Automatic Location of Human Faces by Machine

© Graham Robertson

Thesis submitted for the degree of

Doctor of Philsophy

at the

UNIVERSITY
of
GLASGOW

Department of Electrical and Electronic Engineering

July 1994

Acknowledgements

To my wife **Tracy** whom I love and adore

My eternal gratitude goes to everyone who helped make this thesis possible. I would like to extend my very special thanks to:

Brian Rosner, Vivienne Provan, Vera Petzold and Christine Dixon

Glory be to God

"Trust in the Lord with all your heart and lean not on your own understanding;
in all your ways acknowledge him, and he will make your paths straight"

Abstract

This thesis investigates and develops new methods for locating a human face in a computer image. The need for such a technique has long been recognised in many automatic face processing applications such as face recognition or facial image compression. Other studies have attempted face location and the related task of face feature location, with varying degrees of success, on face images captured in controlled conditions; so that the face is on a plain background or is of a fixed size. The result of this study has been the development of a location system that can cope with background clutter, variable size images, slightly rotated faces and changes in overall lighting. This improvement in face location technique has been achieved without an increase in processing time.

The overall system concurrently finds possible eyes, noses and mouth in the image and a control system combines these feature proposers into the complete face location system. The first part of the system consists of robust preprocessors based on previously identified intensity signatures that indicate the possible presence of the desired facial features. Roughly, these signatures are continuous vertical troughs in intensity for noses, continuous horizontal troughs in intensity for mouths and multi-direction troughs in intensity for the eyes. These preprocessors reduce the image search space without rejecting the true facial features being searched for. This reliable reduction in search space is the main contributor to the efficiency of the entire system. The second part of the system analyses the intensity gradient directions on features using statistical methods that were previously trained on many manually located facial features. This technique, called the PRODIGY, combines probability density functions of real and false features in a likelihood function. This function gives an output for each feature pertaining to the likelihood of it being a real facial feature or a distracter (false) feature. The control system combines the possible features with a statistically controlled spatial model of how the individual features are related to each other. It then selects the most likely set of proposed features as the location of the face in the image.

The overall system proved to be successful on a high proportion of face images and the usefulness of the technique is demonstrated by a simple but effective face recognition system.

Contents

List of Figures

List of Tables

1. Introduction

This thesis is concerned with automatic location of human faces by machine. Previous work in this general area has concentrated upon automatic face feature location, in which case a computer finds facial features such as the nose, mouth, chin, eyes, hair, etc. Two major examples of work of this kind are Kanade (1977) and Tock (1992). The present study focusses primarily on automatic face location, whereby a computer or machine finds the position and size of a human face within a scene containing a person. Incidentally, techniques which concern automatic face feature location are also developed.

It is important for many face processing applications that the location of the face in a computer image is known. The most common of automatic face processing applications, face recognition, depends on such location. For a system to distinguish between two face images it is necessary for it to know the position and size of the face and some of its facial features. In the work of Kanade (1977), Sutherland et al (1992), and Jia and Nixon (1992) the face is located before the task of recognition commences. Other face recognition techniques such as WISARD (Stonham 1986) and principal component analysis in the case of Turk and Pentland (1991) do not involve automatic face location, but the systems are essentially presented with faces that are centred in the image or are of a fixed size. This can be thought of as presenting the system with a pre-located face. In addition to face recognition the work in this thesis is motivated by many other applications of face processing, some of which are given towards the end of this introduction.

1.1 Purpose

For most people it is a trivial task to find a face in a given picture but for a computer, being presented with a list of numbers, this is an onerous task. The face is embedded within the list of numbers and may occupy a small proportion of the numbers or cover almost all of them. Any face detector will be distracted by background clutter much of which can have similar characteristics to the face. The purpose in this research is to design a face location system that minimises the number and magnitudes of the constraints required of an input face image. In our context a constraint is a restriction placed on the image acquisition conditions, the expression of the person, or the position of the person used in the image. Many systems demonstrated in the past have depended on images acquired under a number of constraints. Table 1 shows a list of the categories and types of constraints that others often require of face images.

CATEGORY	SUB-CATEGORY	TYPE OF CONSTRAINT OFTEN USED
Translation	scale	The person is seated at a fixed distance from the camera
	position	The face is in a fixed position, possibly positioned using cross hairs on the viewfinder
	orientation	Camera and subject oriented vertically
Pose	occlusion	Full face seen
	rotation	Head pointing straight on to the camera
	face tilt	Eyes level with camera
Lighting	level	Constant lighting level from image to image
	direction	The lighting across the face should be even
Noise	camera	The same camera should use from image to image
	intensity resolution	256 grey levels
	clutter	Plain background
Artefacts	coloured	All white
	moustaches	No moustaches
	beards	No beards
	glasses	No glasses
	sex	All male
Expression		Straight, expressionless face

Table 1: *Common constraints placed on images being used for face research*

Kirby et al (1990) who performed image compression on faces apply most of the constraints in table 1. Craw et al (May, 1992) have mentioned the use of constraints on expression, artefacts, some noise, lighting and pose. The face location techniques in this thesis are specifically designed to significantly reduce the constraints on position, scale and background clutter and to do this with computational efficiency.

1.2 Approach

Face location and face feature location systems can usually be separated into two functional sections. This first part consists of individual facial feature locators. The second part is the control system that spawns each facial feature locator at the appropriate time. The control system gathers evidence from each detector to determine the most probable location of the face. The system described in chapter 5 uses this model for face location. (See chapter 2 for a further discussion on control systems.)

As already mentioned, the work in this thesis relates closely to theses by Kanade (1977) and Tock (1992). Kanade was one of the first to attempt automatic location of facial features on face images. Kanade's system was simple but effective. The control system followed a fixed sequence of events. This is known as the serial approach. Most of the location used integral projection; a technique that projects the contours of the face onto a one dimensional graph. The program then analyses the graph for specific facial feature patterns. The techniques Kanade described depend on the fact that each image is of a large face on a plain background roughly centred in the image. Tock's (1992) work built upon Sheperd's (1986) police 'mug-shot' retrieval system. Sheperd's system used 37 manually located points, on one thousand face photographs, for face recognition experiments. Tock endeavoured to automate the location of these 37 facial feature points. In Tock's research he noted the weakness in a sequential control system (cf. Kanade, 1977) in that if it fails to find any one feature the rest of the system breaks down, and no further features in the sequence are located. Therefore he developed an expert system for the control structure that 'runs' the facial feature detectors in varying orders depending on the information gathered so far. The failure of a single feature detector does not cause the whole system to fail. This is because another feature detector may find it or predictions from other detectors will assist in location.

The system developed in this thesis for face location has a control system based on multiple feedback between facial feature locators. This is followed by a statistical method that selects the most likely position of the face on the image. This approach maintains the computation efficiency of Kanade's serial approach and introduces flexibility as in Tock's approach. The purpose of the research is to locate faces irrespective of background clutter, face size or position. Therefore, the system cannot make use of integral projection to make an initial estimate of the location of the face, neither can head outline locators be used since the background could be the same intensity as the hair or skin. As a result, one approach adopted was to use internal facial features such as the nose rather than external features like the ears. This is because the image background does not affect these features. To maintain flexibilty to the size and position of faces a second approach was to design scale and position independent preprocessors.

The facial features chosen as for the automatic face location system were the nose, eyes, and mouth. Locating the nose, although novel, is appropriate because it is a relatively ridged facial feature, not usually occluded and does not change significantly with expression. The eyes and mouth were chosen because they also are not usually occluded. The peak and trough preprocessors, described in chapter 3, propose positions of the eyes, nose and mouth on the image. These preprocessors proved to be scale and position independent and computationally efficient compared to other preprocessor techniques such Yuille et al's (1988). The nose and mouth versions of the preprocessor give the coordinates and size of each proposed facial feature.

The preprocessors described above produce many proposed facial feature locations. The approach chosen to determine which of the features is the 'real' one was to apply a technique that would attach a confidence value to each feature. This allows the control system to make valued judgements on the features without deleting poorly located features at an early stage. The process developed to calculate the confidence values is the PRODIGY (proportions of gradient directions, Robertson and Sharman, 1992, see appendix 8.5) technique which statistically analyses the directions of intensity change on the proposed facial features. The strength of PRODIGY is that it is able to cope with change in scale and overall lighting.

1.3 Test results and accomplishments

The techniques developed in the research were tested mainly on a specifically designed bank of faces as described in appendix 8.2, and on a separate independent test database of unconstrained faces. The test showed that the technique was successfully independent of scale[1], position, small head rotations, absolute lighting level, camera, background clutter, digitising equipment, gender and faces with glasses. It is not independent of faces with beards, moustaches, unusual lighting or of significantly rotated or tilted faces. The program to locate faces takes about 1 minute on a 20 MHz 386 or 15 seconds on a 33 MHz 486. Tests with the face location program on the fronto parallel faces in the face bank showed that it successfully located 86% of the faces.

By constrast Kanade's (1977) technique is not independent to background clutter or scale. He reported that his technique located 75% of faces in his face bank successfully. Tock claimed that his system found 86% of faces in his test set, all of which were full face images of a 'reasonable size'. He claims some independence to clutter and scale and the program is reported to take upwards of 5 minutes on a SUN 4. This figure needs to

[1] Independent of scale for images with faces over 20 pixels wide.

be put in context because Tock's system is a facial feature location system as compared to the present study, which involves face location.

1.4 Applications

A prominent application for the work described in this thesis is that of face recognition. There have been numerous researchers working on this problem: Nakamura et al (1991), Huang et al (1990), Gallery et al (1992), Hancock (1990), Rickman et al (1992), Sutherland et al (1992), Allinson et al (1992), Jia et al (1992), Akamatsu et al (1992), Turk and Pentland (1991). Face recognition has uses in computer surveillance of buildings, controlled entry systems, user authentication for computers, or mug shot retrieval (Sheperd, 1986). Face recognition is, however, only a branch of computerised face processing research and is only one consequence of face location.

Automatic face processing can be of use in cosmetic applications such as surgery, hairdressing and make overs. Once a system has located a face and its facial features then it can test changes in facial characteristics, hairstyle[2] or makeup. The operator, customer or patient can then verify the new appearance before enabling the start of any procedure.

Combining automatic face processing with location of the whole human body opens up new avenues in the world of entertainment. If a system electronically tracks the motion of the human body and facial features then it could also animate a cartoon or puppet to imitate the movements.

Video phones and the standards they use for compression are already available (CCITT, 1989). These standards allow for improvements in the compression technique (Trew et al, 1992). The system can achieve this by retaining more detail over certain areas of the image than others. An automatic face processing machine would locate the face and its features in the scene and pass this information to the compression algorithm. The algorithm can then concentrate efforts more on the detailed facial features such as the eyes and mouth and less on the nose and cheeks.

[2] New Image Salon System II, P.O. Box 30, Cranbrook, Kent, TN17 1JQ - This system is not totally automatic. It requires the user to locate the face for it.

Face location has potential in picture enhancement of surveillance videos of crimes. The videos are often of poor quality despite a five second clip having over 100 pictures of the thief. The system would locate the face in each frame and process them together to produce a higher quality picture of the criminal.

