of the RGB and HSL colour models and equations used to convert between them can be found in APPENDIX B.

A pixel's colour (hue) information is represented by its azimuthal value on a colour wheel where colours are mapped onto a set of values ranging from 0 to 360. The sea is blue (H ~ 240) while the sand is reddish brown (H ~ 0). H is cyclic, that is the colour of a pixel with H = 0 is the same as the colour of a pixel with H = 360.

The parameter δ was defined as

$$\delta = (H - \delta_s) \bmod 360 \qquad [3.1]$$

The shift parameter, δ_s, was introduced to ensure that the colour of the regions of interest, that is water and sand, did not straddle the boundary between the hue values of 360 (maximum) and 0 (minimum) due to the cyclic nature definition of the hue values.

The location of the red and blue colours in the hue plane was examined as the water is blueish and the beach redish. This can be seen Figure 3.1 where red can be seen to influence hue values from 0 to 120 and then again from 240 to 360 due to the cyclic nature of the hue plane. Blue can be seen to influence hue values from 120 to 240. Red and blue both contribute to the hue values between 240 and 360.

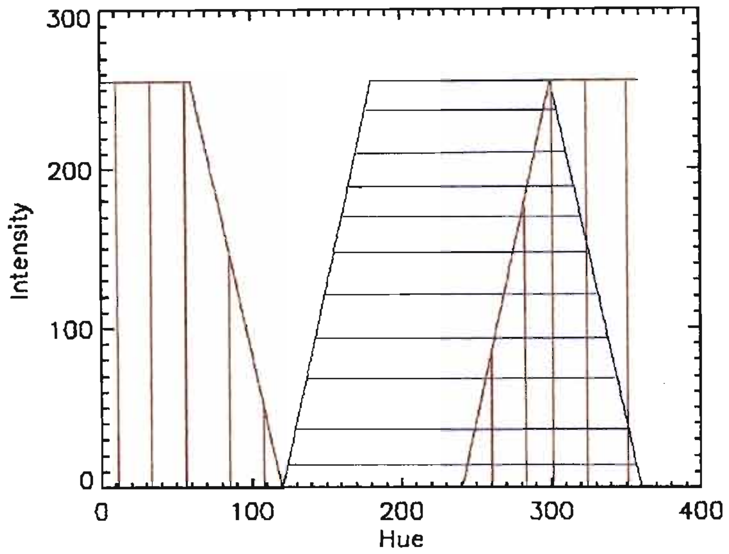


Figure 3.8 Plot showing the location of the colours blue and red in the hue plane. Red can be seen to influence hue values from 0 to 120 and then again from 240 to 360 due to the cyclic nature of the hue plane. Red and blue both contribute to the hue values between 240 and 360.

The distribution of the land (redish) pixels in the regions H = 0→120 and H = 240→ 360 can be seen in the Hue-Saturation (HS) plot of the AVG image in Figure 3.9. The HS plot is obtained by calculating the cumulative frequency of a pixel's H and S value and plotting this as a two-dimensional surface. The pixels corresponding to the land straddle the hue values 360 and 0. The clustering of *wet* (water) and *dry* (land) pixels in the HS plot was first noticed by Aarninkhof and Roelvink [4].

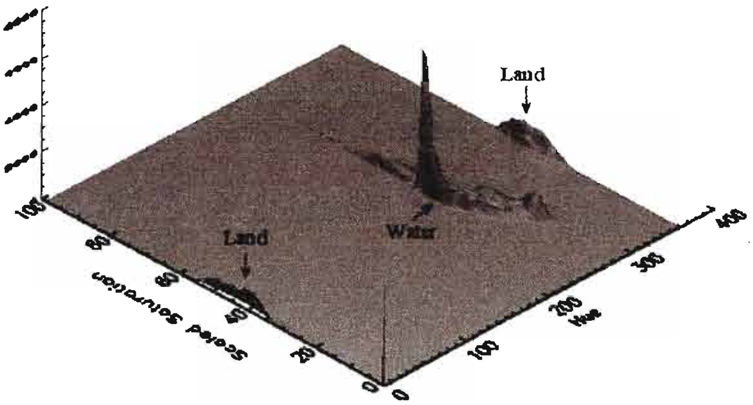


Figure 3.9 Hue-Saturation plot for the AVG image in Figure 3.5.

In order to obtain a single continuous parameter the hue values were shifted by a value of $\delta_s = 120$ so the location of the colour red was concentrated from 120 to 360 in the hue plane. The effect of this shifting on the regions influenced by red and blue colours in the hue plane can be seen in Figure 3.10 while the effect on the HS plot can be seen in Figure 3. 11.

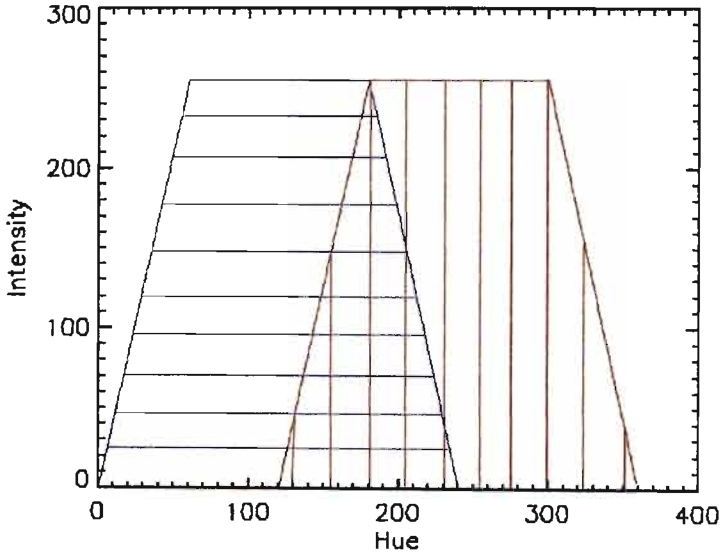
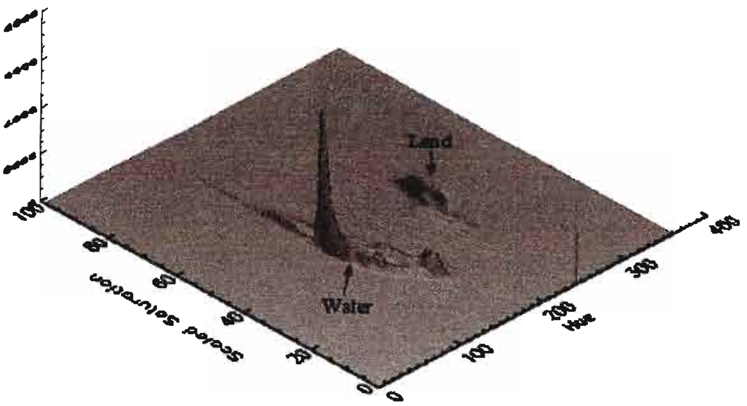


Figure 3.10 Plot showing the location of the colours blue and red in the hue plane after a cyclic shift using $\delta_s = 120$. The influence of the colour red is now



concentrated from 120 to 360.

Figure 3. 11 Hue-Saturation plot for the AVG image in Figure 3.5 after a cyclic shift using $\delta_s = 120$ on the hue values.

The need for the cyclic shift of using $\delta_s = 120$ is clearly seen in Figure 3.12 where the plot of the original H values (black) show large discontinuities due to the straddling of the hue values of 360 and 0. The plot of the δ values is continuous. Application of the $\delta_s = 120$ shift on the hue values of the AVG image can be seen in Figure 3.13.

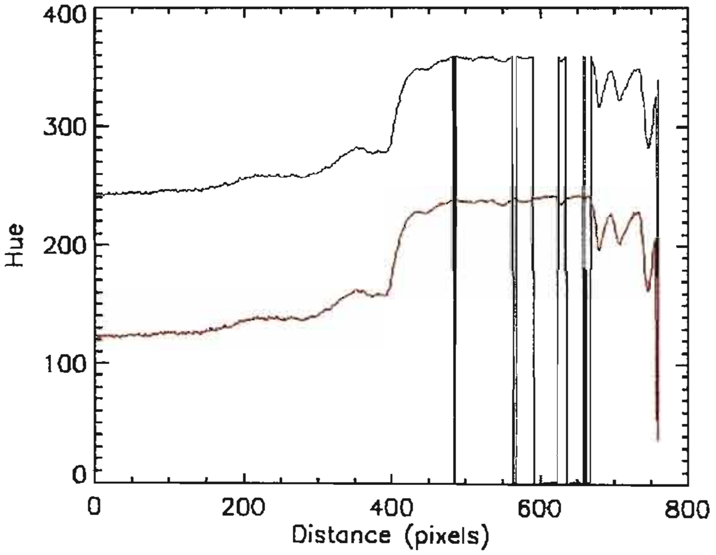


Figure 3.12 Plot of the transect marked in Figure 3.2 for the original hue values (black) and after shifting the hue values using $\delta_s = 120$. The shifted hue transect does not have any large discontinuities.

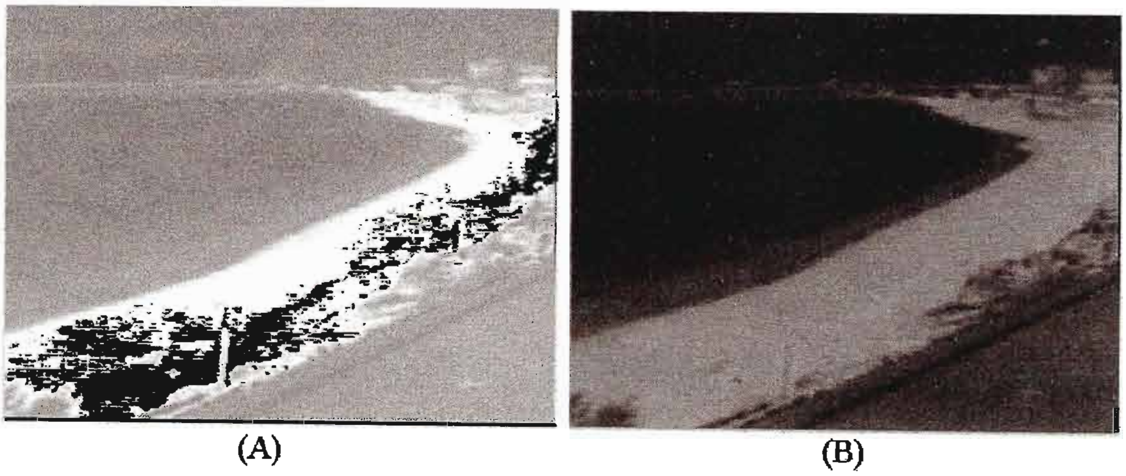


Figure 3.13 (A) hue plane of the AVG image and (B) δ plane of the AVG image after a cyclic shift of the hue values by $\delta_s = 120$.

The waterline was extracted from the image in Figure 3.13 (B) using grayscale thresholding with a value of 125. This waterline can be seen superimposed on the AVG image in Figure 3.14.

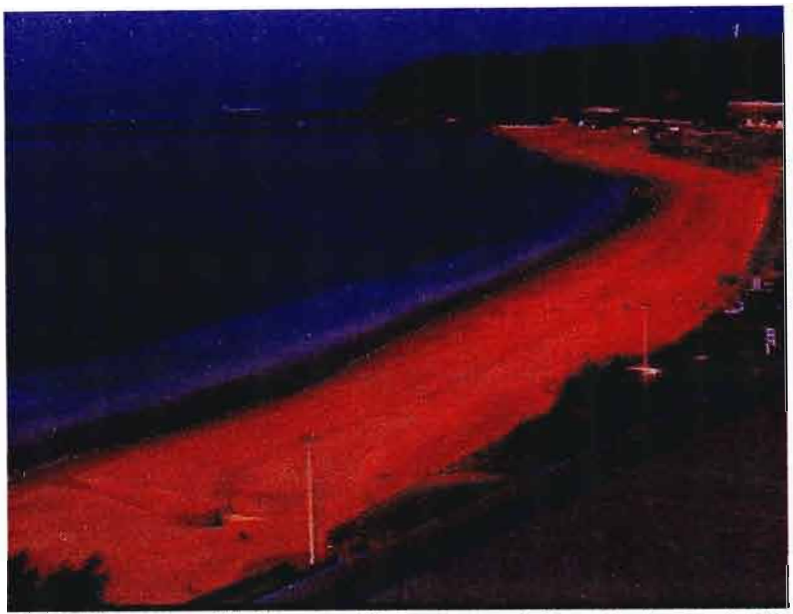


Figure 3.14 AVG image with the waterline, extracted from the image in Figure 3.13 (B), superimposed on it.

In this technique the colour (hue) information in the image was used to extract the waterline. The optimal value for the cyclic shift parameter δ_s was found to be 120. Further investigation is necessary to observe the performance of this modified technique under changing ambient lighting conditions.

3.2.3. Artificial Neural Network (ANN)

It is shown in Figure 3.9 that distinct *wet* and *dry* pixel clusters exist in the HS cumulative histograms of surf-zone images. This clustering of *wet* and *dry* pixels suggests that an Artificial Neural Network (ANN) might be able to classify these features. For the theory and implementation of ANNs see APPENDIX C.

Basson [7] used a Multi-layer Perceptron (MLP) with one hidden layer for his research. The MLP is a *supervised learning, feedforward* neural network that performs very well in classifying, mapping and association. A commercial software package *Neurosolutions*[10] was used to create the MLP (Figure 3.15).

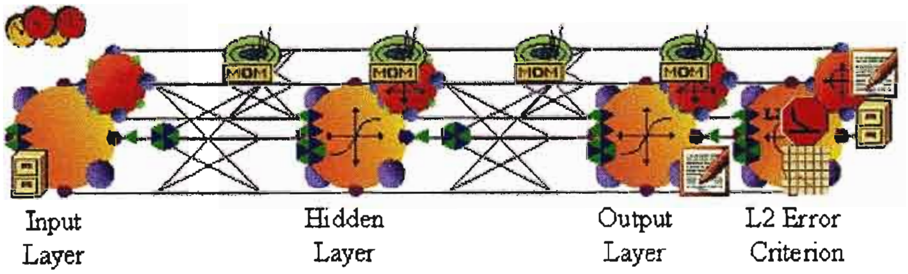


Figure 3.15 MLP created for waterline discrimination.

The MLP first needs to be trained. Training is achieved by feeding the training data set in at the input layer. This data is then propagated forward through the synapses joining the input and hidden layer and then the synapses joining the hidden and output layer. At each stage the activation function is applied to the data. The actual results at the output layer are compared to the desired results and the difference back propagated and the weight adjusted.

Each set of data values is fed into the MLP for a number of cycles (epochs) until the required error criterion being used (usually Mean Square Error) has been satisfied. At this point, the MLP is then considered to have been trained. Then the data set that needs to be classified is presented to the MLP and the results recorded.

[10] http://www.nd.com

Basson used a training data set that included a number of images captured throughout the day. The MLP was trained with 3000 exemplars (instances) from these images. Each exemplar consisted of the three RGB values of each pixel, which was classified as being *wet* or *dry*. A mean square error (MSE) of 0.0231 was attained. A very similar MSE value (0.0259) was obtained when the pixels HSL values were used instead of their RGB values. Basson concluded that this result was expected, as the mapping from RGB to HSL is linear. The ANN was show to perform better than the grayscale thresholding and δ colour discriminator techniques (2.3.7).

In this dissertation the same MLP that was used by Basson was created using *Neurosolutions*. The MLP was first trained with a data set of 740 RGB exemplars (three inputs) and a MSE of 0.0626 was achieved after 16500 training cycles (epochs). Figure 3.16 shows the output of the MLP classification for the AVG image shown in Figure 3.5. The different regions of surf-zone images were classified as water (white) or land (black). The waterline can be easily seen in the image at the white/black boundary. However, there are other significant regions particularly on the hospital roof in the bottom right hand corner of the image, which are incorrectly classified as being wet.

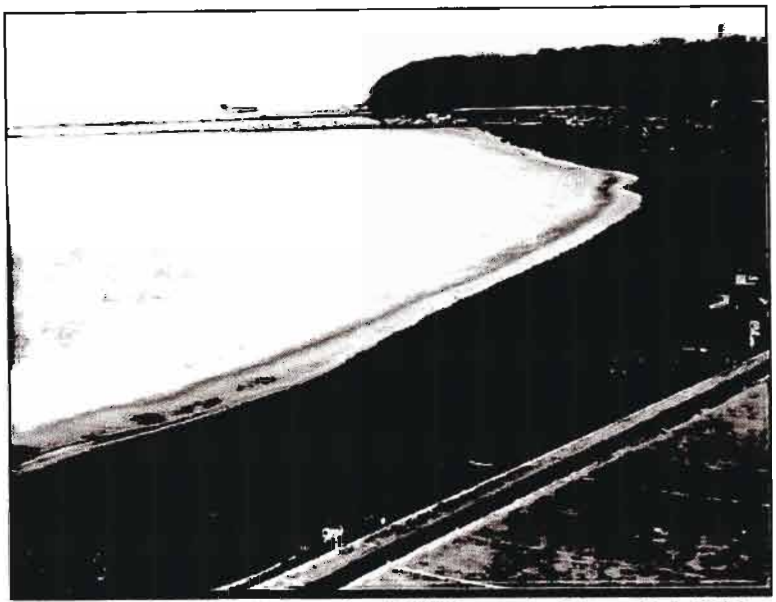


Figure 3.16 Result of RGB MLP on the AVG image in Figure 3.5. White indicates water and black indicates land. The roof of the hospital (bottom right) is blue and hence misclassified as water although it was presented to the MLP as land.

Another MLP was then created and trained with the average and standard deviation values for each of the RGB planes from the respective images in addition to the 740 RGB values. This resulted in nine input values to the MLP. The average of each of the RGB planes was used to provide non-local data about the hourly variation in the ambient lighting conditions. The standard deviation for each of the RGB planes was used to provide non-local data about the distribution of each colour in the image in an attempt to overcome the problems in lighting conditions as a result of patchy clouds moving over the scene. A MSE of 0.0285 was achieved after training for 16500 epochs. The results of classifying the different regions as either water or land on the AVG image in Figure 3.5 can be seen in Figure 3.17.