Face location also has applications in photography. For example if a system locates the face in an image then it can also control the camera to 'focus on', 'zoom in on' or track the face. Often the processing stage of photograph production reveals that the picture suffers from 'red eye'. In this case then a face location system could locate the eye and process it to normalise some of the 'red eye' effect.

1.5 Format of the thesis

This thesis has 6 chapters. Chapters 1 and 2 describe the problem and the background to the research. Chapters 3 and 4 describe techniques for locating facial features and then chapter 5 describes techniques for locating the whole face. Experimental results are displayed at the end of chapters 3, 4 and 5. Finally chapter 6 discusses observations and conclusions.

The background chapter, chapter 2, begins with an overview of computer vision. It discusses some of the approaches to vision problems and argues for the approach taken to solve the face location problem. The importance of face location especially in the context of face recognition is then discussed. This is followed by a survey of recent work in automatic face location, including a discussion of available control structures. The chapter finally examines the testing of face location systems and describes the face bank created for designing and testing our techniques.

Others have used peak and trough preprocessors to aid facial feature location. Chapter 3 presents an efficient algorithmic method of locating peaks and troughs. Techniques for proposing the location of eyes, noses, and mouths based on these preprocessors are described. The chapter also examines further adaptations that cope with uneven lighting on the subject.

Chapter 4 presents the PRODIGY technique, which uses the output of the peak and trough preprocessor as part of its input and gives a confidence value relative to the likelihood of a predicted facial feature being a real facial feature.

Chapter 5 draws the research together by describing a control system to combine the individual feature detectors into a single face location system. It consists of a multiple feedback technique followed by a statistical springs method. This chapter displays examples of the output of the face location system. It then presents a simple face recognition system that demonstrates the usefulness of the face location system.

Chapter 6 draws conclusions on the results given in chapter 3, 4 and 5 and also discusses the viability of the techniques and comments on future development. In closing the chapter discusses the extent of the research and its merits.

2. Background

Automatic face processing is part of the more general area of computer vision. This chapter begins by discussing some aspects of computer vision and lists some general methods of tackling the vision task. This is followed by a detailed survey of recent methods in automatic location of faces. The discussion includes location primitives, higher level control structures and testing methodologies.

First, a brief history of automatic face processing follows which shows how the development of computers has enabled different aspects of the field to be researched. In the early days data was obtained manually. This data consisted of spatial distances measured from photographs (Goldstein et al, 1971, Kaya and Kobayashi, 1972). Later digitisers became available which could convert photographs into an array of pixels in several seconds. However, because computers were not powerful most of the algorithms were programmed in machine code and consisted mostly of integer operations (Kanade, 1977). In recent years researchers have had access to frame grabbers and faster processors enabling the use of complex statistical analysis and morphological operators. As this research is being concluded, colour digitisers and cameras are becoming readily available along with powerful computers capable of processing these images (Akamatsu et al, 1992).

2.1 Computer Vision

Vision research covers a wide range of topics and applications. Rosenfeld (1992) describes Computer Vision as follows:

> *"The general goal of computer vision is to derive information about a scene by computer analysis of images of that scene ... a computer vision system attempts to (partially) describe the scene as consisting of surfaces or objects."*

The computer usually obtains these images from a frame grabber by digitally sampling a picture from a TV camera at regularly spaced grid points. The frame grabber passes the image as an array or matrix of pixels to the vision system. Each element of the matrix indicates the colour or intensity of the scene at one of the grid points. If *I* represents the image matrix then:

Intensity or colour at pixel $i, j = I_{i,j}$ (1)

where i is the row number of the matrix and j is the column number.

There are many approaches to tackling the 'vision problem'. Large vision systems often combine approaches that can be split into three general areas. In these areas the approaches:

1. Analyse the way that humans can visualise scenes and simulate the human vision system on a computer. Many assume that because we know that the human system works then simulating the human visual system will produce a working computer vision system. Neural networks are a class of techniques based on the human neural system. These techniques have the benefit that they can often adapt to new environments. The human neural system is, however, not yet fully understood and this is reflected in the limited capabilitiy of some of these techniques.
2. Treat computer vision as an inverse graphics problem. This entails trying to mathematically calculate what arrangement of background and objects would produce a given scene. This is often called model based vision. If an accurate model of the objects is available then these techniques would be robust. However, appropriate models are difficult to find unless the scene is considerably constrained. These techniques can also be computationally explosive and often require operators such as genetic algorithms (Goldberg, 1989 and Robertson and Sharman, 1990, see appendix 8.4) or simulated annealing to navigate the huge search spaces.
3. Treat each problem as a special case and design an algorithm for each task based on statistical and visual observations of the signals produced by the objects in the scene. An example of such a technique is the bar-code system. The bars on products are designed specifically for the vision system and consist of high contrast black and white stripes that enable simple transformation to a one dimensional signal. Synchronisation bars at the beginning of the object also encode the size of the bar-code. These techniques have the advantage that they are computationally efficient but lack adaptability to unforeseen environments.

The face location problem described in this thesis uses the third approach. Observations in the changes in intensity resulted in the peak and trough algorithms (see chapter 3). Statistical observations about the gradient directions on face features enabled the development of the PRODIGY technique described in Chapter 4. One reason for this approach is the computational efficiency of the technique which gives an acceptable response time on the development equipment. Another is because the human face deserves special attention due to the importance of face processing applications.

Others have demonstrated real time vision systems based on observations. One example of this is in a system where a robot arm picks up a cube[3]. The cube is white on a black table and the algorithm finds it by thresholding the image. The location of the bright part of the thresholded image is the two dimensional image location of the cube. The system translates the 2D coordinates into 3D coordinates and gives the robot instructions on how to pick up the cube. Webster (1992) demonstrated a system where a robot was taught to play golf. To find the golf ball the computer scans the image from the bottom upwards until it finds a bright spot and assumes the spot to be the golf ball. Another demonstration showed a system for tracking the motion of a JCB arm[4]. This system has prior knowledge of where the arm is likely to be (within 10-20 pixels). By fitting a straight line to the output of an edge detector the machine determines the actual position. These three applications show the power of using observations for real time vision systems. However, although the three systems described above have been shown to work they are all likely to fail when taken out of laboratory conditions. This could be due to poor lighting, background clutter or poor contrast. The systems all require highly constrained inputs and so are not robust in the unconstrained 'real' world.

The use of statistics to test and quantify any observations will inevitably improve the strength of the systems. For this reason the statistical PRODIGY technique was developed. The face location system in the present study uses statistics to improve the stability over a wide range of conditions. Even so the range of inputs used to design and the test system, understandably, puts limits on the flexibility of the vision system. It is, however, reasonable that assumptions must be made to facilitate any progress. Rosenfeld (1992) says that valid assumptions can and should be made to confine the vision problem:

> *"In fact, in many situations only a partial description of the scene is needed, and such descriptions can often be derived inexpensively and reliably."*

[3] James Little, University of British Columbia

[4] Computer Science Dept, University of British Columbia

Techniques that use observations will usually include assumptions, which result in information being disposed of or ignored. Although errors will occur this approach has led to a face location system that is demonstratable and testable with unusual inputs.

2.2 Importance of face location

From the early days of automatic face processing research it has been clear that a system capable of locating faces and measure facial features is necessary. In 1972 Goldstein et al designed a method to test the possibility of computerised face recognition. In their tests 34 features on 768 photographs were manually coded into a computer. Kaya and Kobayashi (1972) also conducted an interesting study into the possibility of automatic face recognition and manually measured face features for later statistical analysis. Measuring many more than 1,000 faces is verging on the impractical and supports the need for a system to automatically locate and measure faces.

Gallery et al (1992) also support the need for face location by emphasising that face identification (recognition) naturally divides into 'face location' and 'face classification' stages. Examination of many reported face recognition systems reveals this division. In Sutherland et al's (1992) description of a system for face recognition using Vector Quantisation (VQ) the first step of many involves locating the eyes, bridge of the nose, nostrils, mouth, chin, hair and face. The system uses these locations as reference points to compare similar features on different face images. In the description of one the earliest attempts at automatic face recognition by Kanade (1977) the majority of the work reported was concerned with face and feature location.

Many modern face recognition techniques do not use the geometry of the face but rather the texture. These texture techniques are often enhancements to template matching. Some claim that with these methods there is no need to locate the face. Principal component analysis used by Turk and Pentland (1991) is such a technique. Another such system that uses binary templates is called WISARD and was demonstrated by Stonham (1986). In these systems no active effort is made by the machine to locate the faces but closer examination reveals that the stimulus to the systems were usually faces of the same size and position. This is in effect, as we noted in chapter 1, presenting the system with a pre-located face. In cases where the small faces or poorly positioned faces were presented the systems showed poor performance. Craw et al (September, 1992) and Akamatsu et al (1992) have both presented more recent results on using texture to perform recognition and have stressed the need to locate the face if performance is to be increased.

The importance of face location is supported by its many other applications as described in section 1.4.

2.3 Face location methods

This section describes some of the techniques that are often used for face location and face feature location. The face differs from many synthetic objects in that it does not have sharp edges but consists of smooth intensity changes and contours. All faces consist of the same basic components but their size and geometrical relationship to each other cannot be exactly predicted.

The techniques fall into two areas, global and local locators (Craw et al, 1992, May 1992). Systems normally use global locators to estimate the location of a face or feature within the whole image. Local locators refine an initial estimation often finding sub features.

2.3.1 Deformable templates

To cope with changes in overall size and shape of faces and facial features, a technique called deformable templates has often been used. A deformable template is a mathematical model of a face feature and will have numerous parameters that can be adjusted so that it can fit any instance of a particular face feature. Obvious parameters are overall size, vertical to horizontal size ratio and angle of presentation. Techniques often include non geometric parameters such as the colour, spatial frequency and gradient direction. A core part of the deformable template technique is a method of comparing the model with the image to produce a matching score. This score is used to test whether new values for the parameters have improved the model. Suitable matching techniques can involve template matching or checking edges on the image with lines drawn on the model. New parameter values are usually tried and tested in an iterative style until a comparatively good matching score is achieved. The parameter search spaces often become large and therefore require the assistance of search techniques such as gradient descent or genetic algorithms (Robertson and Sharman, 1990) to optimise these parameters.

The global head outline locator described by Craw et al (May 1992) uses deformable templates. Initially the locator predicts the outline of the head using a model based on the mean shape of the heads in their face database. It then makes adjustments to the model by varying its aspect ratio, position and shape. The shape is described by a set of vectors $[v_1, \ldots, v_n]$ with $v_i \in real$ vectors and $v_1 + \ldots + v_n = 0$.

The locator evaluates a matching score *f*, which indicates how well the shape fits the face in the image. Where

$$f = f_{shape} \cdot f_{edge}$$

f_{shape} is calculated from the statistical distribution of face shapes from the face database. To calculate f_{edge}, the edge score, the system measures the distance from each point on the shape outline to an edge. The system then adds the distances together.

This method described by Craw et al creates new outlines by using simulated annealing to vary the parameters until a 'best match shape' is attained. This technique, although basically model based, clearly applies a statistical analysis of many faces to create shape matching scores. This statistical analysis ensures that only 'faces like shapes' are produced although it was limited by the fact that only male faces were analysed. To create the edge score it uses the observation that there are edges between the head outline and the image background. This observation has the limitation that it may not hold if the head is of a similar intensity to the image background

Yuille et al (1988) described a local method of improving the eye location using deformable templates. The method used the eye template model demonstrated in figure 1.

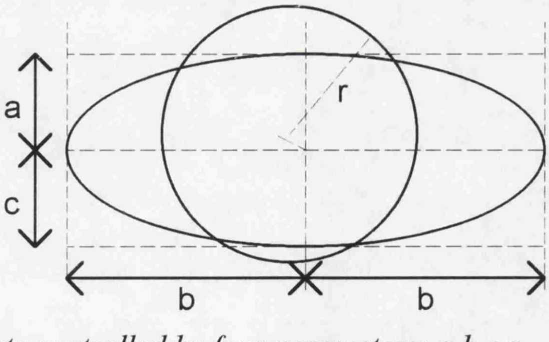


Figure 1: *Eye template controlled by four parameters: a,b,c,r*

The template consists of a circle for the iris and a parabola for each eyelid. The method matches the union between the two regions (i.e. the iris) against dark regions of the image. The remaining part of the parabola region is the white of the eye and the system matches it against bright regions of the image. The procedure also matches appropriate lines on the template to edges on the image and checks the area above the template for dark regions caused by eyebrows. In matching the dark and bright areas the procedure uses peak and trough fields produced with morphological operators (see chapter 3 on peak and trough processors for more details). To optimise the parameters Yuille chose a gradient descent algorithm.

Yuille's local eye location method is based on the observation that the iris is dark and the white of the eyes are bright. Although this observation is correct it led to a method that was insufficient due to a lack of contrast between the whites of the eyes and the skin on the face below the eye. Shackleton and Welsh (1991) noticed this and designed a white enhancing algorithm that darkens the skin below the eyes and brightens the eye whites. They also applied a smoothing operator that makes the edges, the peaks and troughs and the whites effective over a greater area. Shackleton and Welsh apparently designed these enhancements from a second set of observations taken from an examination of failures on new image samples.

Yuille et al (1988) presented another local deformable template method for locating the mouth. The template describes the top of the upper lip by two parabolas and the bottom of the lower lip by one parabola. If the lips are closed then another parabola describes the join between the lips. If the lips are open, the bottom of the top lip, and the top of the bottom lip, are described by further parabola. Again a gradient descent method optimises the system. Yuille et al's mouth detector is typical of most in that it is limited to one or two facial expressions. This is probably due to the inherent difficulty in modelling a feature that has much variability caused by a number of facial muscles.

2.3.2 Intensity variations and integral projection

Examining the intensity variations across a face image (see chapter 3) reveals distinct characteristics at each facial feature. For example, the eyes show up as dark regions and the nose as bright regions. Kanade (1977) pioneered much of the research into detecting intensity variations on faces, although it is important to note that his methods were constrained to photographs of subjects on plain bright backgrounds. Because of this constraint Kanade made the observation that he could locate the top of the head by scanning each raster line until there is a significant fall in intensity. To search for the eyes a rectangular prediction area was derived from the position of the top of the head. Within the rectangle the eyes were found by searching for 'blobs' of dark pixels. He then expanded the blob to encompass neighbouring dark pixels (fusion) and then shrunk it towards the dark centre of the blob which represented the centre of the eye. Kanade located the nose, mouth and chin collectively using a technique called integral projection where the image is essentially transformed into a one-dimensional signal by averaging portions of rows or columns in the image. When using horizontal integral projection, if the system finds three intensity peaks then it assumes the top one is the nose, the middle one the mouth and the lower one the chin.

Jia and Nixon (1992) described a face recognition system that was dependent on face location. They used intensity variation and integral projection in a similar way to Kanade to locate the head outline and predict some facial features. In this case Jia and Nixon photographed faces on a black background. They used vertical integral projecting and their algorithm marks the boundaries of the head on each side where the intensity starts to rise. Their technique predicts an eye region as a trough between the highest peaks of a horizontal integral projection.

2.3.3 Intensity variations and Gradient Direction Techniques

Gradient direction techniques differ from integral projection by being two dimensional. Integral projection reduces data by converting to one dimension whereas gradient direction techniques often reduce the data by quantising it (see chapter 4 for more details).

Craw et al used gradient techniques to trace the outline of the mouth. The technique, which is described further in section 4.1, is a local method that uses observations regarding the gradient directions on the mouth to find the lips.

Nixon (1985) showed that local eye location could be performed as part of a Hough transform. In this technique Nixon defines the iris with a circle and the sclera with an exponential ellipse. Nixon's adaptation of the hough transform makes effective use of the gradient direction information to reduce the effect of noise in the image. In his method, for each edge point on the eye image the hough accumulator array was only increased in the direction of the edge.

2.3.4 Edge based techniques

Although edge based image processing techniques are found in many vision areas they are less common in automatic face processing. This is largely due to the smooth changes in intensity on the human face. Edges can be found around the eyes and mouth and side of the face. However, these edges change unpredictably between subjects. One subject with dark long hair may have strong edges on the side of the face and another with light hair may have no edges at all. Strong edges on dark hair are therefore only useful if another method has already determined that the face has dark hair. In other words the other method must have already located the hair. This reduces the usefulness in edges for initial feature location but indicates that edges are often invaluable in refining facial feature boundaries.

Many have used edge detectors to find the boundary of the chin after location of the mouth. Nevertheless the type of edges found on the chin are subjective and highly dependent of the lighting arrangement. Craw et al (May, 1992) denotes the chin as the first strong edge crossing below the mouth. They proceeded to connect this point to other points located on the jaw line. These other edge points are often found by sending radials out from the centre of the mouth downwards in a 90 degree arc (e.g. as described by Kanade, 1977). This ensures that all edges being searched for have the same direction of maximum gradient. Robertson (1989) introduced a modified edge detector for locating the chin that detects edges and ripple edges. A ripple edge as defined by Robertson (1989) is where the intensity is rising (positive gradient), then intensity continues rising but the rise slows down (decreased gradient), then the intensity starts to rise faster again (increased gradient). A ripple edge is marked at the point where the gradient is at its lowest.

Both Robertson (1989) and Craw et al (May 1992) have shown that a local method of locating the edge of the face is by radiating edge detectors horizontally from the centre of the face until the first strong edge is found. The edges of the face can then be combined with the chin points to make a complete contour. As already mentioned, this method will fail if the hair intensity is the same as the skin intensity.

2.4 Common threads

The techniques described in the previous section (2.3) show the diversity of approaches that others have used in locating face features. Where papers quote results they are usually specific to a particular application or to a particular set of test data (see section 2.6 for a further discussion on testing methodology) and there is no general agreement that any one technique for location of any specific facial feature is best. However, most global techniques are based on observational theory. Common observations are that the eyes are darker that the rest of the face, the mouth produces a dark horizontal line, the chin sometimes produces a curved edge and the nostrils are dark. From these observations it is clear that integral projection is a suitable method for separating features once the system knows the size and location of the face. The PRODIGY technique described in chapter 4 tackles this inadequacy by using statistics to extend the basic observational approach and consequently improves stability over larger sets of test data. Only Craw et al's outline locator has used statistical measures previously to aid in the location process.

2.5 Control Structure

In the context of face location a control structure is a system that connects the various techniques for locating facial features into a single face location program. It is the purpose of the control structure to force the located features to acquire face like spatial separations. This assumes that the control structure knows how facial features are related. In Tock's (1992) control structure he used data from a spatial analysis of 1000 face images that had been manually located and measured. The results of the analysis are demonstrated in figure 2.

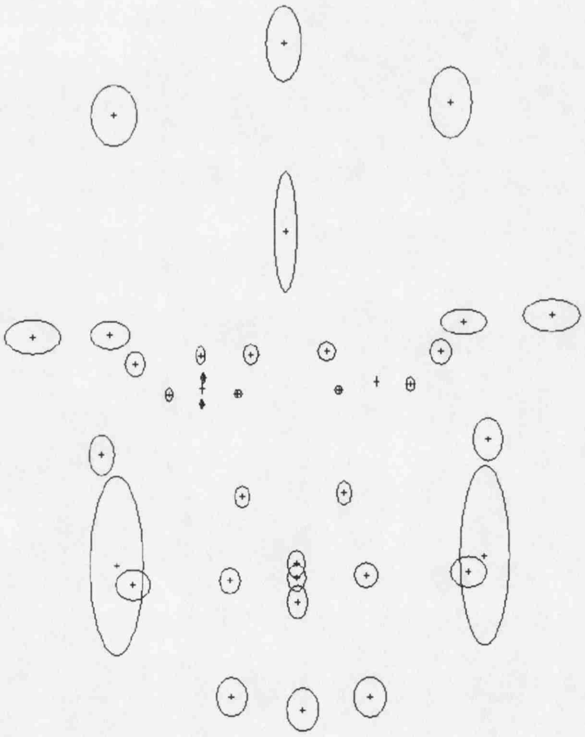


Figure 2: *Statistical distribution of the facial features. This illustration is given by kind permission of David Tock (1992). It shows the statistical distribution of all the facial features in the Aberdeen face database. Each face was spatially normalised about the eye points. The circles around all the other points represent the relative sizes of the standard deviations.*

The results show how face features are spatially related.

Early control structures for face location used the serial approach (Kanade, 1977, Craw et al, 1987 and Robertson, 1989). This approach locates each facial feature in sequence. Craw et al first located the mouth and Robertson the nose. The serial method uses knowledge of the location of the first feature to predict a region for searching for the next feature. That is, it searches for the eyes near to the nose and directly above the edges of the lips. The serial process continues searching for each feature in a set sequence until all the features are located.

The serial process is flawed in that if one feature location section fails then all subsequent ones will also fail. This leads to 'catastrophic' failure whenever a feature detector in the early stage fails. Kanade (1977) reduced the effect of this problem by detecting some of the failures and thereby causing the system to retrace its steps using different parameters. This system is shown in figure 3.