Figure 3.17 shows better classification of the roof of Addington Hospital (smaller white region) when compared to Figure 3.16. In Figure 3.17 the classification of water at the bottom left of the image is not as well defined as Figure 3.16. This discrepancy can be explained by this region being classified as land (black) due to the similar colour of the roof (land) and the water. The waterline extracted from Figure 3.17 can be seen superimposed on the AVG image in Figure 3.18.

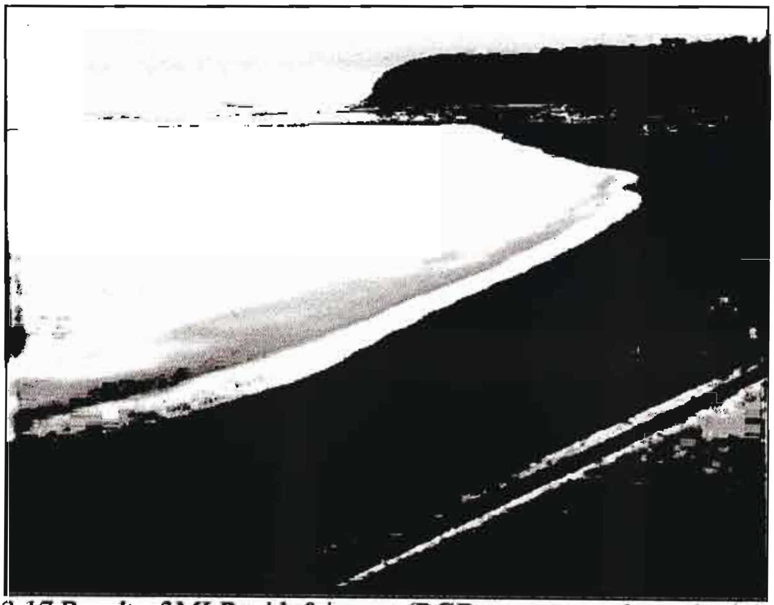


Figure 3.17 Result of MLP with 9 inputs (RGB, average and standard deviation values) on the AVG image in Figure 3.5. White indicates water and black indicates land.

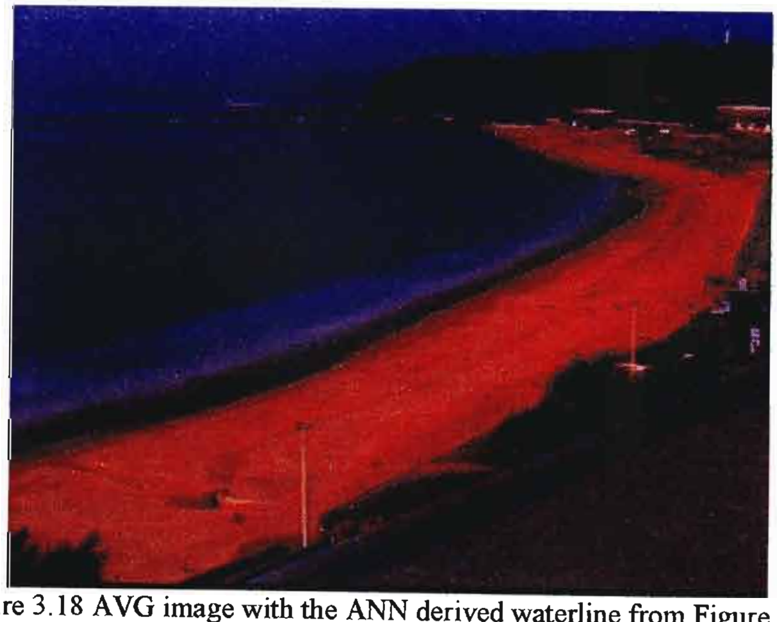


Figure 3.18 AVG image with the ANN derived waterline from Figure 3.17 superimposed in yellow.

It has been shown that a MLP with non-local data (average and standard deviation values in addition to RGB values) as inputs does train better (MSE = 0.0285) for the same number of epochs, when compared to a MLP with just RGB values as inputs (MSE = 0.0626). Further investigation is necessary to find the optimal set of values to use as inputs.

3.2.4. Wavelet Analysis

The δ colour discriminator and the ANN techniques discussed above use the colour information to better discriminate between *wet* and *dry* pixels. The waves breaking in the nearshore have temporal characteristics, which was partially exploited by creating the STD images (see section 3.2.1).

However, the temporal and spatial scale size characteristics of a sequence of images are additional features that can be used to distinguish between pixels located on land and pixels located in the water. Specifically, wave-fronts that propagate through the surf-zone consist of elongated lines of foam having characteristic widths parallel and perpendicular to the shore. These additional spatial and temporal characteristics can be used to more unambiguously identify the surf-zone in an image. The waterline would then be the shoreward boundary of this region.

The temporal evolution of a pixel's grayscale intensity in the surf-zone (black) and on land (red) in Figure 3.19 suggests that the periodic nature of the wave action causes those pixels in the surf-zone to have a spectral structure, which is different from that of pixels on the land. Wavelet analysis (APPENDIX E) can be used to exploit these differences in spatial scales. Fourier analysis (APPENDIX D) will be used to exploit the temporal scale, that is the spectra.

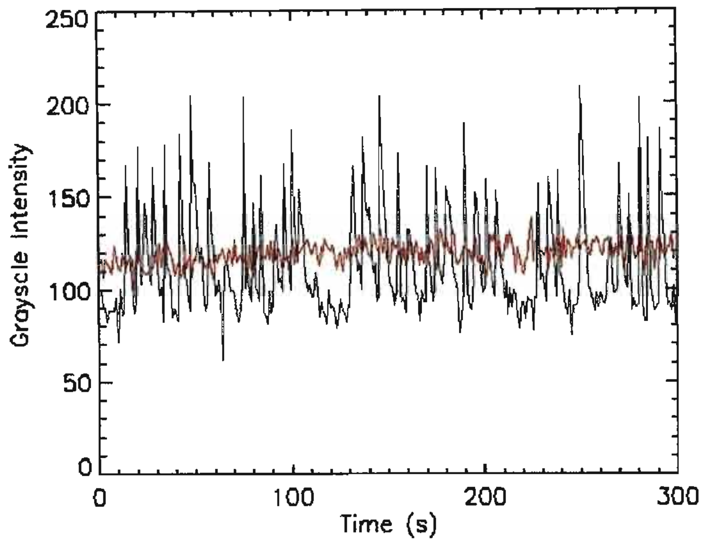


Figure 3.19 Time series of grayscale intensity in the surf-zone (black) (position A in Figure 3.2) and on land (red) (position B in Figure 3.2).

The *magnitude* of the Wavelet Transform (WT) of a one-dimensional time series is a two-dimensional function, which may be represented as an image where the temporal scale size is on the *y*-axis and the shift *T* on the *x*-axis. The Wavelet representation of the time series in Figure 3.19 is shown in the images in Figure 3.20. Lighter regions indicate a higher value at that particular scale and shift. Figure 3.20 used a Morlet wavelet basis of order 3 as it is well localised in the frequency domain (Costa et al. [12]).

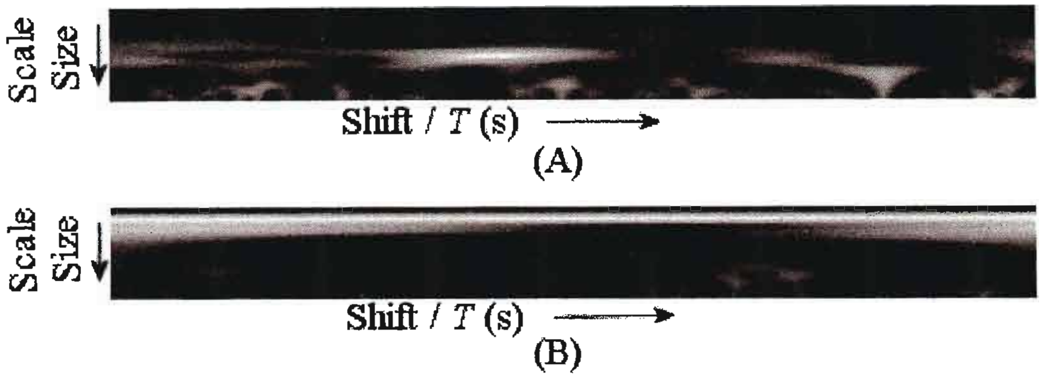


Figure 3.20 (A) WT magnitude for the time series of a pixel in the surf-zone and (B) WT of the surf-zone time series for the lines shown in Figure 3.19. Lighter (white) regions indicate a higher value.

By summing the rows in Figure 3.20 (that is projecting onto the *y*-axis) the contribution to each time scale can be seen Figure 3.21. In Figure 3.21 it can be observed that the scale size in the surf-zone peaks in two bands centred at ~ 25s and ~ 165s.

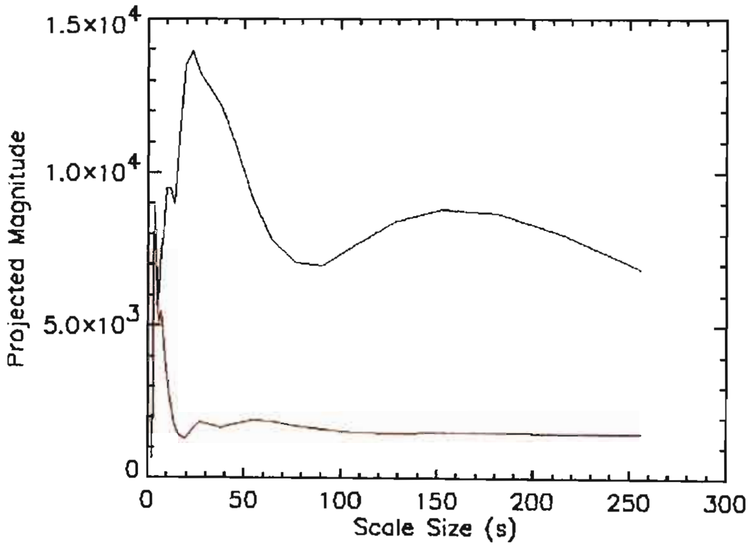


Figure 3.21 The scale size spectrum in surf-zone (black) and out of the surf-zone (red) calculated using the wavelet transforms shown in Figure 3.20.

Once the predominant scale sizes (19→256) in the surf-zone time series had been identified, the *magnitudes* of the WT of the time series in Figure 3.19 were filtered. This filtering was achieved by integration over the amplitudes of only the selected scale sizes (scale selection – see Costa et al. [12]) for each shift *T* in the *magnitude* of the WT. The integration over the selected amplitudes reduces the two-dimensional WT *magnitude* (WTM) to a one-dimensional filtered time series. The result of this filtering is shown in Figure 3.22.

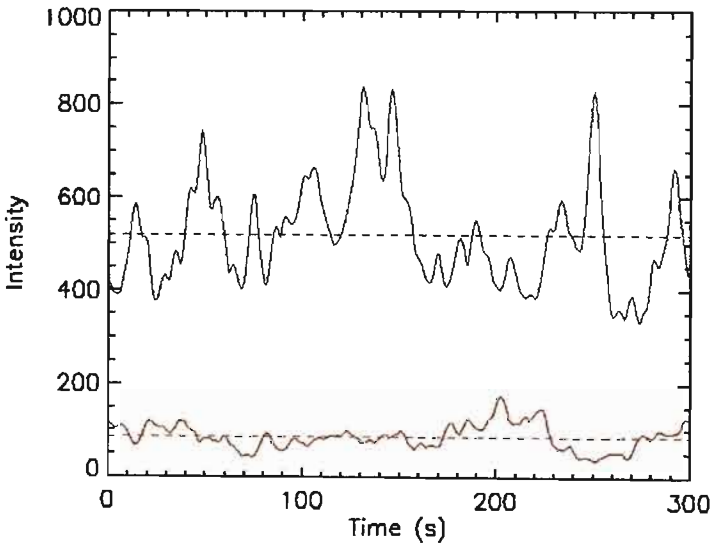


Figure 3.22 Plots of the filtered WTM using scale selection for a time series in the surf-zone (black) and on land (red) that are shown in Figure 3.19. The dashed black and red lines show the average values of these two series.

The average of the filtered time series was then calculated to return a single value, which would be larger for pixels in the surf-zone than on land. This difference is shown in Figure 3.22, where it can be observed that the average pixel intensity of the filtered time series for a pixel on land (86.50) is significantly less that of the pixels in the surf-zone (520.13).

Application of this wavelet filtering and averaging to the time series of every pixel along the transect shown in Figure 3.2 results in the filtered transect shown in Figure 3.23 (black).

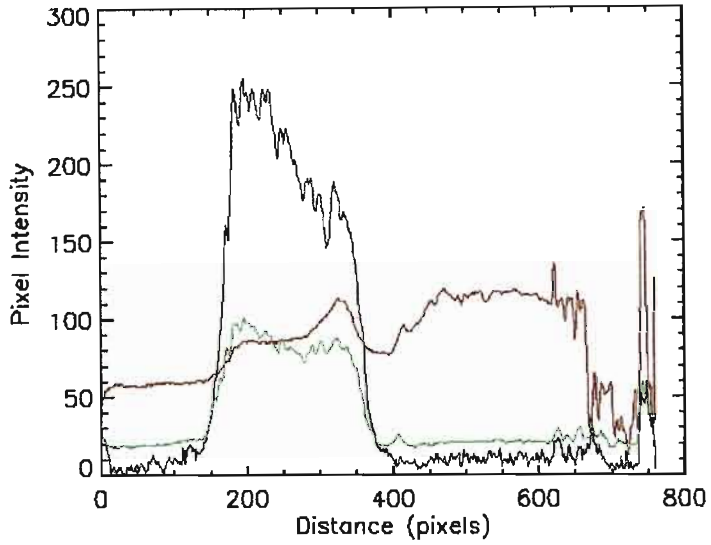


Figure 3.23 The result of applying the WTM filtering and averaging to all pixels along a transect (black), together with the same transect extracted from the AVG (red) and STD (green) images.

This procedure, which was then be applied to all the pixels in a sequence of images, resulted in the image shown in Figure 3.24. The estimated waterline was then obtained by applying a greyscale threshold of 28 and is shown superimposed on the AVG image in Figure 3.25.

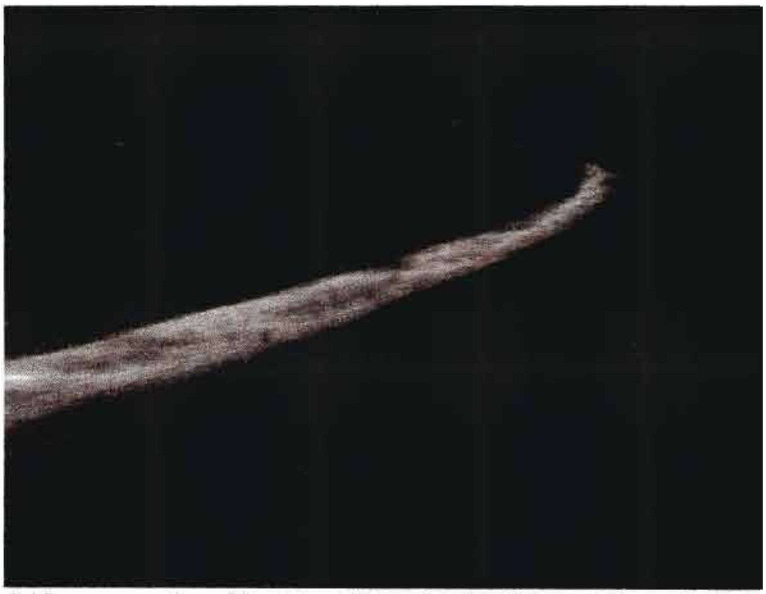


Figure 3.24 Image produced by Wavelet analysis, after applying a band pass filter with scale lengths (19 – 256)s.

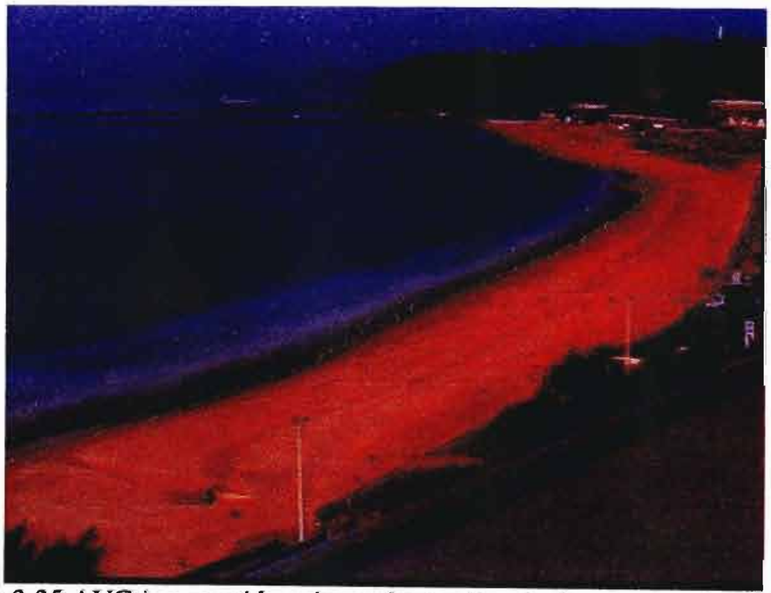


Figure 3.25 AVG image with estimated waterline (yellow) using the Wavelet analysis technique superimposed on it.

This technique is computationally intensive and takes a long period of time (approximately 7 hours) to calculate a single waterline. However, this time could be significantly reduced if the WT was implemented in hardware using an FPGA (Field Programmable Gate Array), but the design and construction of such a board was considered to be outside the scope of this thesis.

It has been shown that the spectral characteristics of the waves can be further exploited to identify the surf-zone. This is in contrast to using the standard deviation, which is a broad-spectrum description of the pixel variations. It should be noted that the images shown in Figure 3.25 are not ideal for application of a scale size discriminator since features in the surf-zone are reduced in scale due to the perspective effect towards the top right hand corner. This makes it difficult to select a unique spatial scale size that characterises the surf-zone. In addition the horizontal transect does not traverse the surf-zone at right angles throughout the image.

3.2.5. Fourier Analysis

It is evident from the above Wavelet analysis results that the time varying nature of the wave action in the surf can also be exploited. The spectrum specific characteristics of the surf-zone can also be exploited using the Fast Fourier Transform (FFT). It is possible to selectively filter the frequency bands that are predominantly localised in the surf-zone.

Figure 3.26 show a keogram for the transect shown in Figure 3.2. Two time series denoted A and B in Figure 3.26 are located in the surf-zone whilst C is located on land. The FFT amplitude spectrum for the time series on land was computed and is shown in Figure 3.27. The FFT amplitude spectra for the time series in the surf-zone denoted A and B in Figure 3.26 shown in Figure 3.28 and Figure 3.29 respectively.

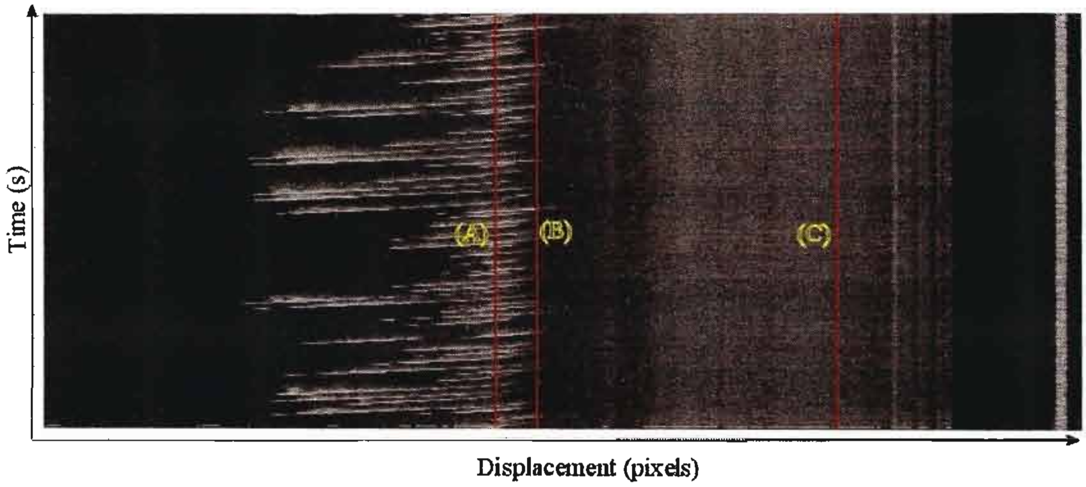


Figure 3.26 Keogram for the transect marked in Figure 3.2 showing the two time series in the in the surf zone (A, B) and on one land (C) that were analysed.

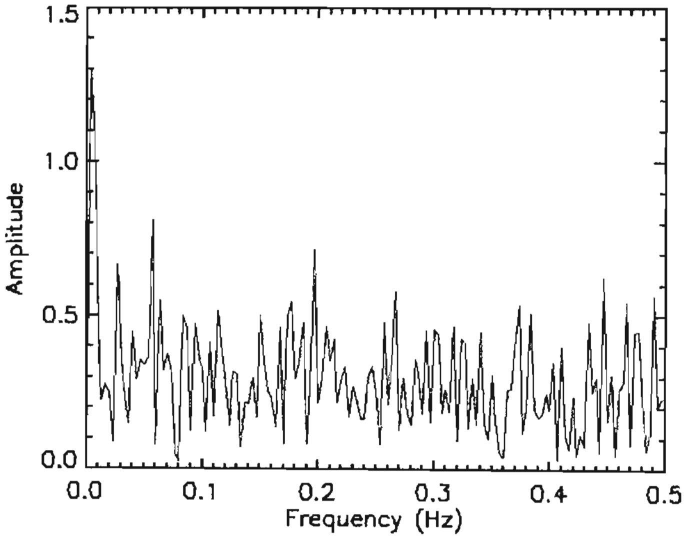


Figure 3.27 Plot of the FFT amplitude spectrum of pixel the intensity located on land (C) in Figure 3.26.

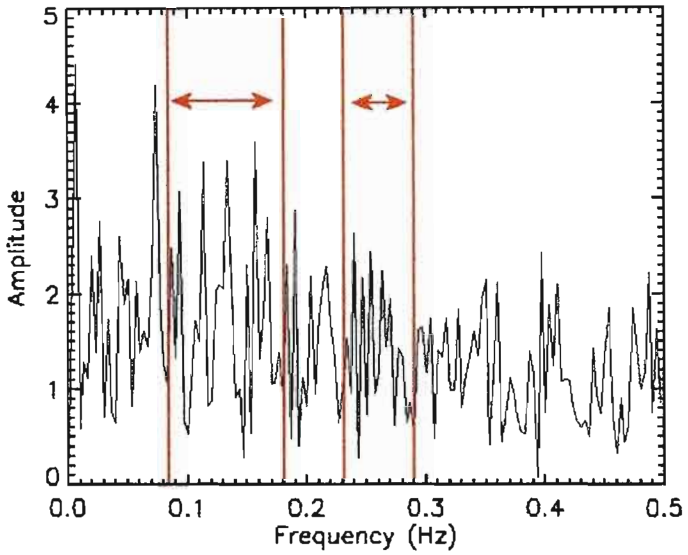


Figure 3.28 Plot of the FFT amplitude spectrum of the pixel intensity located on in the surf-zone (A) in Figure 3.26. The red lines enclose frequency bands, which are more dominant in the surf zone than those shown in Figure 3.27.

From Figure 3.28 it can be seen that frequency bands marked with the arrows are more dominant in the surf zone than on land (Figure 3.27).

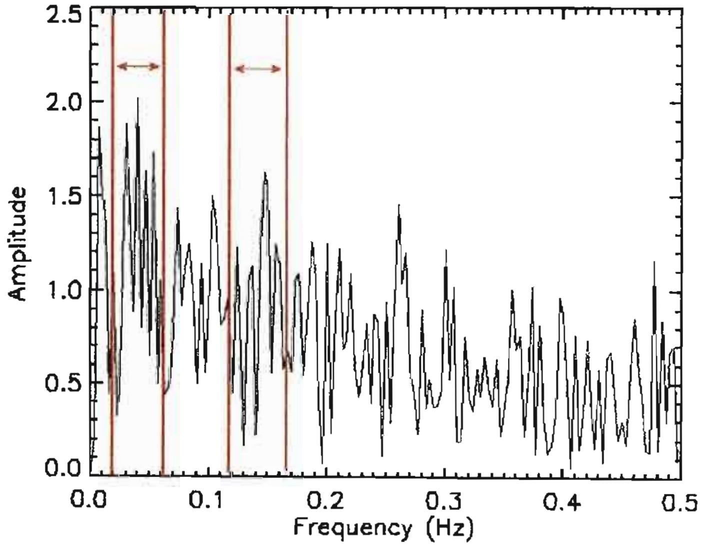


Figure 3.29 Plot of the FFT amplitude spectrum of the pixel intensity located on in the surf-zone (B) in Figure 3.26. The red lines enclose frequency bands, which are more dominant in the surf zone than those shown in Figure 3.27.

In Figure 3.29 it can be seen that the dominant frequency bands, again marked with arrows, occur at slightly lower frequencies at point B in Figure 3.26. This was expected as the waves pass a point higher up on the beach (position B in Figure 3.26) less frequently than a point lower on the beach (position A in Figure 3.26). For clarity, the spectra have been smoothed with a six point moving average and shown in Figure 3.30 where it is clearly evident that there are distinct bands marked by arrows that are present in the surf-zone but not on land.

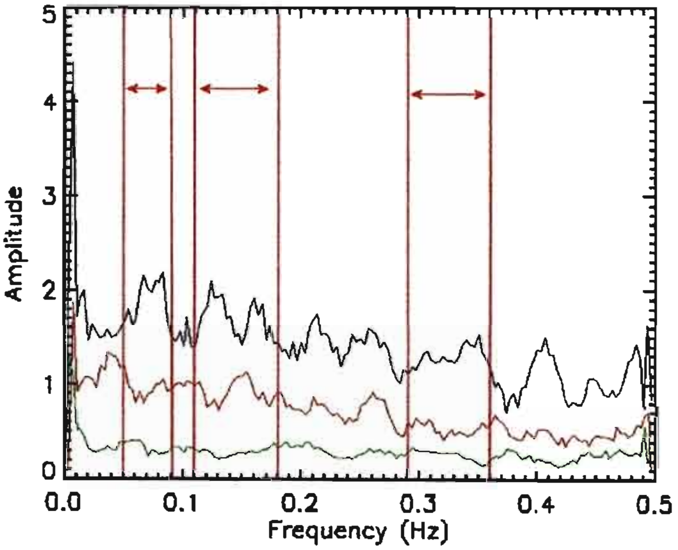


Figure 3.30 Plot of the FFT amplitude spectrum (using a six point moving average) of pixel intensity located on in the surf-zone at location A (black), location B red and on land at location (C) (green) in Figure 3.26.

After analysing all the time series in the surf-zone in the same manner, the frequency bands from (0.003 to 0.05) Hz and (0.15 to 0.35) Hz were selected, as they were the preferentially enhanced in the surf-zone. The amplitude spectrum of each pixel along the transect was integrated over these two frequency bands and plotted as the black line in Figure 3.31.

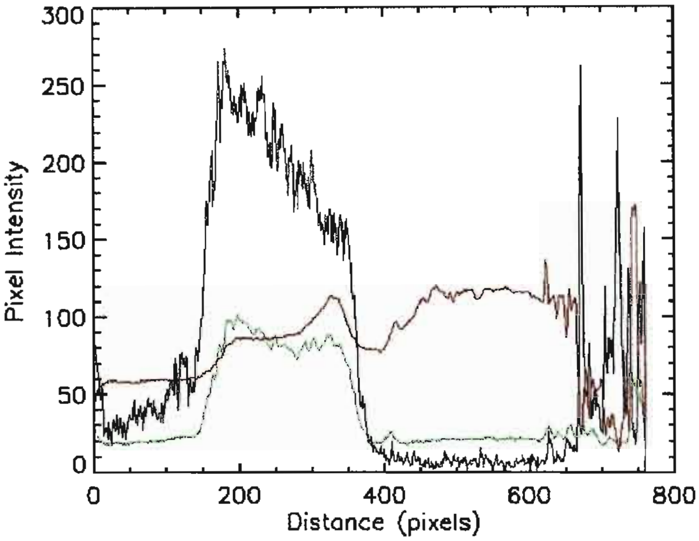


Figure 3.31 FFT amplitude filtered transect (black) compared to the same transect extracted from the AVG (red) and STD (green) images.

The same approach was applied to all horizontal transects in the image and the resulting reconstructed image is shown in Figure 3.32. A greyscale threshold of 15 was used for contouring the extracted waterline is shown superimposed on the AVG image in Figure 3.33.

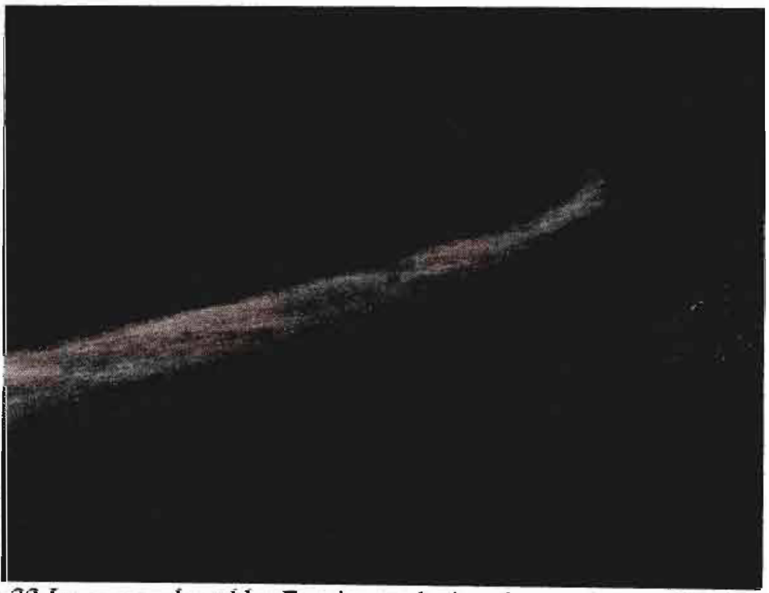


Figure 3.32 Image produced by Fourier analysis, after applying a band pass filter localised at (0.003 to 0.05) Hz and (0.15 to 0.35) Hz.

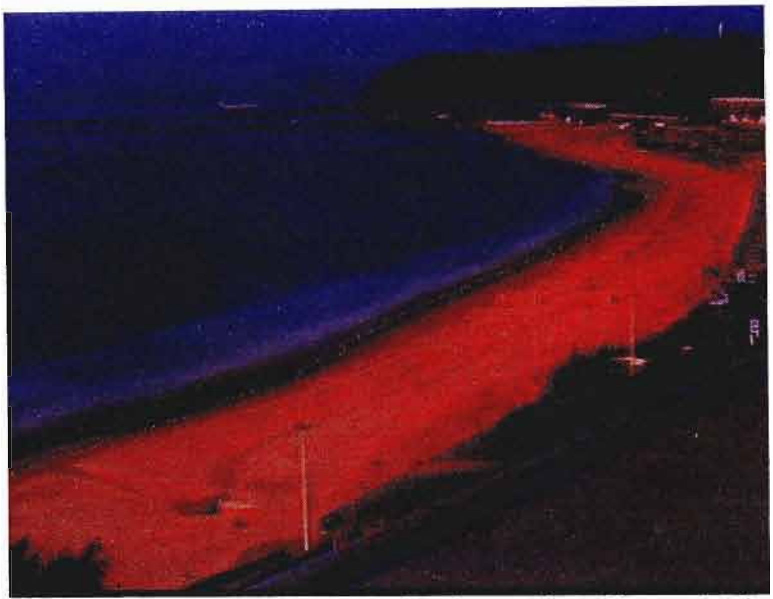


Figure 3.33 Extracted waterline from the band pass filtered image in Figure 3.32 superimposed on AVG image.

Once again using the FFT, it has been shown the temporal characteristics of the waves can be exploited. This technique takes approximately 8 minutes to produce a waterline.

3.3. Comparison and Conclusion

A comparison of the different techniques should consider their robustness, efficiency and accuracy as mentioned before. Unfortunately an assessment of the algorithm's robustness requires long testing periods and a variety of conditions that is beyond the scope of this dissertation.

In considering the efficiency of the different techniques a comparison of the time taken for each technique to extract a waterline is given in Table 3.1.

Technique	Speed
Grayscale Thresholding	5.2 seconds
δ Colour Discriminator	6.4 seconds
Artificial Neural Network	75.7 seconds
Wavelet Analysis	7 hours
Fourier Analysis	8 minutes

Table 3.1 Comparison of time taken by the various waterline extracting techniques.

It is seen that the grayscale thresholding and the δ colour discriminator take approximately the same time. The ANN takes ~ 10 times longer. The Fourier analysis takes 8 minutes, which is acceptable. The Wavelet technique currently implemented takes an unacceptably long time (~7 hours). If the Wavelet transform is implemented so that it returns only scale sizes that are relevant for the filtering of the transects in the Wavelet technique there would be less time taken. In this thesis only 16 of the 29 scale sizes in the WT magnitude used. This would reduce the time taken from approximately 7 hours to 3.8 hours.

Hardware implementation of the Wavelet and Fourier analysis techniques using bandpass filters instead of a software implementation would also make it much faster. A sequence of images captured for 5 minutes at 1 fps (300 in total) requires approximately 360 Megabytes (MBs) of storage space so an online implementation of the Wavelet and Fourier techniques would be required since transferring or storage of this large data set every hour would be difficult.

A measure of accuracy may be obtained by calculating the mean and standard deviation (σ) of the difference between each position for the manually selected waterline and the waterline estimated using the various techniques along horizontal transects. A comparison with a waterline extracted from survey data (see Figure 3.34) using the 0.7m contour level is also given. The data used for the comparison was a sequence of 300 images captured on the day the survey was done (8/6/2000). Only the area spanned by the survey was used for the comparison.

The principal goal was to observe if the extracted waterline was at a constant offset from the manually select waterline. The standard deviation is used as an indication of the spread about the mean difference between the extracted waterlines and the manually selected case – a lower standard deviation indicates a more reliably extracted waterline.

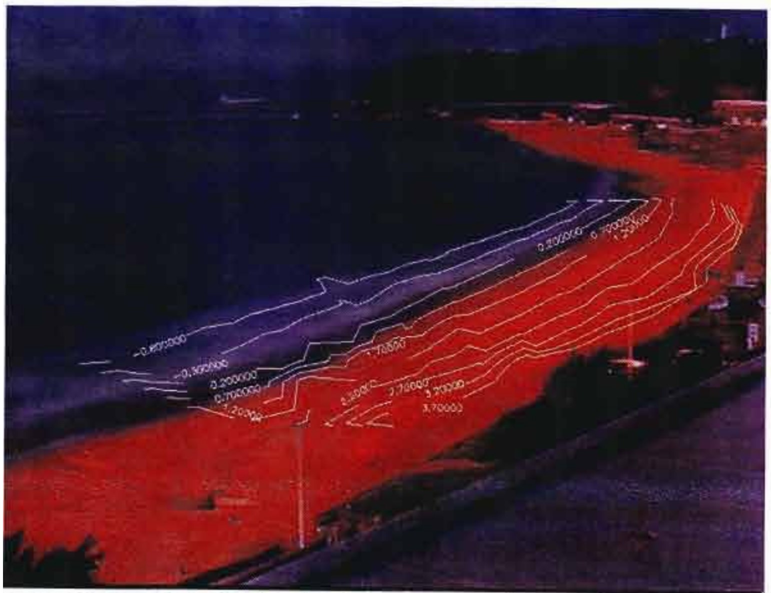


Figure 3.34 Survey (8/6/2000) of Addington Beach superimposed on the AVG image.