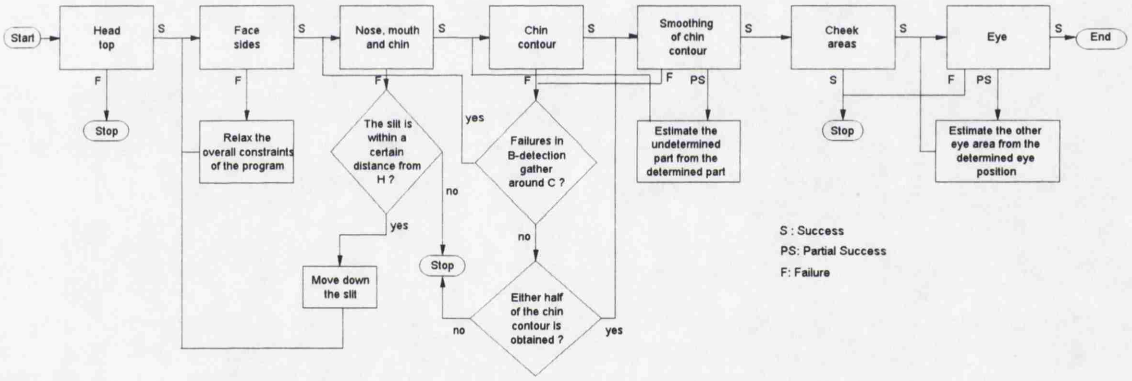

Start
Head top
S
F
Stop
Face sides
S
F
Relax the overall constraints of the program
Nose, mouth and chin
S
F
The slit is within a certain distance from H ?
yes
no
Move down the slit
Stop
yes
Chin contour
S
F
Failures in B-detection gather around C ?
no
Either half of the chin contour is obtained ?
no
yes
Smoothing of chin contour
S
F
PS
Estimate the undetermined part from the determined part
Cheek areas
S
S
Stop
Eye
S
F
PS
Estimate the other eye area from the determined eye position
End
S : Success
PS: Partial Success
F: Failure

Figure 3: *General flow of Kanade's face location technique*

Although Kanade had some feedback in his system if any part of the sequence completely failed then the remaining parts would also fail. The only way to prevent this kind of failure is to design a perfect facial feature location algorithm. This, however, is infeasible.

To overcome the problem of total failure due to an error in one of the locators, it is necessary to design separate feature location algorithms for each facial feature or at least for each key feature (eyes, nose, mouth, head outline). These separate algorithms should be capable of working independently. The control system would then run the locators, influencing their search in various ways depending on the results of others.

Craw, Tock and Bennett named the independent feature location algorithms as *global* techniques and the serial (dependent) algorithms as *local* techniques. Each detector gives a confidence as to the success of its work. They combined these detectors in an expert system to produce a face feature location algorithm. This system spawns off the feature detectors in varying orders and biases the more successful locators when deciding on the final position of the features.

Chapter 5 discusses the control structure for the face location system which was developed for the present study. The system is constructed from face feature location modules that pass information to each other in a multiple feedback loop. This enables each feature locator to reject impossible features, which quickly reduces the number of possible face features. The remaining features are connected by finding the most likely combination of them. The technique has similarities to a springs and template method by Fischler and Elschlarger (1973) described in chapter 5. Our technique increases flexibility by using the springs method but reduces its computational complexity by using the output of the multiple feedback loop.

2.6 Testing face location systems

The testing of a face location system is greatly subjective. A survey of face recognition and face location techniques by Samal and Iyengar (1992) showed that each technique had been tested or designed using a different set of data and the number of faces included in the data sets vary from 6 to 1000. Further analysis of these techniques show that some of them are designed only to function on a particular type of data and so the test sets only contain that one type of data.

In Kanade's (1977) method the image is scanned raster line by raster line from the top downwards and the first line that gives a significant response is the top of the head. This technique actually locates the face in the first two steps. Further examination shows that not only does Kanade’s technique require faces on a plain background but they must also be of a fixed size. These requirements are reflected in his data which consists of 800 faces, 670 of which were successfully analysed.

Jia and Nixon (1992) on the other hand assume that the face is on a black background and search for an increase in intensity to locate the head. Again the face is located in the first step. It is therefore apparent that neither Kanade’s technique, which uses a white background, nor Jia and Nixon’s technique, which uses a black background will function on each others data. It is therefore understandable that both these techniques will fail on all the images designed for this research, as shown in section 2.6.1, because they have cluttered backgrounds.

It is clear from the comments above that it is unreasonable to compare test results directly. The techniques described in this thesis are designed to cope with a wider range of presentations rather than improving the location rate for a particular set of data. An ideal test set would be one that incorporates a large number of people in a wide range of image conditions and face presentations. Some conditions and presentations that could be included are:

- Happy, sad, tired, laughing, frowning, etc.
- Facing forwards, sideways, upwards, etc.
- Lit from the front, left, right, etc.
- Long haired, short haired, bald, bearded, etc.
- Partially occluded.
- Wearing glasses.
- Deformed or blemished.
- Dark skinned, light skinned or somewhere in-between.
- close to the camera, far away from the camera, etc.

2.6.1 Face bank used in the Thesis

As an ideal dataset was not available we designed a data set that encompassed some of these presentations. The dataset, called the '*face bank*', was chosen to allow the face location technique to be designed with a structured range of faces and to enable some basic quantitative analysis of the results. The face bank is, however, limited by the time available to compile the data. The face bank includes eighteen structured pictures of fourteen subjects. The structure of each set of faces is given in table 2 and demonstrated in figure 4.

No	Distance from camera (cm)	Name of presentation	Description
1	20	Full face	Occupies the whole image
2	40	Large face	Face occupies most of the image
3	60	Medium face	Face and shoulders are visible
4	80	Small face	Face is visible as well as much of the background
5	140	Tiny face	Face has the same prominence as the background
6	220	Minute	Very low resolution face
7	40	Profile	Profile of the head
8	40	22 Profile	Head rotated approx. 22 degrees from the camera
9	40	45 Profile	Head rotated approx. 45 degrees from the camera
10	40	67 Profile	Head rotated approx. 67 degrees from the camera
11	140	22 Tiny Profile	Tiny head rotated approx. 22 degrees from the camera
12	140	45 Tiny Profile	Tiny head rotated approx. 45 degrees from the camera
13	140	67 Tiny Profile	Tiny head rotated approx. 67 degrees from the camera
14	40	Eyes closed	Picture of the face with the eyes closed
15	40	Eyes side	Picture of the face with eyes looking to the side
16	40	Smile	Picture of the face with a smile
17	40	Cheesy Grin	Picture of the face with a cheesy grin
18	40	Glasses on	Picture of the face with the glasses on

Table 2: *The structure of presentations used to capture images of each subject*

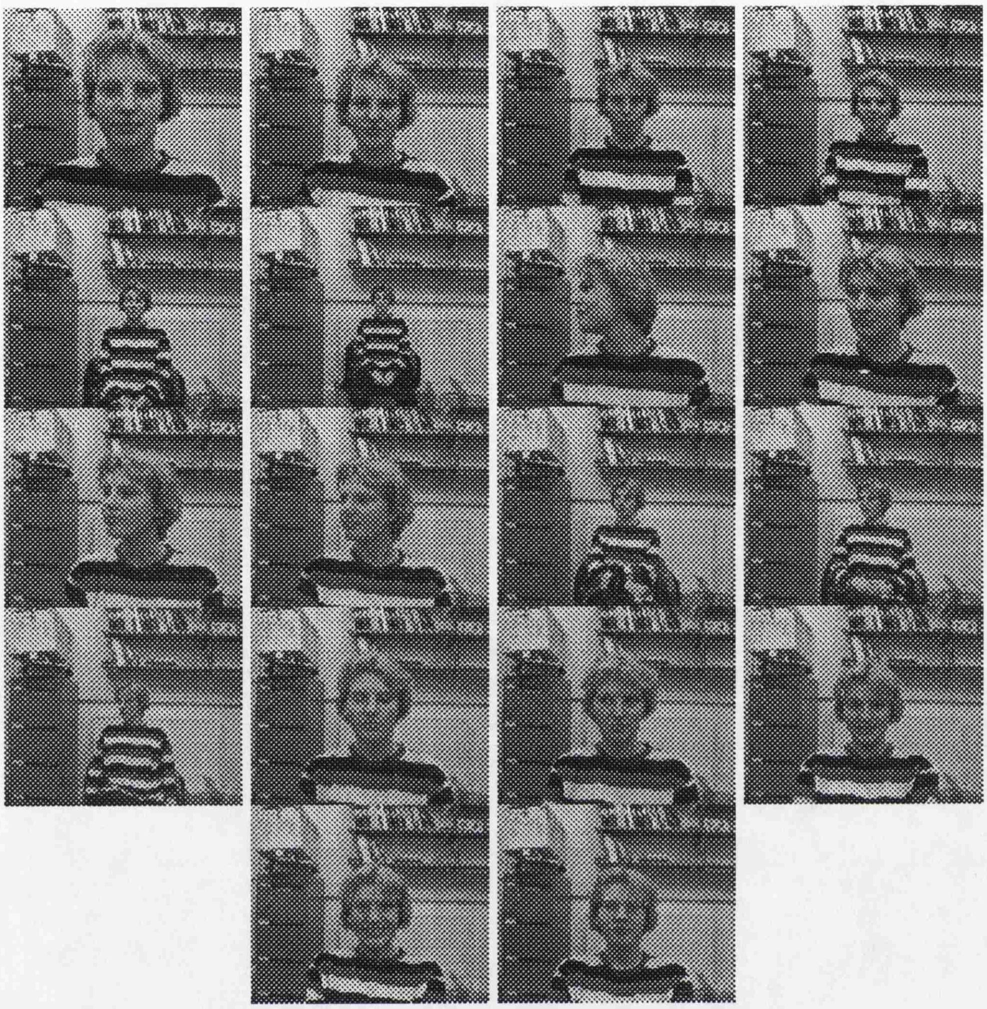


Figure 4: *Eighteen presentations of subject TRACY*

2.6.2 Coding of faces in the bank

To train some of the statistical techniques and enable qualitative tests to be performed on the faces it is necessary for the computer to know where the features are on the faces in the bank. For this reason, each face in the bank was manually coded with 37 points as shown in table 17. The choice of face points is somewhat arbitrary. Tock's (1992) points were chosen to enable facial measurements to be made to aid recognition. Forchheimer and Fahlander (1983) chose points that best represent a face for reconstruction purposes. The points chosen for the present study and shown in appendix 8.2 were chosen to enable the development of algorithms to test the quality of facial feature location techniques. They also reflect the requirements of the algorithms being designed and the type of face data in the face bank. For example, because the face bank consists of mouths of various expressions it was necessary to mark both the top and bottom of both lips.

2.6.3 Analysis of the face bank

To allow quantitative tests to be performed the faces in the bank were measured and the results are displayed in table 3. They show the average size in pixels of the various measurements for the eighteen face presentations.

Image number	Width of head	Length of nose	Width of mouth	Width of eye
1	63.6	28.1	30.3	16.1
2	47.6	21.2	21.9	11.5
3	37.8	15.5	17.3	9.7
4	32.2	13.0	14.3	7.9
5	20.7	8.1	8.2	5.1
6	14.2	5.6	5.6	3.2
7	0.0	18.1	0	0
8	48.4	19.7	17.8	7.2
9	46.7	18.9	13.8	4.1
10	45.8	19.0	11.2	2.5
11	20.4	7.8	6.2	3.0
12	20.3	7.8	5.6	2.0
13	20.2	7.5	4.9	1.5
14	46.8	20.5	20.8	12.8
15	45.9	20.4	21.0	11.8
16	47.3	19.9	23.4	12.0
17	48.6	19.6	26.2	11.9
18	47.0	18.8	22.7	11.1

Table 3: *The length of various face measurements in pixels*

2.6.4 Other test data

It has already been noted that some techniques have only been designed for certain types of data. The techniques that we have developed are designed to function on a wider range of inputs. It is not appropriate to only test the technique on the training data. For this reason, a second set of faces was acquired. The collection of faces was called the '*test database*' of face. The images were downloaded from a larger database of faces in the USA and 152 of the faces were selected at random. The faces have been photographed at many locations and include people of different races. Many of the faces have beards or moustaches, many are smiling or grinning and many of the faces are not pointing directly at the camera. As shown in section 5.5.2 the results on this test set are much lower than the results on the face bank. However, the figures on the test set give a more 'honest' picture of how face location techniques would perform in a general situation.