The mean difference and the standard deviation between the manually selected waterline and that extracted from the survey data are 2.07m and 1.30m respectively. A plot showing the manually selected waterline and the survey derived waterline can be seen in Figure 3.35.The survey was done by taking measurements ~15m apart along the transects and with transects taken ~15m apart. The data was interpolated to extract the waterline and does not perfectly match the manually selected waterline.

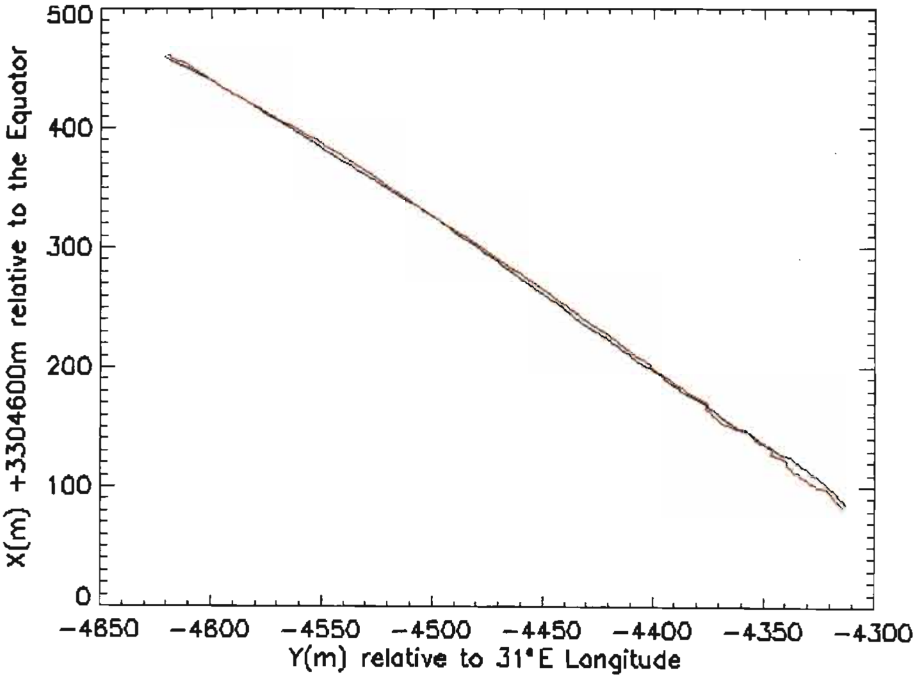


Figure 3.35 Plot of the manually selected waterline (black) and the survey derived waterline (red) given in the Transverse Mercator map projection, which specifies a co-ordinate in meters relative to the Equator and a chosen Longitude. In this case, the Lo31 degrees Cape Datum was used for the projection.

Technique	Accuracy			
	Compared to Manually Selected Waterline		Compared to Surveyed Waterline	
	Mean (m)	σ (m)	Mean (m)	σ (m)
Grayscale Thresholding	4.34	2.85	0.68	0.55
δ Colour Discriminator	1.62	0.54	0.31	0.28
Artificial Neural Network	0.32	0.23	0.52	0.34
Wavelet Analysis	2.47	2.10	0.53	0.37
Fourier Analysis	4.44	2.05	0.77	0.49

Table 3.2 Comparison of the accuracy of the various techniques to the manually selected and surveyed waterlines.

From the above table it is evident that the ANN would be the best technique to implement. The ANN technique has a small standard deviation, which means it does not deviate much from the manually selected waterline. Furthermore, it only has an average offset of 0.32m. The grayscale thresholding technique has the largest standard deviation values for both the comparison with the manually select waterline and survey derived waterline (2.85m and 0.55 respectively) indicating that the other techniques are more accurate. However, further testing of the techniques on image sequences with changing ambient lighting conditions is necessary to confirm the results in Table 3.2 and hence return a measure of robustness.

Further research and testing is necessary to find a technique with acceptable accuracy (low standard deviation) and fast (taking a few seconds or minutes) that would result in an automated waterline extracting coastal imaging station. It has been shown that the colour and temporal aspects of the image sequences captured can be exploited to give a lower standard deviation of the mean between the manually selected waterline and the calculated waterline, hence is more accurate, than the grayscale thresholding technique.

Chapter 4

VALIDATION OF RIP CURRENTS IN THE SURF-ZONE

4.1. Introduction

Alport et al. [6] and Basson [7] applied Digital Correlation Image Velocimetry (DCIV) which calculates the velocity vector field by using cross correlation to measure the displacement of unique features in two consecutive images of the surf-zone. A discussion on DCIV can be found in APPENDIX F.

Results showed that the calculated wave velocities over light areas in the time-averaged images (shallows due to sand bars) were lower as expected. This is consistent with linear wave theory (see section 2.2.3), which implies that in shallower regions the velocities are lower. Stockdon and Holman [39] used linear wave theory to relate the wavenumber, calculated from the Fourier transform of a timestack of the surf-zone, to the water depth (equation [2.5]). The mean difference between the bathymetry derived by this means and a manually surveyed bathymetry was –35cm.

In this chapter an attempt is made to validate the measurement of rip currents in the surf-zone using the advected foam from the breaking waves. Validation was done by calculating the velocities of clearly identifiable objects using cross-correlation of successive images of the surf-zone and comparing this result to the calculated velocities of the unique structure in the advected foam using DCIV.

4.2. Experimental Procedure

900 cylindrical corks (30mm in diameter and 48mm in length) were painted yellow so that they were clearly identifiable in the surf-zone and hence easily tracked in a sequence of images.

A digital Sony CCD camcorder was securely mounted on the North Beach Pier looking down at the surf-zone (~ 2.5m above the sea surface). A two-minute video clip of the surf-zone was then recorded. A number of the corks were then thrown into the surf-zone at regular intervals and recorded. Figure 4.1 shows an example of one of these images. After recording the floating corks for a six-minute period, a further two-minute clip was recorded after the corks had drifted out of the field of view.

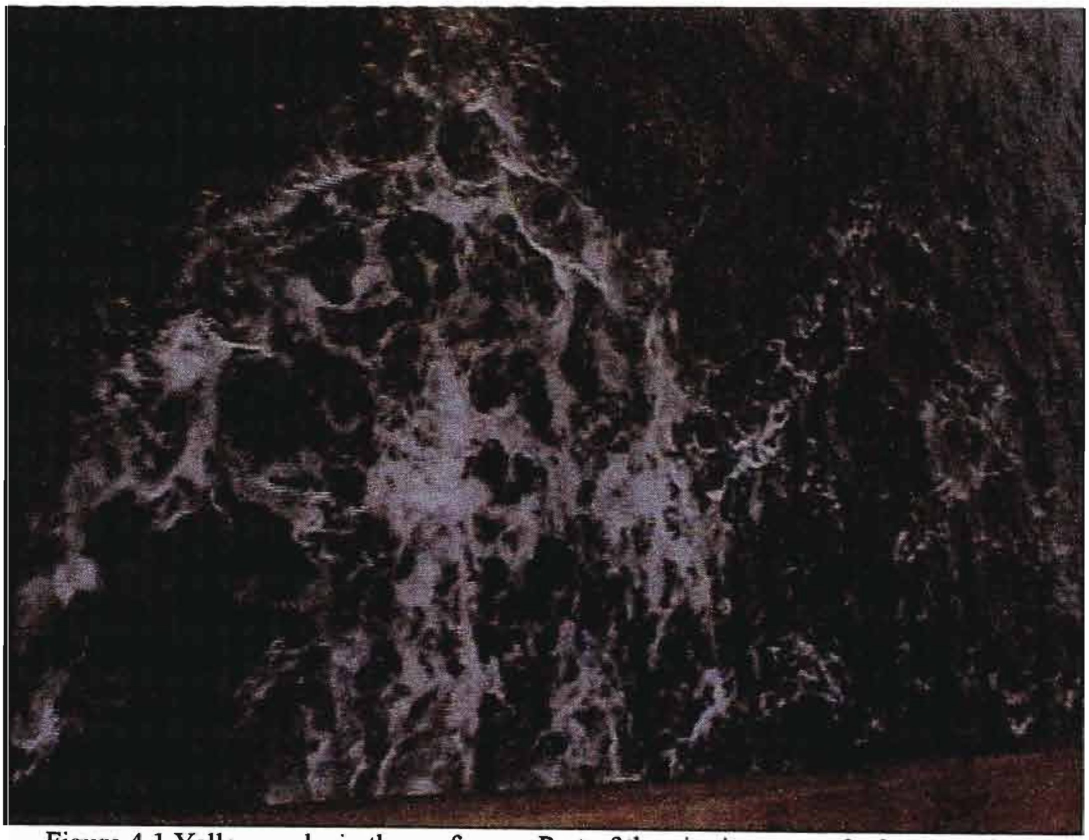


Figure 4.1 Yellow corks in the surf-zone. Part of the pier is seen at the bottom of the image.

The video was captured to an AVI movie clip and later individual bitmaps were extracted. Although the video clip was captured at 25 fps, the bitmaps were extracted at 5 fps since the corks had not moved significantly (more than 16 pixels in any direction) in successive images for a frame rate ≥5 fps.

An IDL routine was then written to step through the individual frames and allowed the user to left-click on each visible cork using a mouse. Cross-correlation using the FFT was used to find the position of each of the selected cork in the next successive frame. This was done by taking a 12x12 pixel sub-image around the point clicked on in the first frame and then padding it (adding zeros) equally on all sides till it was 32x32 pixels in size. The 12x12 pixel sub-image was chosen so that the cork was the main feature in the padded 32x32 sub-image and all other features ignored. Further more the padding was necessary so the dimensions of the sub-image were a power of 2 so the FFT algorithm works faster. This 32x32 pixel sub-image was then cross-correlated with the 32x32 sub-image in the next frame centred about the same point.

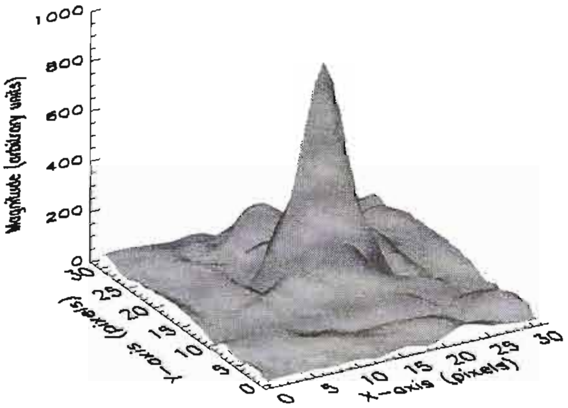


Figure 4.2 An example of a multi-peaked distribution resulting from the cross-correlation.

The choice of 5fps ensured that the cork had not moved more than 16 pixels, that is out of the second 32x32 sub-image, between successive frames. The result of the cross-correlation was a two-dimensional multi-peaked distribution (see Figure 4.2). The highest peak indicated the most probable position of the cork in the second image. The displacement of the highest peak from the centre gave the *x* and *y* displacement vectors of the dominant feature in the sub-image, that is the cork. These displacement vectors were stored in data files and used for analysis. Displacement vectors directed shoreward (left along the *x*-axis of the image in Figure 4.1) and seaward (right along the *x*-axis of the image in Figure 4.1) were stored in separate files.

600 images (two-minutes) were used to calculate the velocity flow fields of the corks using DCIV. The velocity vectors were calibrated by relating the pixel size of the corks in the image to their known dimensions. The corks at the centre of the image had an approximate length of 11 pixels. The length of the cork was 48mm so a calibration factor of 4.36 mm/per pixel was used.

The result of the calculated velocity flow field of the corks in the seaward direction, that is the rip current, can be seen in Figure 4.3 while the calculated velocity flow field of the advected foam can be seen in Figure 4.4. The velocity scale is shown in the upper left hand corner of the images.

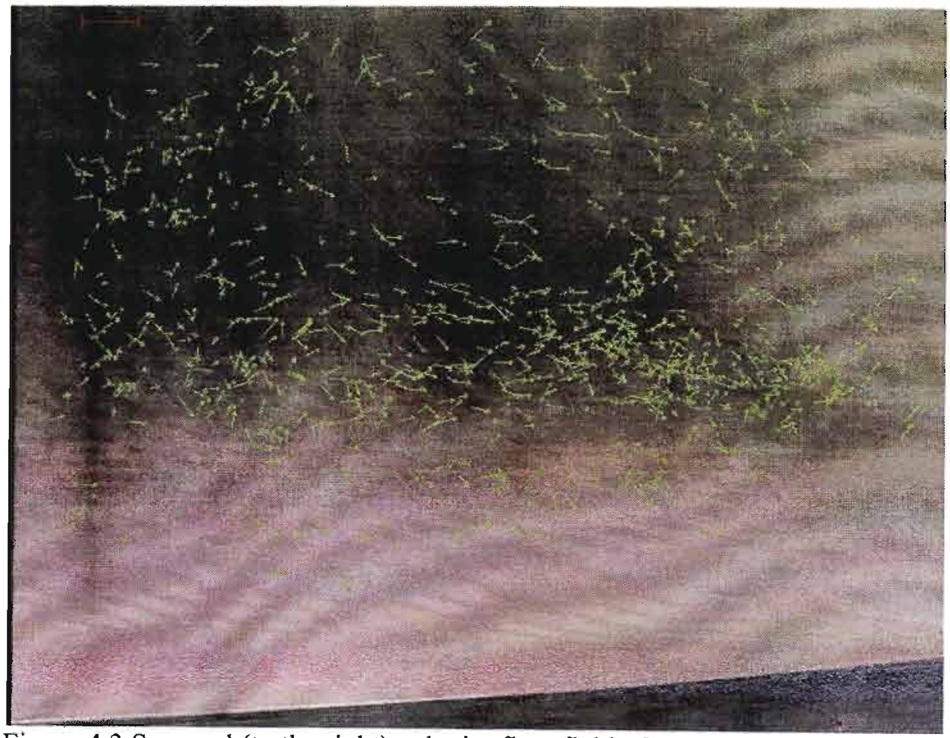


Figure 4.3 Seaward (to the right) velocity flow field of the corks in the surf-zone superimposed on the standard deviation image. The velocity scale is shown in the top left corner of the image.

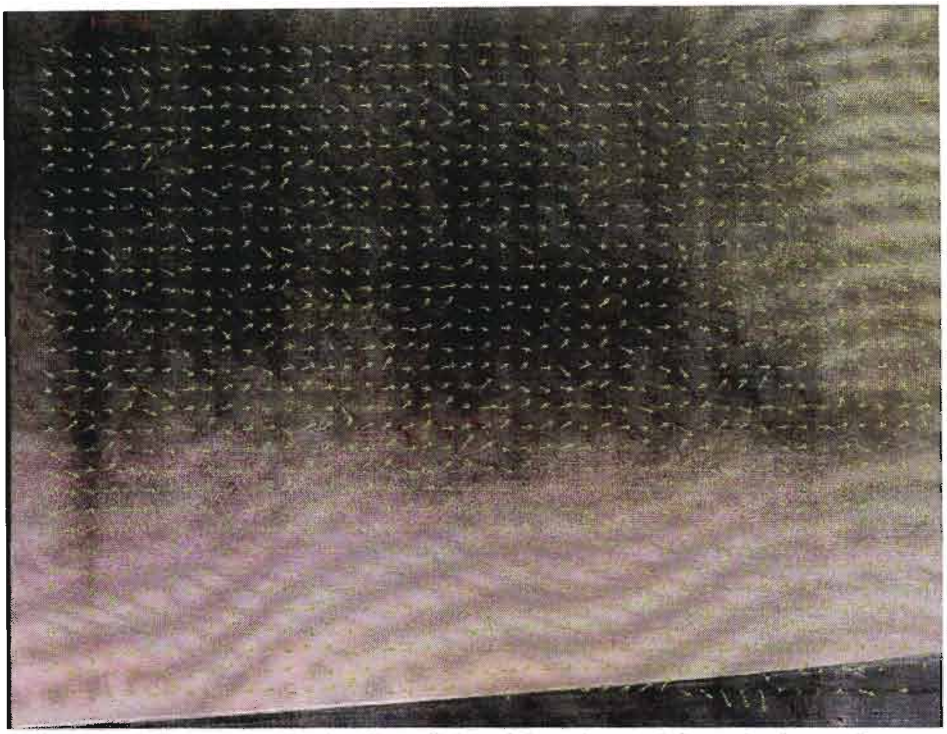


Figure 4.4 Seaward velocity flow field of the advected foam in the surf-zone.

4.3. Results and Conclusion

Although the yellow corks were clearly visible in the surf-zone they could not however be distributed evenly to give a regularly gridded set of data points. The calculated velocity flow field of the corks over a number of frames gives a sparse data set. In the following analysis, it was necessary to average over the seaward and shoreward velocities separately. The seaward velocity is attributed to the rip current, whilst the shoreward velocity is attributed to the forward motion of the advancing roller foam lines.

4.3.1. Seaward-Directed Velocity Flow Field

The magnitudes of the seaward velocity flow field were binned in to 38 equal sized bins each of width of 20 pixels (0.087m) perpendicular to the pier. This bin size was chosen so that there was at least one velocity vector present in each bin. The magnitudes of the velocities within each bin were then averaged. Figure 4.5 shows the resulting plot for the corks (black) and the advected foam (red). The two profiles are similar.

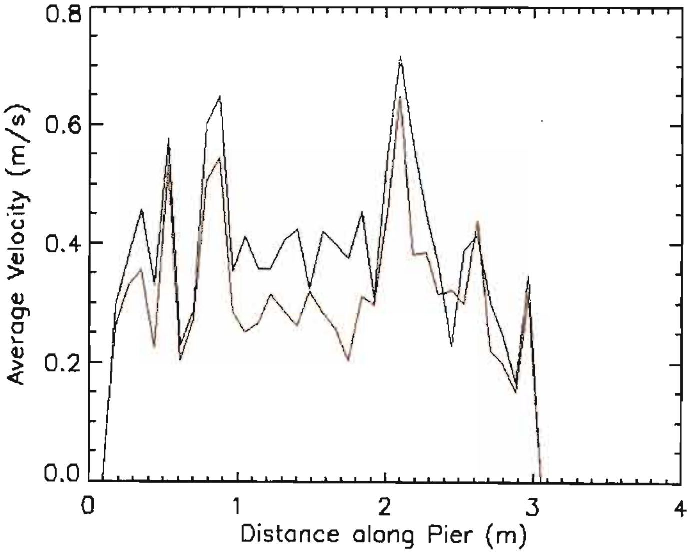


Figure 4.5 Plot of the velocity flow field directed seaward for the corks (black) and the advected foam (red) as a function of the distance along the pier.

The same type of approach was then applied to 27 bins parallel to the pier for the seaward-directed velocity flow field. The result of this analysis can be seen in Figure 4.6, which once again indicates that the two profiles are fairly similar. Furthermore a well-defined channel of width ~0.6m and average velocity ~0.7m/s can be clearly seen at a distance of ~1.3m from the pier.

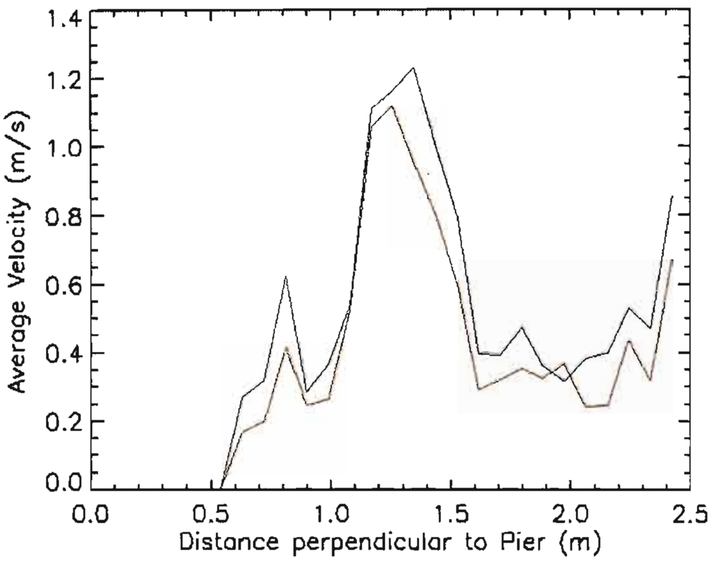


Figure 4.6 Plot of the velocity flow field directed seaward for the corks (black) and the advected foam (red) as a function of the distance perpendicular to the pier.

4.3.2. Shoreward-Directed Velocity Flow Field

The same analysis was then applied to the shoreward directed velocity flow fields with the results shown in Figure 4.7 and Figure 4.8.

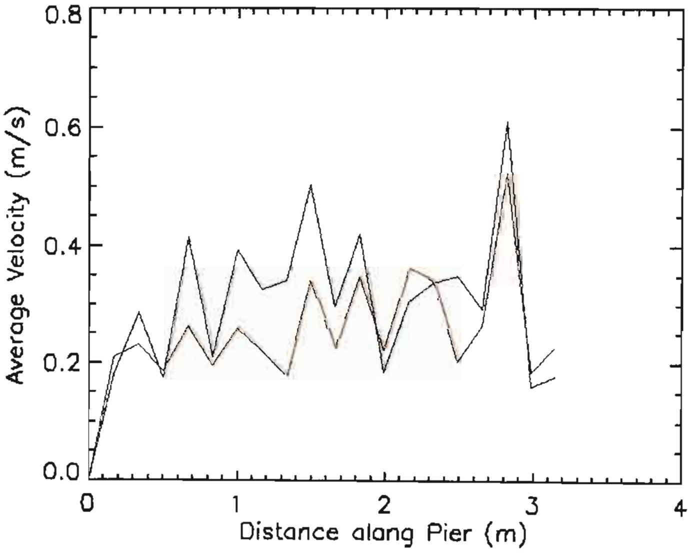


Figure 4.7 Plot of the velocity flow field directed shoreward for the corks (black) and the advected foam (red) as a function of the distance along the pier.

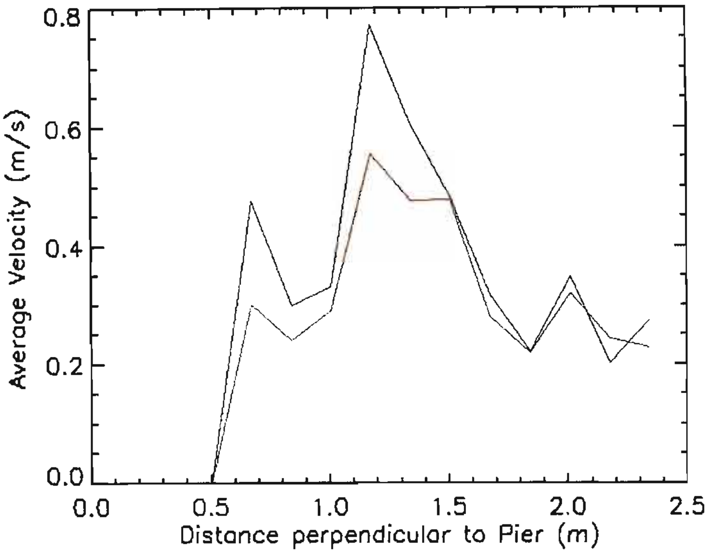


Figure 4.8 Plot of the velocity flow field directed shoreward for the corks (black) and the advected foam (red) as a function of the distance perpendicular to the pier.

In Figure 4.8 the well-defined channel of width 0.6m is again observed. The velocity (~0.6 m/s) in this channel are much higher compared to region outside the channel but less than seaward velocity (~1.2m/s).

In has been shown that the application of the DCIV technique to advected foam gives a similar velocity profile as the application of DCIV to tracking uniquely defined objects (corks) in the same surf-zone. A point-by-point comparison of the profile of the average seaward current velocities across the rip channel differ by an average of 0.1m/s. Further investigation is still necessary. Large objects or more zoom is needed so objects in the surf-zone can be tracked from the permanent coastal imaging.

Finally, it should also be noted that the DCIV technique only measures the surface velocity and for a complete description of the rip current, a depth resolved velocity profile would also be required.

Chapter 5

CONCLUSION

A number of techniques for extracting the waterline from surf-zone video images have been presented and comparisons made. Colour and temporal information have been exploited in obtaining more accurate waterlines. The Artificial Neural Network (ANN) technique uses colour information, which includes the RGB pixel values, and the average and standard deviation for each colour plane, as input. It was shown that an adequate data set spanning various lighting conditions produced a well-defined differentiation between water and sand. A standard deviation value of the mean difference between the manually selected waterline and the ANN derived waterline was shown to be lower (0.23m) than that for the grayscale thresholding technique (2.85m).

The Wavelet and Fourier analysis show a lot of promise for extracting more reliable waterlines than the grayscale thresholding technique. For the Wavelet and Fourier analysis the calculated standard deviations of the mean of the waterlines were 2.10m and 2.05m respectively, which were lower than that of the grayscale thresholding technique (2.85m). These techniques utilise the temporal characteristics of the wave action to enhance the pixel intensity values in surf-zone while suppressing the pixel intensity values outside the surf-zone (on land). More research is necessary in this field before an acceptable accurate and fast solution is found for a fully automated remote coastal imaging station.

The application of Digital Correlation Image Velocimetry (DCIV) to the surf-zone has been validated. The calculated velocity profiles of the corks and the advected foam are very similar and clearly indicate the rip channel. A point-by-point comparison of the profile of the average seaward velocity flow field across the rip channel differ by an average of 0.1m/s. Further investigation is necessary to find the best zoom setting to use to track surface currents using the fixed coastal imaging stations.

It has been shown in this thesis that currently video coastal monitoring is a useful tool for frequent monitoring of the surf-zone and providing useful information about the evolution of the waterline. With more research this will be an indispensable tool as it is less expensive and less labour intensive than current manual survey methods.

REFERENCES

[1] Aarninkhof S.G.J., *Quantification of Bar Bathymetry from Video Observations*, Masters Thesis, Delft University of Technology (Delft Hydraulics, Report H2443), 1996.

[2] Aarninkhof S.G.J., Caljouw M., Stive M.J.F., *Video-based, Quantitative Assessment of Intertidal Beach Variability*, Coastal Engineering 2000, pp 3291-3304, 2000.

[3] Aarninkhof S.G.J., Janssen P.C., Plant N.G., *Quantitative Estimations of Bar Dynamics from Video Images*, Coastal Dynamics '97, pp 365-374, 1997.

[4] Aarninkhof S.G.J., Roelvink J.A., *Argus-Based Monitoring of Intertidal Beach Morphodynamics*, Coastal Sediments '99, pp 2429-2444, 1999.

[5] Alport M.J., Mocke G., Govender K., Marais A., *The Measurement of Nearshore Processes in the Laboratory and Field using Video Imagery*, Proceedings of Ocean 98 Conference (Oceans 98), On CD, 1998.

[6] Alport M.J., Mocke G., Govender K., Naicker J., Basson J., Soltau C., *Video measurements of Nearshore processes in the field and the laboratory*, 5th International Conference on Coastal and Port Engineering in Developing Countries, pp 1038-1049, 1999.

[7] Basson J.F., *Correlation and Neural Network Techniques Applied to Surf-zone Imaging*, Masters Thesis, School of Pure and Applied Physics, University of Natal, 2000.

[8] Battjes J.A., Janssen J.P.F.M, *Energy Loss and Set-up due to Breaking of Random Waves*, Coastal Engineering 1978, pp 569-587, 1978.

[9] Battjes J.A., Stive M.J.F, *Calibration and Verification of a Dissipation Model for Random Breaking Waves*, Journal of Geophysical Research, Vol. 90, No. C5, pp 9159-9167, 1985.

[10] Cartwright M., *Fourier Methods for Mathematicians, Scientists and Engineers*, Ellis Horwood Limited, Great Britain, 1990.

[11] Costa L.d.F., Cesar Jr R.M., *Shape Analysis and Classification: Theory and Practice*, pp 149-196, CRC Press, United States of America, 2001.

[12] Costa L.d.F., Cesar Jr R.M., *Shape Analysis and Classification: Theory and Practice*, pp 467-532, CRC Press, United States of America, 2001.

[13] Davidson M., Huntley D., Holman R., George K., *The Evaluation of Large Scale (km) Intertidal Beach Morphology on a Macrotidal Beach Using Video Images*, Coastal Dynamics '97, pp 385-394, 1997.

[14] Daubechies I., *Ten Lectures on Wavelets*, Society for Industrial and Applied Mathematics, Philadelphia, 1992.

[15] Govender K., *Velocity, Vorticity and Turbulence Measurements in the Surf-zone*, PhD Thesis, School of Pure and Applied Physics, University of Natal, 1999.

[16] Foley J.D., van Dam A., Feiner S.K., Hughes J.F., *Computer Graphics: Principles and Practice*, Second Edition, Addison-Wesley, Massachusetts, 1991.

[17] Graps A.,*An Introduction to Wavelets*, http://www.amara.com/ftpstuff/IEEEwavelet.ps.gz, 1995.

[18] Hesselink L., *Digital Image Processing in Flow Visualization*, Annual Review of Fluid Mechanics, pp 421-485, 1988.

[19] Holland K.T., *Beach Cusp Formation and Spacing at Duck, USA*, Continental Shelf Research 18, pp 1081-1098, 1198.

[20] Holman R.A., Bowen A.J., *Bars, Bumps, and Holes: Models for the Generation of Complex Beach Topography*, Journal of Geophysical Research, Vol. 87, No. C1, pp 457-468, 1982.

[21] Horikawa K., *Nearshore Dynamics and Coastal Processes: Theory, Measurement, and Predictive Models*, University of Tokyo press, Japan, 1988.

[22] Janssen C.M., Hasson W.N., van der Wal R., Ribberink J.S., *Grain-size Influence on Sand Transport Mechanisms*, Coastal Dynamics '97, pp 58-67, 1997.

[23] Jannsen P.C., *Intertidal Beach Level Estimations from Video Images*, Masters Thesis, Delft University of Technology, 1997.

[24] Konicki K.M., Holman R.A., *The Statistics and Kinematics of Transverse Sand Bars on a Open Coast*, Marine Geology 169, pp 69-101, 2000.

[25] LeBlond P.H., Mysak L.A., *Waves in the Ocean*, Elsevier Scientific Publishing Company, Great Britain, 1978.

[26] Lippmann T.C., Holman R.A., *Quantification of Sand Bar Morphology: A Video Technique Based on Wave Dissipation*, Journal of Geophysical Research, Vol. 94, C1, pp 995-1011, 1989.

[27] Lisboa P.G.J. (Ed.), *Neural Networks: Current Applications*, Chapman and Hall, London, 1992.

[28] Maresca J.W. Jr, Seibel E., *Terrestrial Photogrammetric Measurements of Breaking waves and Longshore Currents in the Nearshore Zone*, Coastal Engineering 1976, pp 681-700, 1976.

[29] Mocke G., Govender K., Alport M.J., Diedericks G., *Modelling of Digitally Imaged water levels and flow fields in the Surf-zone*, Proceedings of 27th International Conference on Coastal Engineering (ICCE 2000), pp 1036-1049, 2000.

[30] Nelson M.M., Illingworth W.T., *A Practical Guide to Neural Nets*, Addison-Wesley Publishing Company, Massachusetts, 1990.

[31] Plant N.G., Holman R.A., *Extracting Morphological Information From Field Data*, Coastal Engineering 1998, pp 2773-2784, 1998.

[32] Polikar R., *The Wavelet Tutorial*, http://engineering.rowan.edu/~polikar/WAVELETS/WTtutorial.html, 1995.

[33] Pratt W.K., *Digital Image Processing 2nd Edition*, John Wiley and Sons, United States of America, 1991.

[34] Principe J.C., Euliano N.R., Lefebvre W.C., *Neural and Adaptive Systems: Fundamentals through Simulations*, John Wiley and Son, United States of America, 1999.

[35] Raffel M., Willert C., Kompenhans J., *Particle Image Velocimetry: A Practical Guide*, Springer, Germany, 1998.

[36] Ranasinghe R., Symonds G., Holman R., *Quantitative Characterisation of Rip Dynamics Via Video Imaging*, Coastal Sediments '99, pp 987-1002, 1999.

[37] Rioul O., Vetterli M., *Wavelets and Signal Processing*, Signal Processing Technology and Applications, IEEE Technical Activities Board, pp 85-109, 1995.

[38] Rodriguez A., Bahia E., Diez M., Sánchez-Arcilla Agustin, Redondo, Mestres M., *Experimental Study of Mixing Processes Using Images*, Coastal Dynamics '97, pp 395-404, 1997.

[39] Stockdon H.F., Holman R.A., *Estimation of Wave Phase Speed and Nearshore Bathymetry from Video Imagery*, Journal of Geophysical Research, Vol. 105, No. C9, pp 22015-22033, 2000.

[40] Torrenace C., Compo G.P., *A Practical Guide to Wavelet Analysis*, http://paos.colorado.edu/research/wavelets/bams_79_01_0061.pdf, 1998.

[41] Valens C., *A Really Friendly Guide to Wavelets*, http://perso.wanadoo.fr/polyvalens/clemens/download/arfgtw.pdf, 1999.

[42] van Enckevort, I.M.J., Wijnberg K.M., *Intra-Annual Changes in Bar Plan Shape in a Triple Bar System*, Coastal Sediment '99, pp 1094-1108, 1999.

[43] Wasserman P.D., *Neural Computing: Theory and Practice*, Van Nostrand Reinhold, New York, 1989.

[44] Wijnberg K.M., Holman R.A., *Cyclic Bar Behaviour Viewed by Video Imagery*, Coastal Dynamics '97, pp 375-384, 1997.

[45] Worley C.R., Lippmann T.C., Haines J.W., Sallenger A.H., *The Spatial Variability of Large Scale and Bars*, Coastal Engineering 1998, pp 2865-1876, 1998.

APPENDICES

APPENDIX A - CO-ORDINATE TRANSFORMS

A.1. Introduction

An image captured by a CCD camera returns a two-dimensional (2D) representation of an actual three-dimensional (3D) real world scene. The discrete points that make up the image are known as *pixels*. It is essential to be able to transform the 2D image pixel co-ordinates (x,y) to the 3D real world scene *(X,Y,Z)* and vice versa.

The co-ordinate transforms are used in the software routines (APPENDIX G). In the software the pixel co-ordinates *(x,y)* are referred to as the ***Camera Co-ordinate System (CCS)*** with (0,0) at the centre of the image. The real world or ground co-ordinates are given in meters in accordance with the Transverse Mercator (Gauss Conformal) map projection and is referred to as the ***Scene Co-ordinate System (SCC)***.

A.2. Geometric Transformation

The geometric relationship, illustrated in Figure A.1, between the Ground Co-ordinates *(X,Y)* and Image Co-ordinates *(x,y)* is taken from Aarninkhof [1] and Janssen [23].