2.7 Summary

This chapter has discussed the general approach to computer vision and has reasoned that a statistical and observational approach would be an appropriate one for face location. Several examples of face feature location have been analysed which demonstrate the diversity of the techniques currently available. The following chapters make use of some of the attributes of intensity techniques for predicting face feature locations but without restricting them to particular types of image backgrounds as was the case with (Kanade, 1977) and (Jia and Nixon, 1992). Face feature models that encompass the range of face feature shapes are used to define feature boundaries and gradient techniques combined with statistical analysis are used to find confidence levels for the feature locations. Later in this chapter control structures were discussed. These structures usually consist of feedback or statistical methods. These two methods of control have been effectively combined in the control system introduced in chapter 5. Finally it was shown that many types of test data have been used in the past to test face location systems. We have designed a test set of data consisting of 252 pictures that have enabled structured testing of the techniques developed in this thesis.

3. Peak and Trough Preprocessors

The peak and trough preprocessors introduced in this thesis are used to propose the position of noses, mouth or eyes within an image scene. Intensity techniques have been described in section 2.3.2, the preprocessors described in this chapter are based on intensity observations and take an algorithmic approach to locating features from the intensity data.

The preprocessors are designed to find local peaks or local troughs in intensity. Figure 5 shows image 'SHEILA1' with an eye level intensity graph superimposed. From the graph in figure 5 the nose and eyes can be observed as peaks and troughs at the appropriate spacial locations. The nose is the local peak in the middle of the graph and the pupils of the eyes are local troughs to the left and right of this peak.

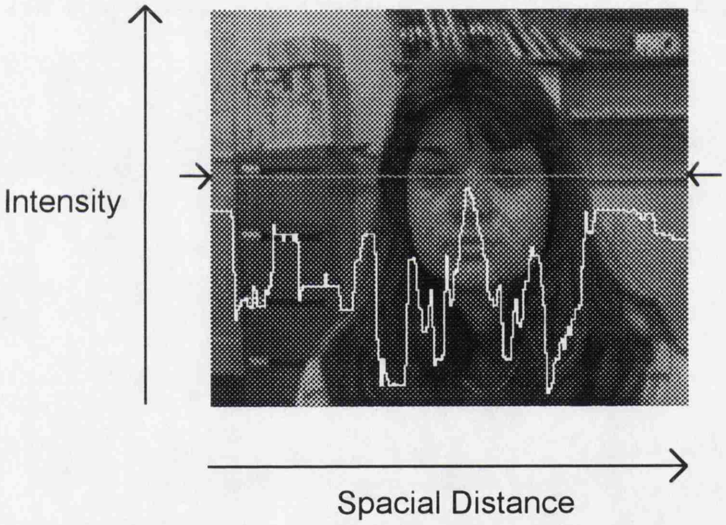


Figure 5: *Intensity levels across the image at eye level. The nose and eyes can clearly be seen in the graph.*

Kanade (1977) noted that various parts of the face produce characteristic horizontal intensities and designed a technique for scanning horizontal intensity graphs and comparing them with the graphs of previously analysed faces. This technique differs from general purpose object locators that make use of edge detectors to find object boundaries. The approach is consistent with comments made in chapter 2 that faces are usually oriented in one direction, have distinct surface characteristics and do not have edges that are particularly consistent from one image to another.

Yuille et al (1988) used peaks and troughs (or valleys) to preprocess images to produce three fields.

- Edge field
- Peak field
- Valley field

The peak and valley fields consist of a transformation of the whole image[5]. The transformation is performed by a combination of morphological opening and closing operators (Serra, 1982). Yuille et al uses the peak and valley fields for eye and mouth location. The same method was subsequently used by Shackleton and Welsh (1991) and Craw et al (1992). The preprocessing was used as part of the a deformable template method (see section 2.3.3) and is mathematically intensive[6]. Yuille et al's method also requires an intial prediction of where the facial feature is in the image. If the initial prediction is out by more than about 20 pixels the method is likely to fail. This makes Yuille's technique a local one where as the ones presented in this chapter are global techniques. This also makes the two techniques complementary in that Yuille's technique can be used to refine the output from the preprocessor described in this chapter.

The technique for finding peak and troughs, presented in this chapter, is computationally efficient[7] and simple to implement. It does not provide as much information as the morphological type (i.e. it does not give any indication of the intensity of the peaks or troughs) but the preprocessors have been optimised around finding noses, eyes and mouths and therefore only search for characteristics associated with these features. The preprocessors also link information together to detect ridges in intensities in the likely directions of the facial features. In the final parts of this chapter, methods are described for optimising the technique to cope with changes in illumination.

[5] The image is mapped to another with the same size and spatial resolution. (i.e. a pixel in the transformed image corresponds to a point on the original image)

[6] Yuille et al states that their process takes about 5 minutes on a Sun-4

[7] By nature much faster than the morphological method.

3.1 Peak and trough preprocessors for feature detection

This section describes the peak and trough algorithms and explains how they can be used to propose the location of eyes, noses and mouths.

3.1.1 Peak Preprocessor for nose location

The peak pre-processor proposes possible noses by finding a line of peaks in a vertical direction. The algorithm is shown in figure 6

```
Filter the image in a horizontal direction;
Filter the image in a vertical direction;
Locate the local peaks along each horizontal raster line
for (each peak in the image from the top)
{
    set the V-line = the location of the peak;
    while (there is also peak on the image on the raster line below
    the bottom of the V-line either on the same column or one pixel to
    the left or right)
    {
        add the location of the new peak to the current V-line and
        delete the peak from the image;
    }
    record the V-line in the list;
}
```

Figure 6: *Algorithm for proposing noses using the peak preprocessor*

The filters in the algorithm are used to smooth out any spurious noise in the image or blemishes on the face. Two filters are used, one in the horizontal direction and the other in the vertical direction as shown in equations 2 and 3.

$$\text{If } H_s \text{ is even } N_{i,j} = \sum_{t=-(H_s-1)/2}^{(H_s-1)/2} I_{i,j+t}, \text{ if } H_s \text{ is odd } N_{i,j} = \sum_{t=-(H_s)/2}^{(H_s-2)/2} I_{i,j+t} \qquad (2)$$

where H_S = Horizontal smoothing factor

$I_{i,j}$ = original image

$N_{i,j}$ = New filtered image

$$\text{If } V_s \text{ is even } N_{i,j} = \sum_{t=-(V_s-1)/2}^{(V_s-1)/2} I_{i+t,j}, \text{ if } V_s \text{ is odd } N_{i,j} = \sum_{t=-(V_s)/2}^{(V_s-2)/2} I_{i+t,j} \qquad (3)$$

where V_S = Horizontal smoothing factor

$I_{i,j}$ = original image

$N_{i,j}$ = New filtered image

The choice of filter smoothing factors can affect the performance of the algorithm. Section 3.2.1 gives a detailed analysis of the filters in view of choosing the best parameters. Figure 7 shows the image after it has been filtered.

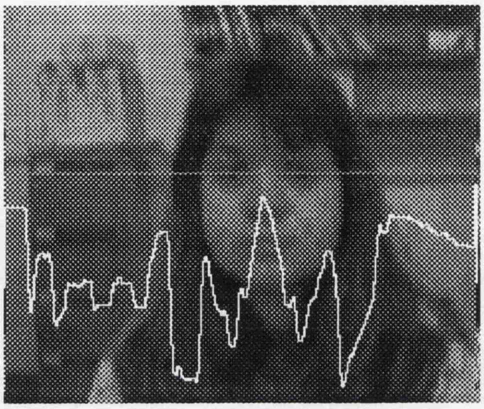


Figure 7: *Graph of intensity after image has been filtered*

The graph in figure 7 shows smoother curves than figure 5 and it is therefore easier to locate the local peaks than with the original image assuming there is no interest in high frequency peaks

The third step in the algorithm in figure 6 locates the local peaks on each horizontal raster using the algorithm in figure 8.

```
select one horizontal raster line;
set THRESHOLD variable;
if intensity is rising over the first few pixels of the raster set FLAG='rising' or else set FLAG='falling';
set POINT to the first pixel in the raster;
scan each pixel on the raster from left to right {
    if FLAG = 'rising' {
        if the intensity of current pixel > intensity of POINT then set POINT to the current pixel;
        if intensity of current pixel < intensity of (POINT) - THRESHOLD mark the POINT as a peak and set FLAG = 'falling';
    }
    if FLAG = 'falling' {
        if the intensity of current pixel < intensity of POINT then set POINT to the current pixel;
        if intensity of current pixel > intensity of (POINT) + THRESHOLD mark the POINT as a trough and set FLAG = 'rising';
    }
} until reached the end of the raster;
```

Figure 8: *Algorithm for locating horizontal peaks and troughs*

The purpose of this algorithm is to mark local peaks that protrude more than the value of the variable THRESHOLD. If the intensity ripples slightly but does not change by more than the THRESHOLD then no peak or trough will be marked. If the intensity rises, plateaus and then drops, then a peak will be marked at the beginning of the plateau. If the intensity rises, ripples and then falls, a peak will be marked at the pixel with the highest intensity in the ripple.

Varying the value of the THRESHOLD will affect the performance of the algorithm. Section 3.2.4 analyses the effect of various THRESHOLD values. Figure 9 shows the peaks located using the algorithm in figure 8.

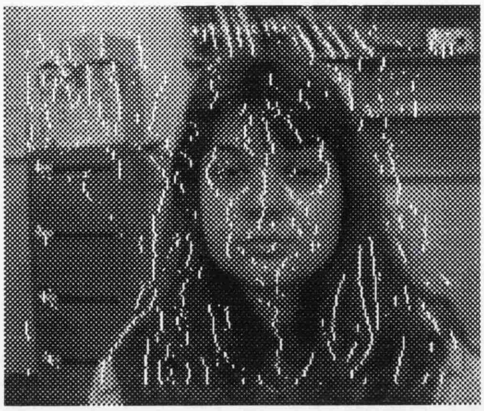


Figure 9: *Results of peak processor on image 'Sheila1'*

On figure 9 the nose can be clearly seen. The results in section 3.2.2 demonstrate the robustness of the peak processor. This is because noses on preprocessed images are usually as clear as seen on the face shown in figure 9.

The main section of the algorithm in figure 6 creates a table of all vertically connected pixels[8] (V lines). Each entry in the table gives the location of the first pixel in the line, the length and shape of the line.

There are many V lines in figure 9 many of which are short. To reduce the number of V lines all the ones of length less than MAXV are deleted. The value of MAXV affects the size of the smallest nose that can be detected. Figure 10 shows all the V lines from figure 9 with length greater than MAXV=5.

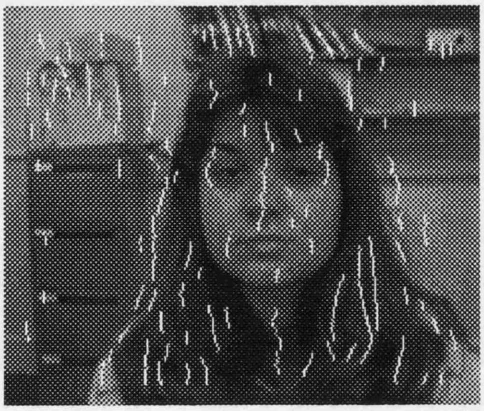


Figure 10: *V lines with length greater than 5*

[8] A pixel is vertically connected to another if it is a peak pixel that has another peak pixel either directly below it or one pixel to the left or right below it.

Analysis has shown that the number of V-lines varies from image to image but is generally in the region of 100-200.

3.1.2 Trough Preprocessor for mouth location

The trough preprocessor proposes mouths using the same algorithm as in figure 8 except that it scans each vertical line of pixels in the image and locates horizontal H lines from the troughs.

Figure 11 shows a typical output from the mouth preprocessor.

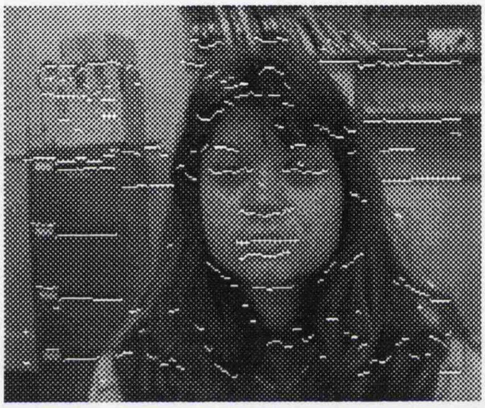


Figure 11: *Horizontally connected troughs (H lines) for proposing mouth locations*

3.2 Performance of the peak and trough preprocessors

For the peak and trough preprocessors to be of use they must fulfil one criterion:

> The peak and trough preprocessors must propose a set of points **P** that could be the desired feature. The desired feature <u>must</u> be present in the set **P**.

Therefore, the main indicator of the preprocessor performance is the proportion of images for which the above criterion is met. More technically the criterion says that there should be no ‘true rejects’.

Other indicators of performance are the percentage of the given feature that is found and the amount the proposed location deviates from the actual location of the feature. Note that the actual locations of the face features are known because they have been entered manually (see appendix 8.3).

In earlier sections of this chapter some parameters of the peak and trough algorithms were given. These settings affect the performance of the preprocessors and are discussed in the following sections relative to how well they meet criterion given above.

3.2.1 Performance of filters for the peak and trough preprocessors

The two filters used in the preprocessors are affected by smoothing factors H_S and V_S. Tests have been carried out to find out how changing the smoothing factors affect the performance of the peak and trough preprocessors. Figures 12 to 18 are graphs showing the performance of the peak preprocessor for proposing nose locations. The graphs show the performance with various smoothing factors.

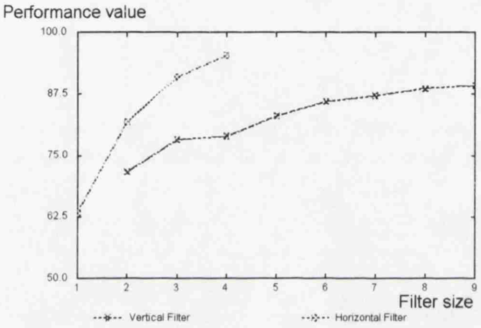


Figure 12: *Filter performance for all full sized noses*

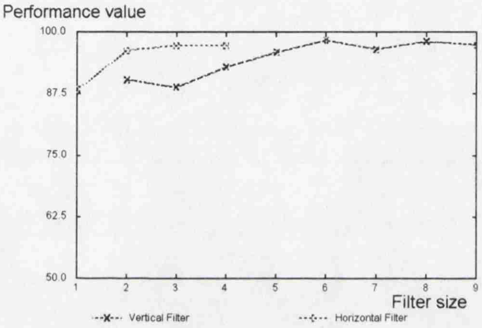


Figure 13: *Filter performance for all large noses*

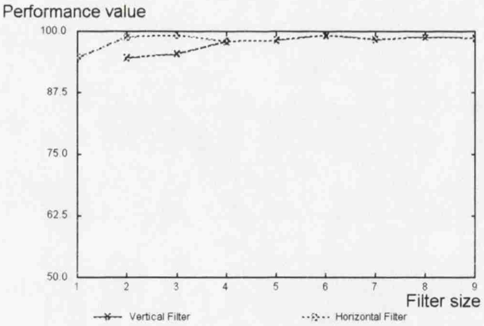


Figure 14: *Filter performance for all medium noses*

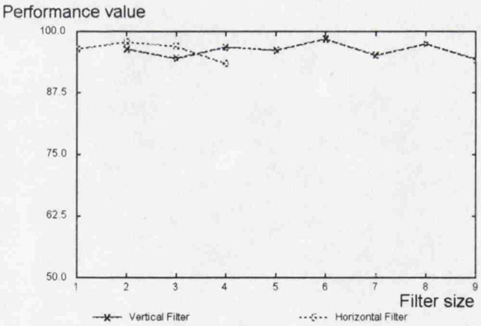


Figure 15: *Filter performance for all small noses*

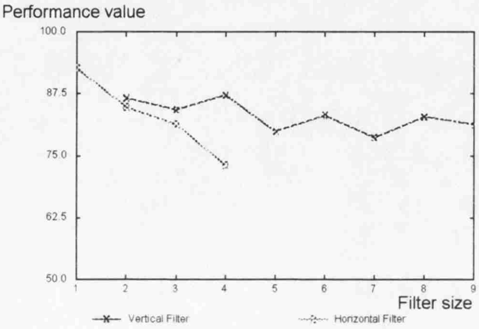


Figure 16: *Filter performance for all tiny noses*

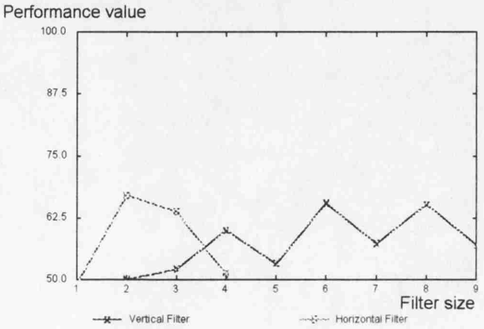


Figure 17: *Filter performance for all minute noses*

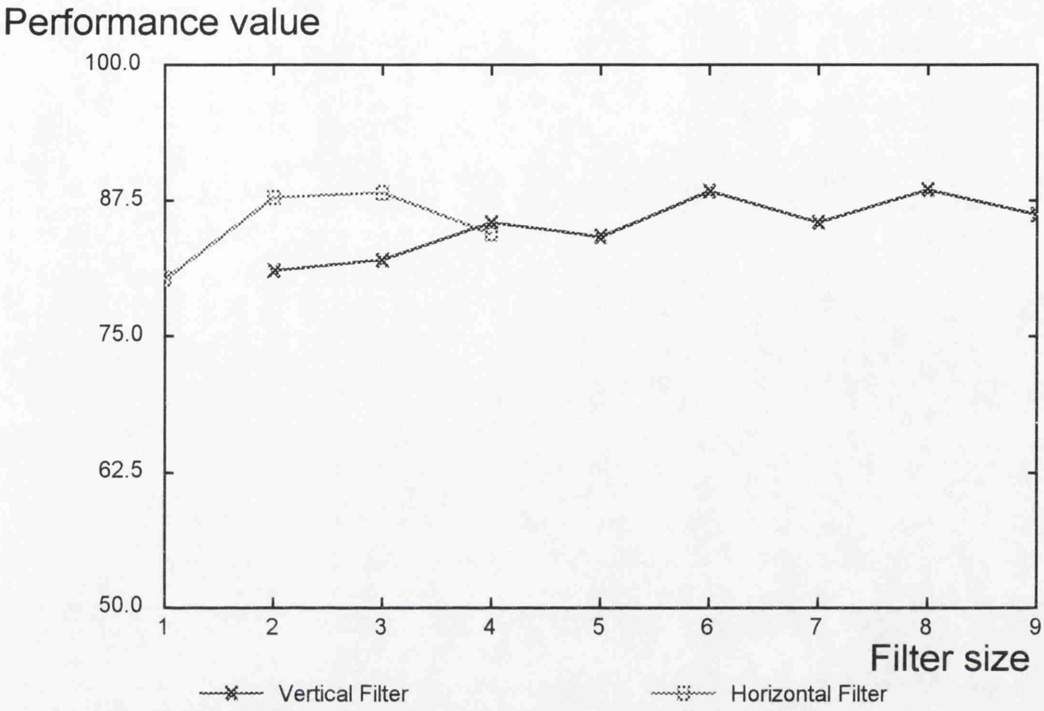


Figure 18: *Filter performance for all noses sized large to minute*

The formulae chosen for the performance value is

performance = (percentage of noses found[9]) x (percentage of V-line that covers the nose[10])

From an examination of the graphs, changes in the value of H_S have the following effect:

Figure 12 shows that for large noses with $H_S = 1$ the filter gives a poor result but as the smoothing factor increases so does the performance. Figure 14 shows that for medium faces the performance of the filter is good irrespective of the smoothing factor. Figure 16 shows that for tiny faces a value of 1 gives good results but the performance drops as the smoothing factor increases. Figure 17 shows that the filter is poor for low and high values of H_S. The filter improves for values 3 and 4 because the peak preprocessor begins to find the centre of the face rather than the centre of the nose. This is because the whole face is merged into one blob as the filter size increases, but as it increases further

[9] This figure is the percentage of images where one of the V-lines fell on the real nose.

[10] Each V-line stretches down a portion of the nose. This figure is the average percentage that each V-line covers for every image that is tested.

the face is then merged into the background. Figure 18 shows figures 12 to 17 combined into one graph. This graph shows that the best value for H_S is either 2 or 3 for the database of.faces on which the filter was tested.

Analysis of figures 12 to 18 also shows that a good compromise value for V_S is 6, which gives reasonable performance for all face sizes.

3.2.1.1 Discussion

Observation of the intensity on the human nose shows that the top surface of the nose is 'flatish'. As a result the peaks on the noses can shift randomly by few pixels on any raster line. The filters remove this movement in peaks. This observation is consistent with the fact that for *large* noses higher filter parameters are better. However a compromise needs to be reached as a large filter will remove the whole nose on an image of a *small* face. For this reason the smoothing factors given above are chosen. No value of filter size gives a reasonable result for faces smaller than 17 pixels wide. This puts obvious limits the smallest faces that can be located using this system.

The values of the smoothing factors chosen for the peak preprocessor for locating noses are not the optimum for the trough preprocessor for locating mouths. Figures 19 to 26 show the performance of the filters for proposing mouths.