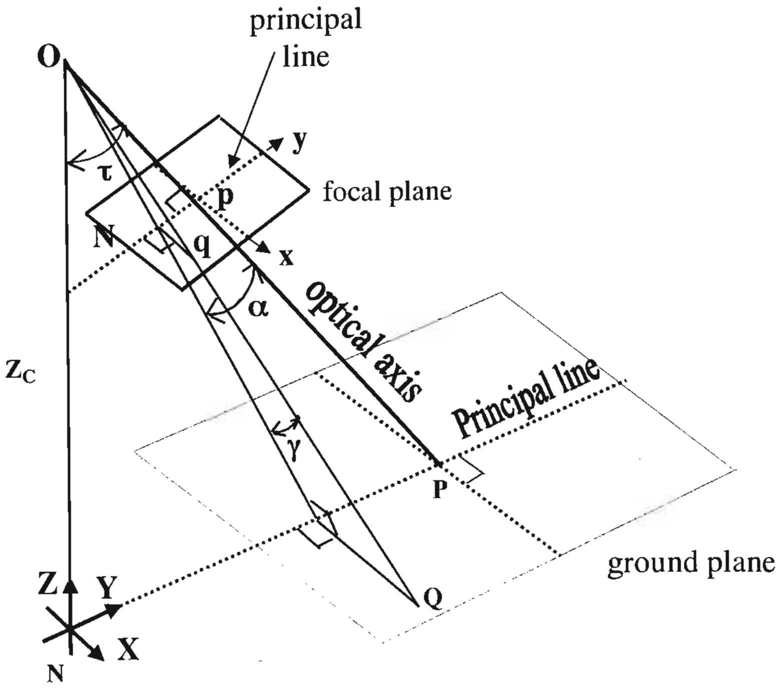


Figure A.1 The geometrical relationship between the ground co-ordinates (*X*,*Y*) and image co-ordinates (*x*,*y*)

The principal line is the line drawn through the principal point (*p* - the centre) of the image and projected onto the corresponding principal point (*P*) in the ground plane.

The camera is located at point *O* in the figure and tilted at an angle τ. Z_C is the vertical height of the focal point above ground level. The angle of roll θ describes the angle the image makes with the horizon. In addition, the angle ϕ is specified to describe the angle that the beach makes with the *x*-axis of the image.

For a given point *Q* in the ground co-ordinate system (X_Q,Y_Q),

$$X_Q = \frac{Z_C \tan\gamma}{\cos(\tau+\alpha)} \qquad [A.1]$$

$$Y_Q = Z_C \tan(\tau+\alpha) \qquad [A.2]$$

The angles α and γ are given by

$$\alpha = \tan^{-1}\left[\frac{y_q}{f_C}\right] \qquad [A.3]$$

$$\gamma = \tan^{-1}\left[\frac{x_q}{\sqrt{y_q^2 + f_C^2}}\right] \qquad [A.4]$$

with f_C is the focal length of the lens.

The above equations [A.1] and [A.2] can be rewritten to instead give the image co-ordinates (x,y) in terms of the ground co-ordinates (X,Y):

$$y_q = f_C \tan\left[\tan^{-1}\left[\frac{Y_Q}{Z_C}\right] - \tau\right] \qquad [A.5]$$

$$x_q = \sqrt{\frac{y_q^2 + f_C^2}{Z_C^2 + Y_Q^2}}\,X_q \qquad [A.6]$$

The focal length of the camera lens, in terms of pixels, is given by,

$$f_C = \frac{x_e}{\tan\left(\frac{\delta}{2}\right)} \qquad [A.7]$$

where x_e is the number of pixels from the principal point to the edge of the image.

A.3. The Transformation Matrices

Capturing an image when the video camera is tilted an angle, τ (measured from the *Z*-axis) results in both the *X* and *Y* (ground) co-ordinates being "compressed" due to perspective. The following discussion can be found in Pratt [33] and the *IDL Reference Manual.* The rotation and perspective matrices can describe this mapping of the ground co-ordinates to image co-ordinates. The inverse of these matrices can then used to map the image pixel co-ordinates to the ground co-ordinates. This is the alternative approach to using equations [A.1], [A.2], [A.5] and [A.6].

The matrices R_x and R_y rotate an image about the *x* and *y*-axes. These rotations occur when the camera has tilt τ and roll θ. The original co-ordinates (*x*,*y*) are transformed to the new co-ordinates (*X*,*Y*).

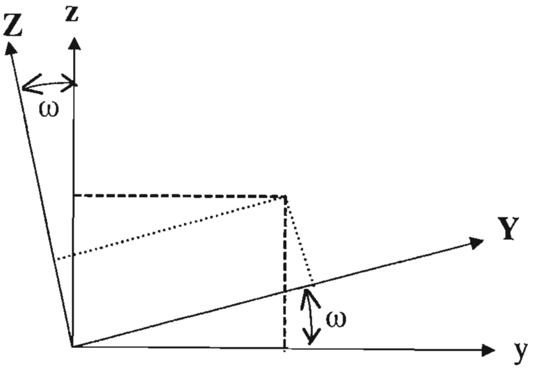


Figure A.2 Rotation about the x-axis for an angle ω

The matrix that represents a rotation about the *x*-axis is given by

$$R_X = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\omega & -\sin\omega & 0 \\ 0 & \sin\omega & \cos\omega & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad [A.8]$$

where $\tau = -\omega$.

The matrix that represents a rotation about the y-axis is given by

$$R_Y = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad [A.9]$$

The matrix necessary to transform a three dimensional image to a two dimensional image is the perspective matrix $\wp$ (see below). f_c is the focal length of the camera, which should be measured in terms of pixels.

$$\wp = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & {}^{-1}\!/_{f_c} & 1 \end{bmatrix} \qquad [A.10]$$

Using the above matrices, the following general transformation are constructed:

$$[X,Y,Z,1] = [z,y,0,1].R_x.R_y\wp \qquad [A.11]$$

and the corresponding reverse transformation

$$[x,y,0,1] = [X,Y,Z,1].R_x^{-1}R_y^{-1}\wp^{-1} \qquad [A.12]$$

which results in the *rectified* image. The *rectified* image has an equi-distant *footprint* (distance between neighbouring pixels either in the x or y direction).

$$R_Z = \begin{bmatrix} \cos\phi & -\sin\phi & 0 & 0 \\ -\sin\phi & \cos\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad [A.13]$$

Once the image has been rectified, it can be rotated by angle ϕ using equation [A.13] so that the x-axis now represents the co-ordinates along the beach (longshore).

A.4. Using the Transformation

For the purpose of rectifying an image the camera position (longitude, latitude and height) as well as the position of at least three ground control points (GCPs), that are clearly visible in the image, are required. With this data in addition to the corresponding pixel positions (x,y) of the GCPs one can use the geometric transform to calculate the focal length f_c and the angle of tilt τ. Once these parameters are found it is possible transform pixel co-ordinated to real world co-ordinates and vice versa.

APPENDIX B - COLOUR REPRESENTATION

B.1. Introduction

For any image processing application the concept of colour and its representation in an image play a major role in defining the approach taken to extracting the necessary information. In this dissertation the colour information in images is used and exploited in many of the techniques presented therefore a discussion on colour representation is necessary.

Many colour models are presented and discussed in image processing texts such as Foley et al. [16] and Pratt [33] however for the purpose of this thesis just the RGB, grayscale and HLS colour models are discussed.

B.2. RGB Colour Model

As the name suggests this model uses a combination of RGB (red, green, blue) to represent a pixel's colour. For each point in a colour image there is a RGB colour triplet. Each value in the triplet is a real value from 0 to 1. Every colour can be represented by various combinations of the values in the RGB triplet, eg. yellow (1,1,0).

This model is used by television sets and PC monitors as the RGB triplet define the primary colour scheme necessary for reconstruction of other colours. The RGB model can be illustrated by the colour cube shown in Figure B.1. The diagonal joining black (0,0,0) and white (1,1,1) gives the gray levels.

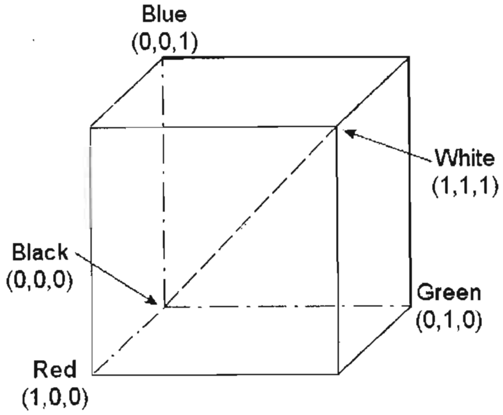


Figure B.1 Colour cube illustrating the RGB colour model.

For digital images each RGB triplet value is mapped so that they have integer values between 0 (minimum) and 255 (maximum) and hence each triplet value can be stored in a byte. Hence the RGB triplet (0,0,0) denotes black, (255,255,255) denotes white, (255,0,0) denotes red and so on. With this representation it is possible to have up to 16,777,215 colours, some of which can be seen in the colour bar in Figure B.2. These colour images are referred to as *24-bit colour* images or *true colour* images.

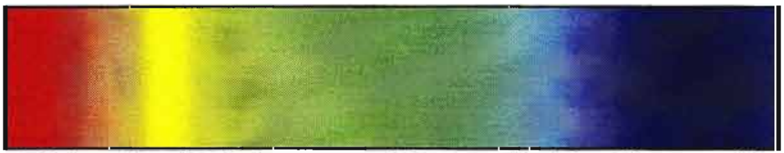


Figure B.2 Colour bar displaying some of the possible 16 million possible colours.

B.3. Grayscale Model

As illustrated above in Figure B.1, the gray levels of the image are represented by the diagonal from (0,0,0) to (1,1,1) in the colour cube. In a grayscale model a single real value from 0 to 1 is used to represent the different gray levels. A 0 is used to represent black and a 1 white. For a digital grayscale image each pixel is represented by a single value (from 0 to 255) where 0 is black and 255 white. This is shown in Figure B.3.

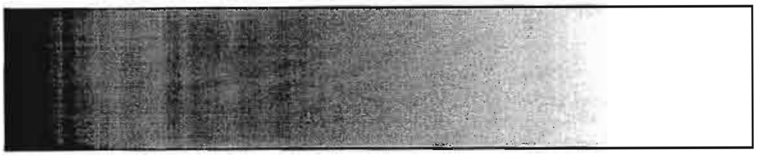


Figure B.3 Grayscale bar for a digital image with a value of 0 (black) on the left and 255 (white) on the right.

Grayscale images are generally used as it takes less space to store and a line extracted from the grayscale image yields a single plot of the pixel intensities in the image. In this thesis a grayscale image is calculated by taking the average RGB value for each pixel in the colour image.

B.4. HLS Colour Model

The HLS (hue, lightness, saturation) colour model like the RGB colour model is represented by a triple. This triple however does not have all real values from 0 to 1. The hue is cyclic in nature and is represented by a real value from 0 to 360 where 360 is equal to 0. This can be described as a colour wheel with red at 0 , green at 120 and blue at 240 . The lightness and saturation are real values from 0 to 1. The relationship between hue, lightness and saturation is illustrated in Figure B.4.

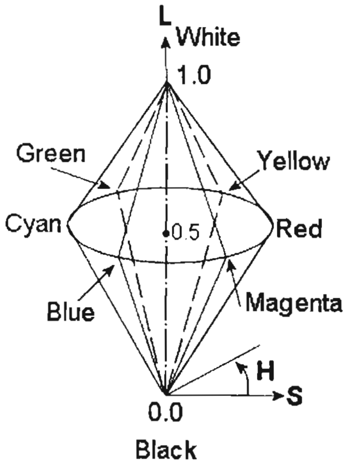


Figure B.4 Visual representation of HLS colour model.

From Figure B.4 it is observed that white is given by L = 1.0 and H and S undefined and black is given by L = 0.0 and H and S again undefined. The primary colour red is given by L = 0.5 and H = 0 and S = 1.0. Similarly all the other colours are given by different values of H around the colour circle with L = 0.5 and S = 1.0. By varying S and L different shades of the colours are obtained.

For this thesis it was necessary to convert from the RGB colour model to the HLS colour model. The mapping from RGB to HLS implemented by IDL is defined by Foley et al. [16] for the RGB triplet (r,g,b) having real values from 0 to 1 as

$$L = \frac{\max(r,g,b) + \min(r,g,b)}{2} \qquad \text{[B.1]}$$

For the achromatic case (gray levels) where $\max(r,g,b) = \min(r,g,b)$ since $r = g = b$, S = 0 and H is undefined.

For the chromatic case, S is defined as

$$S = \frac{\max(r,g,b) - \min(r,g,b)}{\max(r,g,b) + \min(r,g,b)} \qquad \text{for } L \leq 0.5$$

$$S = \frac{\max(r,g,b) - \min(r,g,b)}{2 - [\max(r,g,b) + \min(r,g,b)]} \qquad \text{for } L > 0.5 \qquad \text{[B.2]}$$

and H is defined as

$$H = \frac{g - b}{\max(r,g,b) - \min(r,g,b)} \qquad \text{for } r = \max(r,g,b)$$

$$H = 2 + \frac{b - r}{\max(r,g,b) - \min(r,g,b)} \qquad \text{for } g = \max(r,g,b) \qquad \text{[B.3]}$$

$$H = 4 + \frac{r - g}{\max(r,g,b) - \min(r,g,b)} \qquad \text{for } b = \max(r,g,b)$$

The H is then converted to positive degrees by

$$\begin{gathered} H = H * 60 \\ H = H + 360 \text{ for H} < 0 \end{gathered} \qquad \text{[B.4]}$$

This mapping was necessary so that all the colour information present in a colour image is mapped onto a single plane (the hue plane).

APPENDIX C - ARTIFICIAL NEURAL NETWORKS (ANNS)

C.1. Introduction

Artificial Neural Networks (*ANNs*) form part of a relatively new field of research to solving problems that cannot easily be solved by conventional programming techniques. The theory and application of ANNs to problem solving can be found in texts like Lisboa [27], Nelson et al. [30], Principe [34] and Wasserman [43]. ANNs are implemented in hardware but software has been developed to simulate the ANN. The software can be used to fulfil the same purpose, albeit more slowly, or as an aid to deciding the architecture of the ANN before implementing it in hardware.

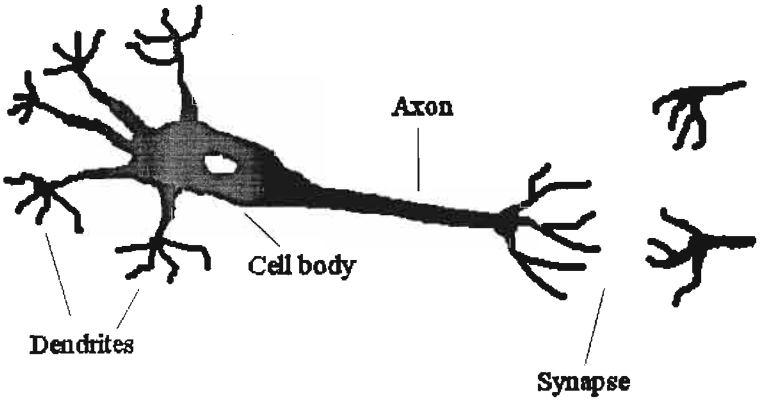


Figure C.1 A simplified representation of a biological neuron.

ANNs as the name suggests are based on the biological neural networks found in the human brain. Neural networks consist of a number of *neurons* that form a network. Biological neurons consist of a *cell body*, a number of *dendrites* and a fibre called *axon*. The gap between the *axon* of one neuron and the *dendrites* of the next is called the *synapse*.

The *dendrites* receive impulses from receptor organs and/or from *axons* from other neighbouring neurons. Each neuron has a *threshold* and the incoming impulses can excite or inhibit that neuron towards this threshold. If the *threshold* is equalled or exceeded then an impulse is generated and passed on to neighbouring neurons and/or effector organs.

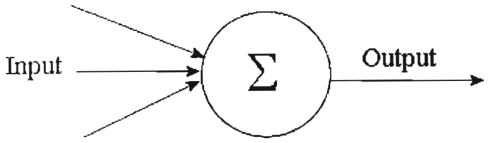


Figure C.2 A schematic representation of an artificial neuron.

The artificial *neuron* (a processing element in hardware terms) plays the same role as a biological neuron in a neural network. It receiving a number of inputs and based on those inputs it performs an *activation function* and outputs a result.

An ANN consists of a number of *layers*, an *activation function* for the neurons and a *learning rule*. The choice of which type of these components (layers, activation function and learning rule) are used in created the ANN affect the efficiency and accuracy of the solution being sought.

C.2. Layers in an Artificial Neural Network

A simple ANN will consist of a number of neurons in the *input layer* and a number of neurons in the *output layer*. The number of neurons in the *input layer* is dependent on the number of elements in the input while the number of neuron in the *output layer* is dependent on the number of possible user-defined outcomes.

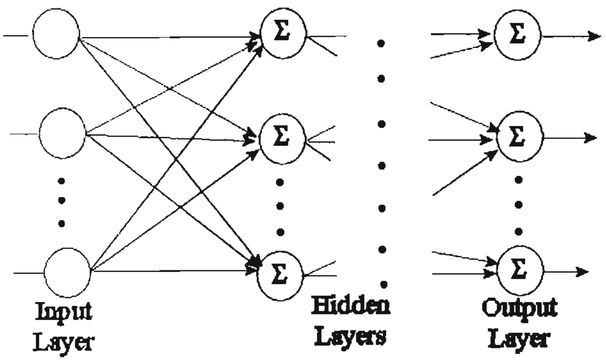


Figure C.3 Neural network architecture with *n* hidden layers.

These simple ANNs are limited to solving *linearly separable* (simple) problems. To solve more complex problems one or more *hidden layers* of neurons must exist between the input and output layers. Neural networks with this architecture (with hidden layers) are referred to as *multi-layer neural networks*.

Once the number of hidden layers (if any) has been chosen, the type of inter-connections between the different layers is set. A *feedforward neural network* has neurons in a particular layer only connecting to neurons in another layer. A *feedback neural network* however has neurons connecting to neurons in other layers and to other neurons within its own layer.

A *forwardpropagation neural network* has the outputs of the neurons propagating only in one direction. A neural network that has each neuron in a particular layer connected to every neuron in the next layer is referred to as being *fully connected.*

C.3. Activation Functions

The connections between the neurons hold *weights* (real values). Neurons in one layer accept a number of inputs from the previous layer. These inputs are multiplied by the appropriate weight and the sum of this is passed to the *activation function.* The *activation function* can be a *step* function, a *sigmoid* function or a *tanh* function. The *activation function* performs an operation on the result passed to it and determines the output of that neuron. The user chooses the *activation function* to be used for the ANN architecture.

C.4. Learning/Training Rules

A neural network needs to be "taught" how to associate the input with the desired outputs. The *weights* connecting the neurons are adjusted either by *supervised learning* or *unsupervised learning.*

In *supervised learning*, the test patterns are presented successively to the neural network and the resulting output compared to the desired output. Changes are made to the weights by a process know as *backpropagation* so that the actual output is closer to the desired output. This continues until the difference between the actual output and desired output, usually the mean square error (MSE), is within an acceptable tolerance. Each cycle in which the data (*exemplars*) is presented is known as an *epoch.*

In *unsupervised learning* there is no desired output for comparison. The network classifies the inputs into its own categories. This phenomenon is known as *self-organisation.*

C.5. Application of ANNs

As a result of its ability to classify, optimise, predict, compress and analyse data ANNs have been used in numerous fields:

- Automated inspection and monitoring
- Business and finance
- Computer vision
- Medical applications
- Optimisation problems
- Speech processing

APPENDIX D - THE FOURIER TRANSFORM (FT)

D.1. Introduction

The Fourier Transform (FT) is a popular and widely used mathematical operation; the theory of which can be found in texts like Cartwright [10], Costa et al. [11] and Pratt [33]. The FT is extensively used in signal processing and image analysis due to the ability to return the frequency spectra of the input function.

The forward continuous one-dimensional FT *F(u)* of an input function *f(x)* is defined by

$$F(u) = \int_{-\infty}^{\infty} f(x)e^{-2\pi iux}dx \qquad [D.1]$$

and the inverse transform is defined by

$$f(x) = \int_{-\infty}^{\infty} F(u)e^{2\pi iux}du \qquad [D.2]$$

F(u) and *f(x)* are referred to as a *Fourier pair.* In general *f(x)* and *F(u)* are complex functions. When the input function *f(x)* is a time series, the FT returns the frequency spectrum (Hz) showing the spectral composition of the function. When *f(x)* is a function of displacement, the FT returns the spatial distribution of *f(x)*. Generally *f(x)* is referred to as a function in the *time domain* and *F(u)* as the function in the *frequency domain.*

D.2. Interpreting the Fourier Transform

If *Re{F(u)}* and *Im{F(u)}* are the real part and the imaginary part of *F(u)*, respectively, then the *magnitude* (or *modulus*) of *F(u)* is defined by

$$|F(u)| = \sqrt{[Re\{F(u)\}]^2 + [Im\{F(u)\}]^2} \qquad [D.3]$$

and the *phase* is defined by

$$\Phi\{F(u)\} = \tan^{-1}\left[\frac{Im\{F(u)\}}{Re\{F(u)\}}\right] \qquad [D.4]$$

And finally the *two-sided power spectrum* of *f(x)* is defined by

$$P_f(u) = |F(u)|^2 = F^*(u).F(u) \qquad [D.5]$$

where $F^*(u)$ is the complex conjugate of *F(u)*. The *power spectrum* shows the energy distribution for the different frequencies hence showing which frequencies are predominant in the input signal.

C.2. Properties of the Fourier Transform

The following properties hold for two functions, *f(x)* and *g(x)*, and their respective Fourier transforms, *F(u)* and *G(u)*.

a) Symmetry Property:

$$F(x) = \int_{-\infty}^{\infty} f(-u)e^{2\pi iux}du \qquad [D.6]$$

b) Shifting Property:

$$f(x - x_0) = \int_{-\infty}^{\infty} F(u)e^{2\pi iu(x-x_0)}du \qquad \text{(time domain)}$$

$$[D.7]$$

$$f(x).e^{2\pi iu_0x} = \int_{-\infty}^{\infty} F(u-u_0)\, e^{2\pi i(u-u_0)x}du \qquad \text{(frequency domain)}$$

c) Scaling Property:

$$f(ax) = \frac{1}{|a|}\int_{-\infty}^{\infty} F(u)e^{\left(\frac{2\pi iux}{a}\right)}du \quad \text{(time domain)}$$

[D.8]

$$\frac{1}{|a|}.f\left(\frac{x}{a}\right) = \int_{-\infty}^{\infty} F(a.u)e^{2\pi iaux}du \quad \text{(frequency domain)}$$

where a is any real constant.

d) Linearity Property:

$$\begin{aligned}\Im\{a.f(x)+b.g(x)\} &= a.\Im\{f(x)\}+b.\Im\{g(x)\}\\ &= a.F(u)+b.G(u)\end{aligned} \quad [D.9]$$

where $\Im$ represents the forward FT, a and b are real constants.

e) Convolution Theorem

$$(f \otimes g)(x) = F(u).G(u) \quad [D.10]$$

where the "⊗" denotes the convolution operator.

f) Correlation Theorem

$$(f \circ g)(x) = F^*(u).G(u) \quad [D.11]$$

where $F^*(u)$ is the complex conjugate of $F(u)$ and the "o" denotes the correlation operator.

The convolution theorem, represented by [D.10], is regularly used as it faster to perform convolution in the frequency domain for large data sets. High-pass, low-pass and band-pass filtering is achieved by taking the FT of the signal, multiplying it with an appropriate filter and then taking the inverse. This is useful for removing noise and smoothing signals.

D.3. Short-Time Fourier Transform (STFT)

One of the major shortcomings of the FT is the fact that it returns the frequency composition of the whole input signal. Small disturbances in the signal affect the whole frequency spectrum. To overcome this The Short-Time Fourier Transform (STFT) or windowed Fourier transform was created. The STFT is defined as

$$STFT(v,u) = \int_{-\infty}^{\infty} g^{*}(x-v) f(x) e^{-2\pi i u x} dx \qquad [D.12]$$

where the function *g(x)* is the window function that moves along*f(x)*.

Disturbances outside the observation window are ignored so that the STFT is a two-dimensional time-frequency representation of the localised spectral composition of the signal centred about *v*. Since the STFT uses a fixed window size it results in the time-frequency resolution being limited by the time-bandwidth product, which is defined as

$$\Delta x . \Delta u \geq \frac{1}{4\pi} \qquad [D.13]$$

This implies that each part of the signal can be analysed with either good time resolution or good frequency resolution but not both at once.

D.4. Two-dimensional Fourier Transform

The data to be transformed for analysis purposes is not always one-dimensional, and in the case of image analysis, it is two-dimensional. Equation [D.1] can be extended to the two-dimensional form by

$$F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) e^{-i2\pi(ux+vy)} dx dy \qquad [D.14]$$

and the inverse of this is defined by

$$f(x,y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}F(u,v)e^{2\pi i(ux+vy)}dudv \qquad [D.15]$$

D.5. Discrete Fourier Transform (DFT)

Most of the analysis done today uses data that has been sampled, that is, composed of a number of discrete points. In the case of image processing, the data is a number of pixels and for a time series, it is data captured at particular time intervals.

The discrete forward FT is defined by

$$F(u)=\frac{1}{N}\sum_{x=0}^{N-1}f(x)e^{\left(\frac{-2\pi iux}{N}\right)} \qquad [D.16]$$

and the inverse is defined by

$$f(x)=\sum_{u=0}^{N-1}F(u)e^{\left(\frac{2\pi iux}{N}\right)} \qquad [D.17]$$

The above equations this can be extended to the two-dimensional case.

The limitation of the discrete form is known as the *Nyquist frequency*. The *Nyquist frequency* is defined by

$$f_{\max}=\frac{1}{2.\Delta t} \qquad [D.18]$$

where *Δt* is the *sampling interval.*

f_{max} is the maximum frequency at which a signal can be resolved. If higher resolution of the signal is attempted then a phenomenon known as *aliasing* occurs. To overcome this problem it is imperative that one chooses the proper *sampling interval*, which is as small as possible.

D.6. Fast Fourier Transform (FFT)

Utilising the DFT above ([D.16]) on an N-point sample would require $O(N^2)$ operations, where $O(x)$ represents the order of x. For large sample sizes, this will require a lot of processing time. The Fast Fourier Transform (FFT) proposed by Cooley and Tukey drastically reduces this to $O(N \log_2 N)$ operations. However, the size of the data has to be a power of 2. If the sample size is not a power of 2, the easiest option is to *pad* the sample. Padding refers to adding a number of zeros to the data until the size is at the next power of 2.

The FFT has been extensively used in this dissertation, especially to perform convolution, cross-correlation and to produce the power spectra of various data sets.

APPENDIX E - THE WAVELET TRANSFORM (WT)

E.1. Introduction

The theory of the Wavelet Transform (WT) can be found in Daubechies [14], Costa [11], Graps [17], Rioul and Vetterli [37], Polikar [32], Torrence and Compo [40] and Valens [41]. The Wavelet Transform was formulated to overcome the shortcomings of the Fourier Transform (FT) (APPENDIX D). The FT would return the spectrum of a time series, with the assumption that the spectrum does not vary in time, that is it is stationary. The WT returns both frequency and temporal details of an input signal resulting in a two-dimensional output for a one-dimensional input function.

The one-dimensional Continuous Wavelet Transform (CWT) $W(s,\iota)$ of a function $f(t)$ is defined by

$$W(s,\iota)=\int_{-\infty}^{\infty} f(t)\,\psi_{s,\iota}^{*}(t)\,dt \qquad \text{[E.1]}$$

where * denotes the complex conjugate operator and ι and s are real values. $\psi_{s,\iota}(t)$ is referred to as the *wavelets* or the *wavelet basis functions*, which are defined by

$$\psi_{s,\iota}(t)=\frac{1}{\sqrt{s}}\psi\left(\frac{t-\iota}{s}\right) \qquad \text{[E.2]}$$

where $s>0$ is the scale (or *dilation*) factor, $\iota \geq 0$ is the translation factor. $f(x)$ is decomposed using scaled and translated versions of the *mother wavelet,* $\psi(t)$, to return $W(s,\iota)$.

The inverse CWT is defined by

$$f(t) = C_{\psi}^{-1} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} W(s,\iota)\psi_{s,\iota}(t)\frac{d\iota\, ds}{s^2} \qquad [E.3]$$

where C_{ψ}^{-1} is defined by

$$C_{\psi} = 2\pi \int_{-\infty}^{\infty} \frac{|\Psi(\omega)|^2}{|\omega|} d\omega < \infty \qquad [E.4]$$

where $\Psi(\omega)$ is the Fourier Transform of $\psi(t)$. The admissibility condition for wavelets states that if C_{ψ} is finite then the inverse exits. This implies that the *wavelets* are finite and are oscillatory in nature with an average of zero in the time domain.

The other restrictions imposed on wavelets are the regularity conditions which states that the wavelets should have a finite number of vanishing moments, that is,

$$\int_{-\infty}^{\infty} t^i \psi(t) dt = 0 \qquad i = 0,1,\ldots,n \qquad [E.5]$$

which implies that the wavelets are smooth and well located in both domains.

E.2. Interpreting the Wavelet Transform

If *Re{*$W(s,\iota)$*}* and *Im{*$W(s,\iota)$ *}* are the real part and the imaginary part of $W(s,\iota)$, respectively, then the *magnitude* (also referred to as the *scalogram*) of $W(s,\iota)$ is defined by

$$|W(s,\iota)| = \sqrt{\left[Re\{W(s,\iota)\}\right]^2 + \left[Im\{W(s,\iota)\}\right]^2} \qquad [E.6]$$

and the *phase* is defined by

$$\Phi\{W(s,\iota)\} = \tan^{-1}\left[\frac{Im\{W(s,\iota)\}}{Re\{W(s,\iota)\}}\right] \qquad [E.7]$$

And finally the *wavelet power spectrum* is defined by

$$P_W(t) = |W(s,\iota)|^2 \qquad [E.8]$$

The *wavelet power spectrum* gives the signal energy distribution. This provides a good visual interpretation of the scale/position composition of the signal.

Although the STFT (D.3) returns a similar two-dimensional result as the WT, the WT has the advantage of a varying window size depending on the frequency. Whereas the STFT has fixed time and frequency interval, limiting the time-frequency resolution ([D.13]), the WT exhibits a relative bandwidth or constant Q property defined as

$$Q = \frac{\Delta f}{f} \qquad [E.9]$$

This allows high frequency components to be analysed with better time resolution and low frequency components to be resolved with better frequency resolution.

E.3. Properties of the Wavelet Transform

The following properties hold for *f(t)* and its Wavelet transforms $W(s,\iota)$ using the wavelet basis function $\psi_{s,\iota}(t)$.

a) Translation Property:

$$W(s,\iota) = \int_{-\infty}^{\infty} [f(t) + C]\psi^*_{s,\iota}(t)dt \qquad [E.10]$$

where *C* is a constant complex value.

b) Scaling Property:

$$\begin{aligned} W(s,\iota) &= \int_{-\infty}^{\infty} c_1 f\left(\frac{t}{c_2}\right)\psi^*_{s,\iota}(t)dt \\ &= c_1\frac{1}{\sqrt{s}}\int_{-\infty}^{\infty} f(t)\psi^*_{s,\iota}\left(\frac{t-\iota/c_2}{s/c_2}\right)dt \end{aligned} \qquad \text{[E.11]}$$

where c_1 and c_2 are constants real values.

c) Rotation Property:

$$\begin{aligned} W(s,\iota) &= \int_{-\infty}^{\infty} e^{j\theta} f(t-t_0)\psi^*_{s,\iota}(t)dt \\ &= \frac{1}{\sqrt{s}}\int_{-\infty}^{\infty} f(t)\psi^*_{s,\iota}\left(\frac{t-(\iota-t_0)}{s}\right)dt \end{aligned} \qquad \text{[E.12]}$$

where θ is the angle of rotation.

D.4. Discrete Form of the Wavelet

For the discrete basis wavelet functions the mother wavelet is represented in a discrete form, that is, the parameters ι and s have discrete values (Daubechies [14]). By setting $s = s_0^m$ (where s_0>1) and $\iota = n\iota_0 s_0^m$ wavelet definition [E.2] can be rewritten as

$$\begin{aligned} \psi_{m,n}(t) &= \frac{1}{\sqrt{s^m}}\psi\left(\frac{t-n\iota_0 s_0^m}{s_0^m}\right) \\ &= \frac{1}{\sqrt{s^m}}\psi\left(t/_{s_0^m} - n\iota_0\right) \end{aligned} \qquad \text{[E.13]}$$

and the Wavelet transform is now represented as

$$W_{m,n} = \frac{1}{\sqrt{s^m}} \int_{-\infty}^{\infty} f(t)\psi^*\left(t/_{s_0^m} - n\tau_0 \right) dt \qquad [E.14]$$

The inverse transform is defined as

$$f(t) = \sum_{m,n} W_{m,n}.\psi_{m,n}(t) \qquad [E.15]$$

D.5. The Morlet Wavelet

The Morlet wavelet was one of earliest created and is used in this dissertation as the wavelet basis function for the transformation. The Morlet wavelet was used as it is well localised in the frequency domain. The Morlet function (Costa et al. [11]) is described as a Gaussian modulated by a complex exponential and is defined by

$$\psi_M(t) = e^{i2\pi f_0 t}.e^{-t^2/2} + \eta(t) \qquad [E.16]$$

where $\eta(t)$ is the correction necessary to fulfil the admissibility condition [E.4].

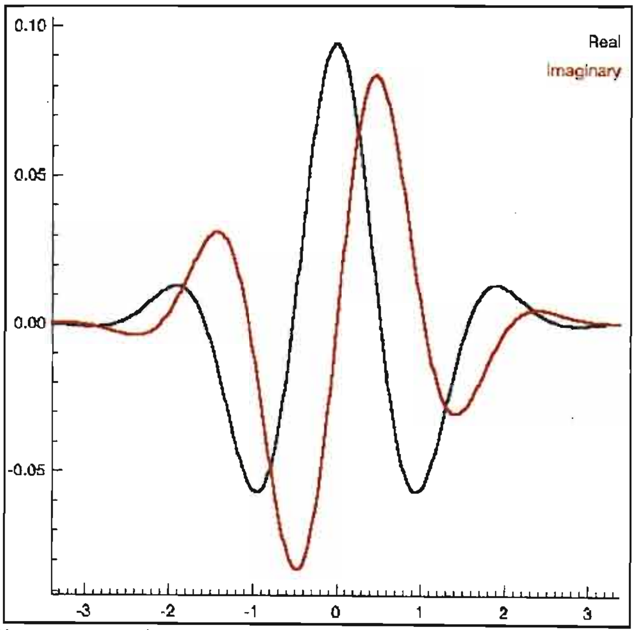


Figure E.1 Plot of a Morlet basis function of order 3.

APPENDIX F - DIGITAL CORRELATION IMAGE VELOCIMETRY (DCIV)

F.1. Introduction

Particle image velocimetry (PIV) is a popular technique used to track moving particles in images. The theory and application of this technique is well documented (Hesselink[18] and Raffel [35]) and used by recent graduate students Basson [7] and Govender [15]. PIV has been used for many years in fluid flow analysis. This is done by adding tracer particles (*seed particles*) to a fluid and imaging a plane with *seed particles*. By observing the motion of these particles over time, the velocities of the individual *seed particles* can be calculated and the velocity flow fields determined. Digital correlation image velocimetry (DCIV) was developed as an extension of PIV and uses cross-correlation of consecutive images to find the velocities of identifiable features.