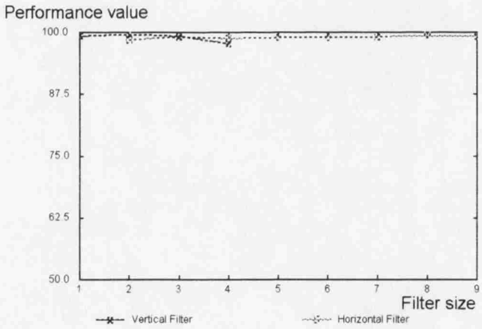


Figure 19: *Filter performance for all full size mouths*

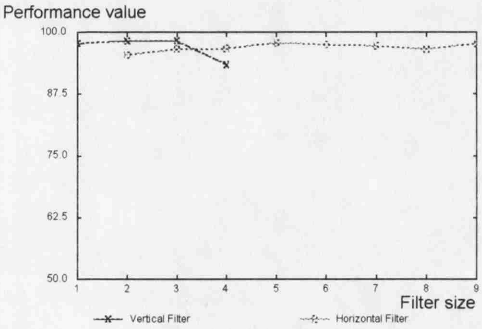


Figure 20: *Filter performance for all large mouths*

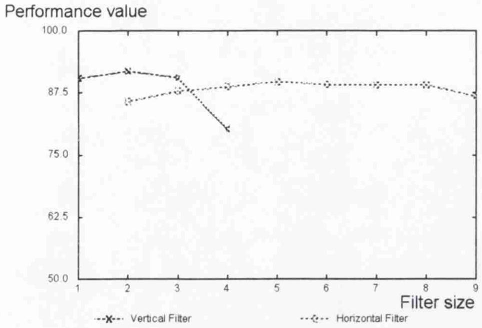


Figure 21: *Filter performance for all medium mouths*

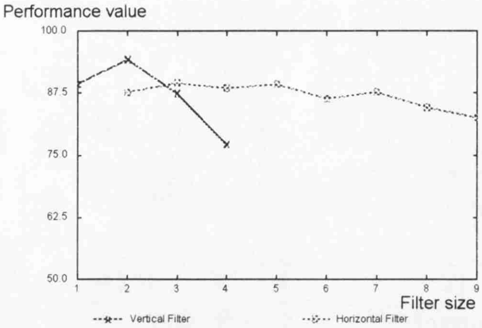


Figure 22: *Filter performance for all small mouths*

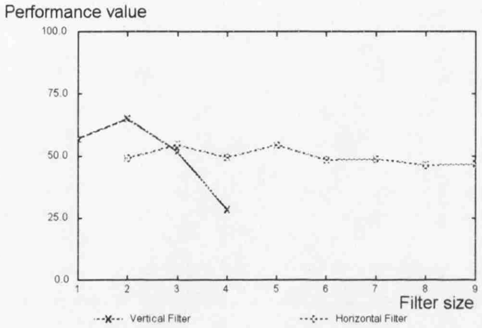


Figure 23: *Filter performance for all tiny mouths*

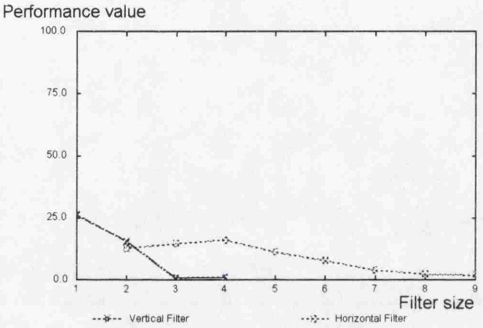


Figure 24: *Filter performance for all minute mouths*

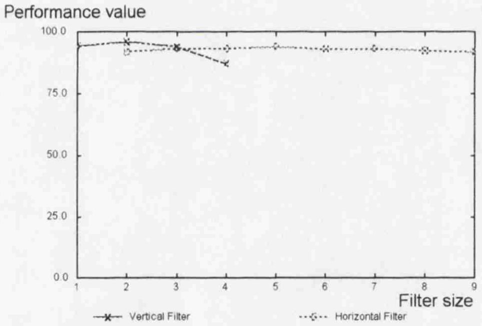


Figure 25: *Filter performance for all mouths sized large to small*

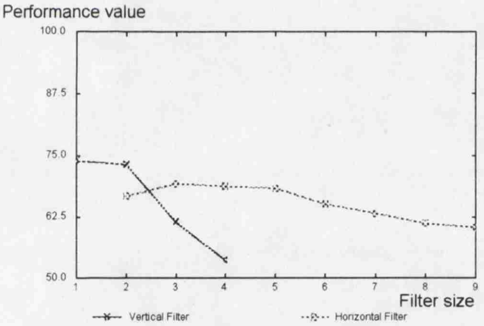


Figure 26: *Filter performance for all mouths sized large to minute*

Note that figures 23 and 24 have a different scale than the rest of the graphs as the performance of the trough algorithm for mouth location is poor for tiny and minute mouths. As a result the *tiny* and *minute* mouths have a significant effect on figure 26. For this reason figure 25 gives a better indication of the overall performance of the filter where tiny and minute mouths are excluded. Making this exclusion show that faces smaller than, approximately, 20 pixels wide will fail using trough technique for proposing mouths.

From the graphs it can be seen that a good value for V_S is 2. It is harder to establish a good value for H_S as the performance does not seem to vary so much. For this research a value of 5 was chosen.

3.2.1.2 Discussion

Examinations of the graphs 19-26 reveals that filter size used with the trough preprocessor for mouth location has less effect that the peak preprocessor for nose location. This shows that the mouth troughs are more consistent from one image column to the next than for nose peaks that move slightly from row to row. It can be concluded that the intensity drops sharply and then rises again quickly. The filter is therefore used only to remove blemishes or noise in the image. However, too large a filter will remove the trough in intensity making the mouth impossible to locate.

3.2.2 Performance of the peak preprocessor for nose location

This section gives details of tests carried out on the peak preprocessor using values of H_S=3 and V_S=6 as chosen in the previous section.

The peak preprocessor has been tested using the filter settings noted above and it was 100% successful on all the faces in the bank except for the profiled images (i.e. at least one of the noses proposed was the real nose) This success shows the robustness of the peak preprocessor technique. The peak preprocessor generates a V-line down the centre of the nose. Ideally the V-line will stretch from the bridge to the tip of the nose. The closer the V-line is to this goal the more chance the PRODIGY algorithm has of locating the nose zones correctly.

The graphs in this section are created from averages of all the images in the face bank, (except profiled images), and show the performance of the peak preprocessor. The graphs show the performance values against the size of the face in each image. This is because it is important that the preprocessors work consistently for all sizes of face.

The first graph (figure 27) shows two variables.

1. The length of the V-line divided by the length of the nose.

2. The amount of V-line that fell on the nose divided by the length of the nose.

The distinction between these two values will become apparent later.

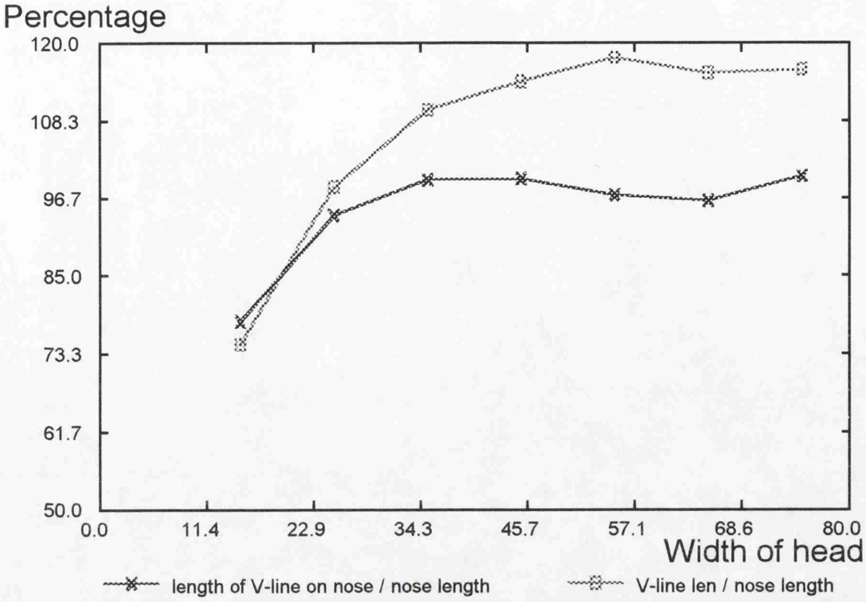


Figure 27: *V-lines as percentage of actual nose length*

The reason why the top graph in figure 27 becomes greater that 100% is that as the nose gets larger the V-line actually becomes larger than the nose. This is because the V-line starts to extend up the forehead as the face gets larger as demonstrated in figure 28. This will have an inevitable effect on any subsequent processing as its initial prediction of the nose size will be too large. However, observing and noting this problem allows adjustments to be made in the next processing stage.

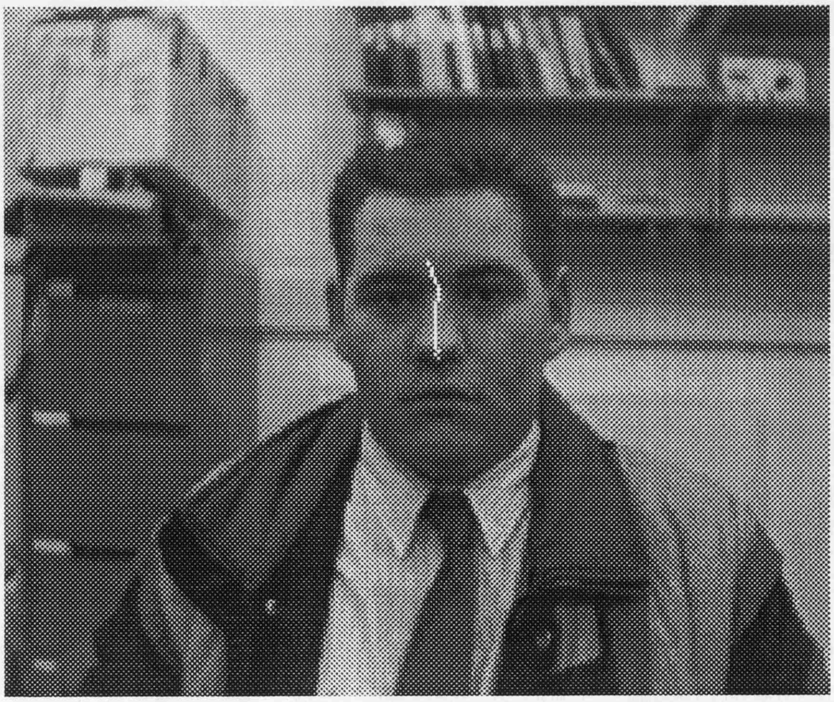


Figure 28: *Image of a large nose showing the V-line extending up between the eyes*

The second graph (figure 30) shows the number of V-lines that are found on the nose.