F.2. PIV Experimental Requirements and Arrangement

The equipment needed for a typical PIV measurements are:

- Seed particles that are easily visible in the fluid medium.
- Laser and lens assembly to produce a light sheet.
- Imaging equipment to record the particles.

The laser and lens assembly are used to create a light sheet that illuminates the seed particles in a single plane. The imaging equipment records the particles in the light sheet.

The *seed particles* must not be too densely or sparsely distributed in the light sheet. Medium particle density is ideal for PIV. The laser characteristics are dependent on the seed size and camera's spectral response. The imaging equipment could be of an optical nature (photographic film) or of a digital nature (CCD camera connected to a VCR or a PC via a frame-grabber).

F.3. PIV Recording Techniques

PIV recording techniques consist of a single frame or multiple frame technique. The single frame technique is limited to optical imaging (photographic film). The single frame technique uses time-lapsed photography (long exposure time which is the time needed to track the particles). The particles appear as streaks on the processed film. The distance the particle has travelled is easily estimated from the length of the streak and the time taken is the exposure time so the velocity is easily calculated.

The multiple frame technique requires individual images to be captured at a particular time interval. The position of particles in the successive images gives its start and end positions and using the set time interval one can calculate the velocities of the individual particles.

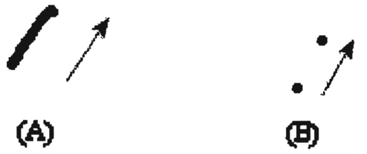


Figure F.1 PIV techniques: (A) shows the streak using the single frame technique and (B) shows the result of the multiple frame technique

The choice of the exposure time in the single frame technique and the time interval between successive frames in the multiple frame technique is critical. If the time is too long the particle will be out of the image before being recorded properly. Also of importance is the conversion of co-ordinates from the pixel co-ordinates to the real world co-ordinates. If the camera is not normal (perpendicular) to the light sheet this is not a linear relationship and the co-ordinate transformations discussed in APPENDIX A can be applied.

F.4. DCIV Recording Technique

The DCIV recording technique utilises digital images captured at a particular time intervals (Δt). Two or more frames are captured and the cross-correlation method implemented to return the displacement of each particle in successive frames. The cross-correlation is usually implemented by using the correlation theorem [D.11] of the Fourier Transform (APPENDIX D). This is done by extracting sub-images containing the individual particles and cross-correlating it with the successive image. The result returned is a multi-peaked distribution where the position of the maximum value is the most likely position of that particle in the successive image.

Plotting these displacement vectors results in the instantaneous velocity field being observed. By averaging these vectors over a number of frames the time averaged flow field can be estimated.

APPENDIX G - SOFTWARE ROUTINES

Many of the software routines written in IDL use a structure called **scene,** which contains the parameters and data necessary to rectify the oblique images. The **scene** structure has the following tags:

Tag	**Description**
Tilt	Calculated angle of tilt of the camera
Fov	Calculated field of view of the camera
Focal	Calculated focal length of the camera
Rotangle	Angle of rotation between the *CCS* and *SCS*
CameraX	Latitude of the camera
CameraY	Longitude of the camera
CameraZ	Height of the camera
ImageName	Image file name
Coordname	Data co-ordinate file name
Roll	Calculated angle of roll
BeachAngle	Angle of rotation so the beach is parallel to the image *x* direction

Since most of the routines do many floating-point operations, these routines are resource intensive. A fast computer with lots of memory and available hard-drive space is required.

Cont_Color

This procedure accepts the names of a snap image and a standard deviation image and a value to perform a contour on as well as a value for the median filter width and displays the snap image with a contour obtained from the standard deviation image superimposed on it. The value specified for the contour level should be between 0 and 255.

Calling Sequence

Cont_Color,SnapName,StdName,C_Level,M_Width

Arguments

SnapName	Name of the file containing the snap image
StdName	Name of the file containing the standard deviation image
C_Level	Contour level value
M_Width	Width to be used in the median filter

Example

```
Cont_Color,"c:\Mbos_980318_164022_C1snap.jpg","c:\Mbos_980318_16
4022_C1std.jpg",60,8
```

Footprint

The procedure **Footprint** displays a contour plot of the pixel footprints for an oblique image. The user is allowed to choose a **roi**, otherwise the whole image processed. If the keyword **roi** is set then the image is displayed and the routine pauses until the user has selected the **roi**.

The size of the pixels as projected in both the x and y directions are calculated using equations 5 and 6. Two contour plots are made on the same set of axes with this data. The parameters nx and ny are used to specify the number of contour levels in the respective directions.

Calling Sequence

Result = Footprint,Scene,nx,ny

Arguments

Scene	**Scene** structure
nx	number of contour levels in the x direction
ny	number of contour levels in the y direction

Keywords

Roi	By setting this keyword, the routine allows the user to select a region by drag-clicking the mouse in the image
NoBeachAngle	By setting this keyword, the routine does the calculation for the image in the **SCS**

Example

```
Scene = Populate_Scene(“c:\Nbcoord.txt”,”c:\Durb_Snap.jpg”)
FootPrint,Scene,6,8,/nobeachangle
```

Get_Beach

This function accepts the **scene** structure and returns an array containing the longitude, the latitude and the height of the beach profile markers in the file.

Calling Sequence

Result = Get_Beach(scene)

Arguments

Scene	**Scene** structure

Example

```
Scene = Populate_Scene("c:\Nbcoord.txt","c:\Durb_Snap.jpg")
Help,Get_Beach(scene)
IDL returns:
<Expression>    DOUBLE    = Array[59, 3]
```

Get_BeachAngle

This function uses the camera height and the image size together with the pixel co-ordinates and the longitude/latitude co-ordinates of the two points either parallel or perpendicular to the beach to calculate angle of rotation so the beach is parallel to the image x direction. The default for the beach indicators are that they are parallel to the beach. This routine is called by **Populate_Scene**.

Calling Sequence

Result = Get_BeachAngle(Scene)

Arguments

Scene	**scene** structure

Keywords

Perp	By setting this keyword, the user indicates that the beach indicators are perpendicular to the beach

Get_BeachIndicator

This function accepts the **scene** structure and returns an array containing the longitude, the latitude and the height of the two Beach Indicators necessary to calculate the **beachangle**. This function is called by the routine **Get_BeachAngle**. The beach indicators define a line parallel or perpendicular to the beach.

Calling Sequence

Result = Get_BeachIndicator(scene)

Arguments

Scene	**Scene** structure

Get_Camera

This function accepts the **scene** structure and returns an array containing the longitude, the latitude and the height of the camera. This function is called by the routine **Get_Parameters**.

Calling Sequence

Result = Get_Camera(scene)

Arguments

Scene	**Scene** structure

Example

```
Scene = Populate_Scene(“c:\Nbcoord.txt”,”c:\Durb_Snap.jpg”)
Print,Get_Camera(scene)
```

IDL returns:

```
-3388.5300        302746.14        114.31000
```

Get_GCPGround

This function accepts the **scene** structure and returns an array containing the longitude, the latitude and the height of the first three GCPs in the file. This function is called by the routine **Get_Parameters**. These three GCP's are used to rectify the image.

Calling Sequence

Result = Get_GCPGround(scene)

Arguments

Scene	**Scene** structure

Example

```
Scene = Populate_Scene(“c:\Nbcoord.txt”,”c:\Durb_Snap.jpg”)
Print,Get_GCPGround(scene)
```

IDL returns:

```
-3966.8800      -3689.3900      -3872.6200
303199.83       303043.38       302911.47
5.1599998       3.7200000       5.1700001
```

Get_GCPPixel

This function accepts the **scene** structure and returns an array containing the x and y pixel co-ordinates of the first three GCPs in the data co-ordinate file which may subsequently be used to rectify the image. This function is called by the routine **Get_Parameters**.

Calling Sequence

Result = Get_GCPPixel(scene)

Arguments

Scene	**Scene** structure

Example

```
Scene = Populate_Scene("c:\Nbcoord.txt","c:\Durb_Snap.jpg")
Print,Get_GCPPixel(scene)
```

IDL returns:

```
457.00000       569.00000       89.000000
320.00000       192.00000       266.00000
```

Get_MinMax

This function accepts the **scene** structure and returns an array containing co-ordinates of the extreme x and y ground co-ordinate of the corners of the image. This routine is called by the function **Rectify** and other routines that need to scale the rectified image to fit the current window. The co-ordinates are returned as (xo,yo,x1,y1)

Calling Sequence

Result = Get_MInMax(scene)

Arguments

Scene	**Scene** structure

Keywords

NoBeachAngle	By setting this keyword, the user indicates that the calculation must not take into account the beach angle. Measurements returned are hence in **SCS**

Example

```
Scene = Populate_Scene("c:\Nbcoord.txt","c:\Durb_Snap.jpg")
Print,Get_MinMax(scene)
```

IDL returns:

```
      100.94921       112.11929       2193.7685       1190.5136
```

Get_Parameters

This function uses the camera height and the image size together with the pixel co-ordinates and the longitude/latitude co-ordinates of three surveyed GCPs to calculate the focal length, the field of view, the angle of rotation between the ***Scene Co-ordinate System (SCC)*** and the ***Camera Co-ordinate System (CCS)*** and the angle of tilt. This routine is called by **Populate_Scene.**

The routine **Get_Parameters** takes the three GCP co-ordinates and incrementally rotates them about the z-axis for various angles θ. The routine then finds the angle θ for which f_c is the closest for each co-ordinate.

A similar procedure is followed to determine camera roll.

Calling Sequence

Result = Get_Parameters(Scene)

Arguments

Scene	**Scene** structure

Get_Pixel_Size

This function accepts the structure **scene** and the optional **nobeachangle** keyword. If the keyword is set the pixel sizes returned are those for the image with **beachangle** equal to zero else the pixel sizes returned are those for the image with the beach parallel to the image x axis.

Calling Sequence

Result = Get_Pixel_Size(Scene)

Arguments

Scene	Structure containing the parameters necessary to rectify oblique image

Keywords

NoBeachAngle	By setting this keyword, the routine does the calculation for the image in the **SCS**

Example

```
Scene=Populate_Scene("c:\Nbcoord.txt","c:\Durb_Snap.jpg",/perp)
```

Get_Tide_Height

This procedure accepts the structure returned by Read_Tidal_Data and the parameters **year**, **month**, **day**, **hour**, **minute**, and **second** and returns the tide height in meters for that specific time.

Calling Sequence

Result = Get_Tide_Height(Struct,Year,Month,Day,Hour,Minute,Second)

Arguments

Struct	Structure returned by Read_Tidal_Data
Year	The year from which the height should be read from
Month	The month from which the height should be read from
Day	The day from which the height should be read from
Hour	The hour from which the height should be read from
Minute	The minute from which the height should be read from
Second	The second from which the height should be read from

Example

```
TideStruct = Read_Tidal_Data("c:\C1.tmm")
Print,Get_Tide_Height(TideStruct,1998,3,18,9,28,48)
IDL prints
     1.02000
```

Get_Transect

This function accepts the structure **scene** and returns an array containing beach height and position along a transect which is perpendicular to the image x-axis and passes through the point specified by the parameters **latitude** and **longitude**.

Calling Sequence

Result = Get_Transect(Scene,latitude,longitude)

Arguments

Scene	Scene structure

Keywords

NoBeachAngle	By setting this keyword, the routine does the calculation for the image in the **SCS**
Device	Set this keyword to have the x co-ordinates returned in device co-ordinates

Example

```
Scene=Populate_Scene("c:\Nbcoord.txt","c:\Durb_Snap.jpg",/perp)
TideHeight = Get_Transect(scene,-3684.95,303000.54)
Plot,TideHeight(0,*),TideHeight(1,*) ), title="Transect
Plot",ytitle="Height (m)",
```

Perspective_Grid

The procedure **Perspective_Grid** displays the oblique image with a superimposed perspective grid. Each square cell in the grid has a side of length *len* (in metres).

Calling Sequence

Perspective_Grid,Scene,len

Arguments

Scene	**Scene** structure
Len	length of one side in meters

Example

```
Scene = Populate_Scene("c:\Nbcoord.txt","c:\Durb_Snap.jpg")
```

Plot_Tidal_Data

This procedure accepts the structure returned by Read_Tidal_Data and the parameters **year**, **month**, **day**, **hour** to specify the year, month, day, hour from which to start the plot of the tide height. The parameter, **nhours**, the number of hours of data to be plotted.

Calling Sequence

Plot_Tidal_Data,Struct,Year,Month,Day,Hour,Nhours

Arguments

Struct	Structure returned by Read_Tidal_Data
Year	The year from which the data should be plotted from
Month	The month from which the data should be plotted from
Day	The day from which the data should be plotted from
Hour	The hour from which the data should be plotted from
Nhours	The number of hours of data to be plotted

Example

```
TideStruct = Read_Tidal_Data(“c:\D1.tmm”)
Plot_Tidal_Data,TideStruct,1998,1,28,0,24
```

Populate_Scene

The function generates and populates the **scene** structure by calculating and storing the various parameters in the scene's tag locations using the data co-ordinate and file names.

Calling Sequence

Result = Populate_Scene(FileName,ImageName)

Arguments

FileName	the data co-ordinate file name whose format is specified above
ImageName	the image file name

Keywords

Perp	By setting this keyword, the user indicates that the beach indicators are perpendicular to the beach
NoBeachAngle	By setting this keyword, the user indicates that the beach angle should not be calculated

Example

Using the data co-ordinate file for North Beach *NBcoord.txt* and the image *Durb_Snap.jpg*, the scene structure is generated:

```
Scene=Populate_Scene("c:\Nbcoord.txt","c:\Durb_Snap.jpg",/nobeac
hangle)
Print, Scene
```

IDL prints:

```
{79.610492   39.150988   1079.8586   304.30936  3.5500178  -
3388.5300 302746.14  114.31000 c:\durb_snap.jpg  c:\nbcoord.txt
0.00000}
```

Populate_Scene should thus be called in an initialisation procedure before any other routines are accessed.

Read_Parameters

This function accepts the **scene** structure and returns a structure that contains all the records of the **data co-ordinate file**. This function is called by all the routines that access the data stored in the **data co-ordinate file**.

Calling Sequence

Result = Read_Parameters(scene)

Arguments

Scene	**Scene** structure

Example

```
Scene = Populate_Scene("c:\Nbcoord.txt","c:\Durb_Snap.jpg")
Help,/struct,Read_Parameters(scene)
```

IDL returns:

```
** Structure <15339e0>, 7 tags, length=3304, refs=1:
   NAME            STRING     Array[75]
   LON             DOUBLE     Array[75]
   LAT             DOUBLE     Array[75]
   HEIGHT          FLOAT      Array[75]
   XPIX            FLOAT      Array[75]
   YPIX            FLOAT      Array[75]
   NOTE            STRING     Array[75]
```

Rectify

The **Rectify** function returns a rectified image of the whole/part of the input oblique image. The returned image is the same size of the original tilted image. This function is passed the **scene** structure. If the keyword **roi** is set then the *region of interest* (*roi*) can be selected by drag-clicking a rectangle. Checks are made to exclude points which are too close to the horizon.

Rectification is achieved by taking each pixel in the *roi* and translating it for the roll. Then the corresponding positions in CCS are calculated and scaled to fit the image. These values are stored in an array and are interpolated using the IDL routine **TriGrid** and **Interpolate** to produce the output image.

The new image thus represents an ortho-photographic image, which is no longer distorted by perspective foreshortening. This rectified image is required to sallow comparisons with the models, which are performed in rectilinear geometry.

Calling Sequence

Result = Rectify(Scene)

Arguments

Scene	**scene** structure

Keywords

Roi	By setting this keyword, the routine allows the user to select a region by drag-clicking the mouse in the image
NoBeachAngle	By setting this keyword, the user indicates that the beach angle should not be included in the calculations, that is the image returned will be in **SCS**
White	By setting this keyword, the user indicates that the background of the image should be white instead of the default black

Example

```
Scene = Populate_Scene("c:\Nbcoord.txt","c:\Durb_Snap.jpg")
Image = rectify(Scene,/nobeachangle,/white)
Tv_Color,Image
```

Read_Tidal_Data

This function accepts the name of the file containing the tide data and returns a structure that contains the field **Time**, which contains the time in seconds since midnight on the first of January 1998, and the field **Height**, which contains the tide height in meters. The records in the input file have format as supplied by the SAN Hydrographic Office.

Calling Sequence

Result = Read_Tidal_Data(FileName)

Arguments

FileName	Name of file containing the tide height data

Example

See Plot_Tidal_Data for an example

Show_GCP

This routine displays the image with a superimposed black circle marking each GCP together with its label.

Calling Sequence

Show_GCP,Scene

Arguments

Scene	**Scene** structure

Keywords

All	By setting this keyword, the routine shows the position of all surveyed markers
Skip	By setting this keyword, the routine show only every *n*'th label for the beach profile marker where n is the number assigned to the keyword

Example

```
Scene = Populate_Scene("c:\Nbcoord.txt","c:\Durb_Snap.jpg")
Show_GCP,scene
```

XGround

XGround is an interactive procedure that displays the pixel, ***CCS*** and ***SCS*** co-ordinates of a particular location on the image. These co-ordinates are displayed in a frame to the right of the image window when the left mouse button is clicked. The procedure remains in this loop and terminates when the right mouse button is clicked. **XGround** uses the **scene** structure as the input parameter.

The user moves the cursor about the displayed image and left clicks on a particular point. The device co-ordinates are firstly converted to image co-ordinates where (0,0) is in the centre of the image.

Using the values for the image co-ordinates, Z_c, f_c and τ, we can calculate the y and then the x ground co-ordinates in the ***CCS***. These co-ordinates are printed to the screen, followed by the ground co-ordinates in the ***SCS***.

Calling Sequence

XGround,Scene

Arguments

Scene	**scene** structure

Example

```
 Scene = Populate_Scene("c:\Nbcoord.txt","c:\Durb_Snap.jpg")
      XGround, Scene
```