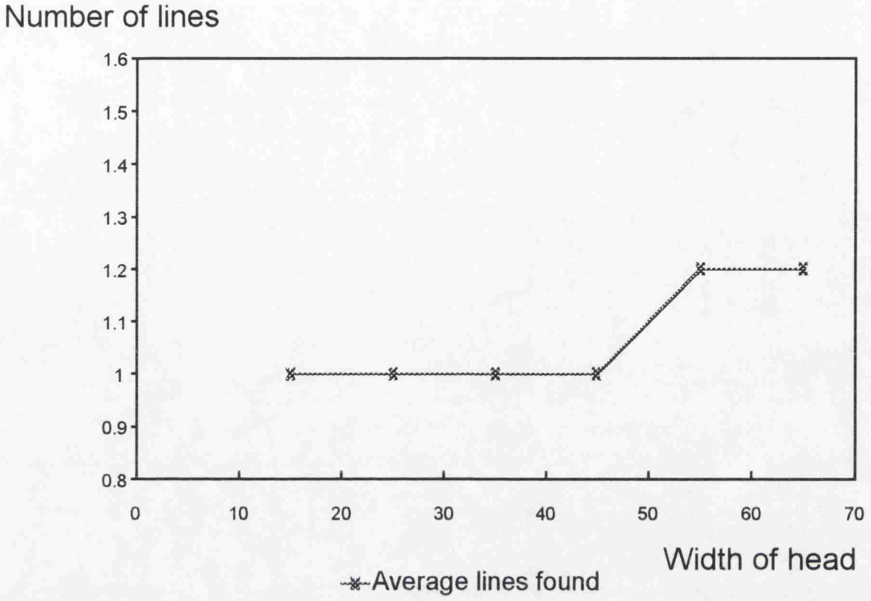


Figure 29: *Number of V-lines found on each nose*

Figure 30 shows that as the size of the nose gets larger more V-lines are found on the nose. The reason for this is that as the nose gets larger the V-line can jump more than one pixel to the left or right causing a break in the V-line. This is demonstrated in figure 30.

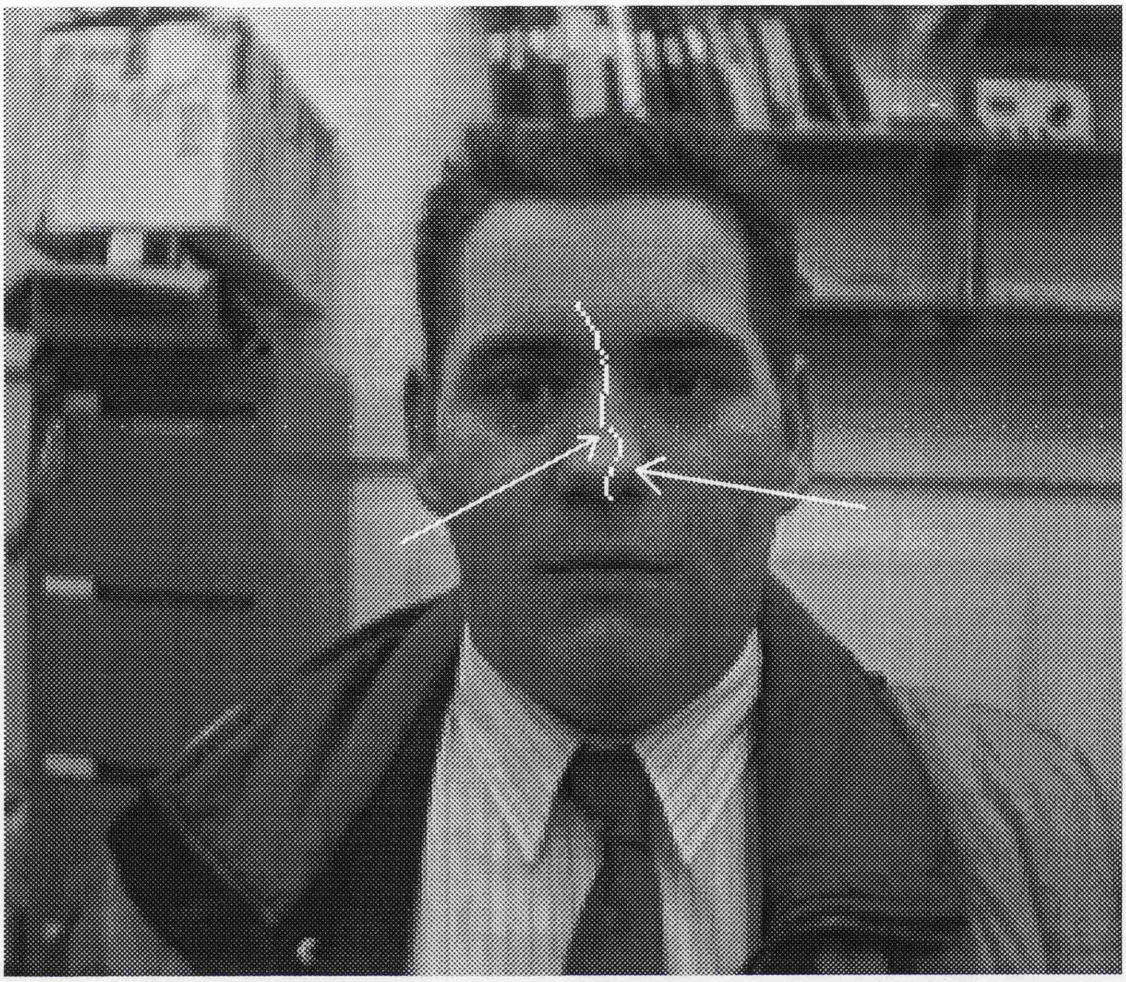


Figure 30: Image of full size face. *V-line of nose is in sections*

The algorithms designed for testing the performance of the peak processor ignore the broken sections of the V-line. This means that the V-line does not cover the whole nose. The fact that the V-line begins to be split into sections accounts for the drop in the second value in figure 27, I.e. the percentage of V-line actually on the nose drops slightly. It can be noted at this point that the peak and trough algorithms are designed to be multiscale, the problem previous described indicates that an upper limit in scale is likely to be reached. To further extend the range of these algorithms it would be necessary to perform the algorithms several times at appropriate scales.

In summary, the peak processor produces a V-line that covers 70%-95% of the nose for *minute* and *tiny* faces, and 90%-100% for *small* to *full* faces. The V-line begins to extend up the forehead and split into sections as the face gets larger. Whether these results show that the nose proposer is good or not depends on the requirement of the subsequent processing stage. By examining the errors given in the PRODIGY results (see section 4.5.1) it is concluded that a higher performance is desirable to improve the zone location part of the technique.

3.2.3 Performance of the trough preprocessor for mouth location

This section gives details of tests carried out on the trough preprocessor using values of H_S=5 and V_S=2.

Tests have shown that the trough preprocessor was successful on 100% of faces *tiny* to *full* (i.e. at least one of the mouths proposed was the real mouth), but only successful on 72% of minute faces. In a similar way to the peak preprocessor producing V-lines the trough preprocessor gives an H-line across the centre of the mouth.

Unlike the peak processor the H-lines do not protrude out of the sides of the mouth onto the cheeks. This is because the intensity rises at the edges of the mouth. Figure 31 shows the amount of the mouth marked by H-lines.

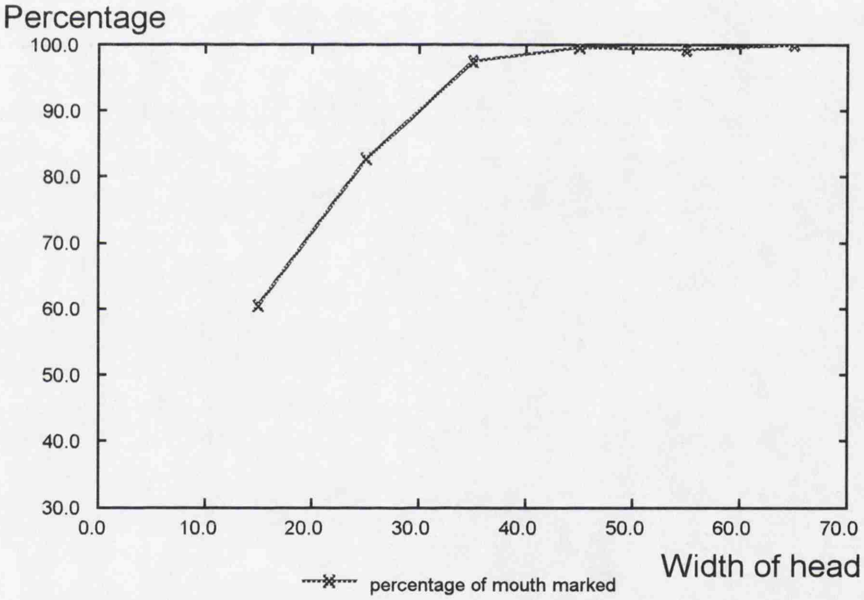


Figure 31: Percentage of mouth width containing *H-lines*

In a similar way to V-lines on the nose the H-lines can break into sections. This is more common on mouths than on noses. For this reason the graph in figure 31 shows a combination of all the sections. This observation was taken into account and catered for in the zone location algorithm (the subsequent processing stage described in the next chapter). The graph in figure 31 includes the smiling and cheesy grin images, which do not significantly affect these performance figures.

Figure 32, shows the average number of H-lines found on mouths.

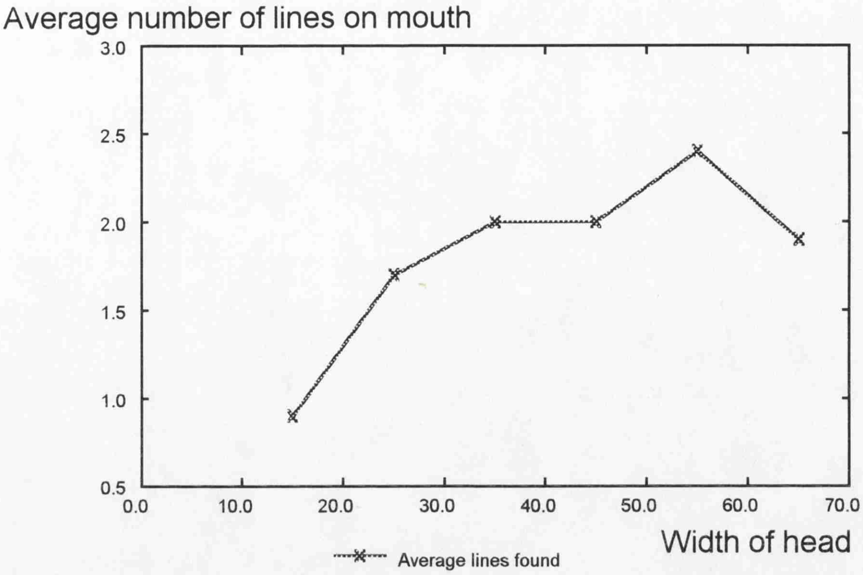


Figure 32: *Number of H-lines found on each mouth*

Figure 32 shows than on average the mouth is split into two sections. The reason for this high number of sections is demonstrated in figure 33.

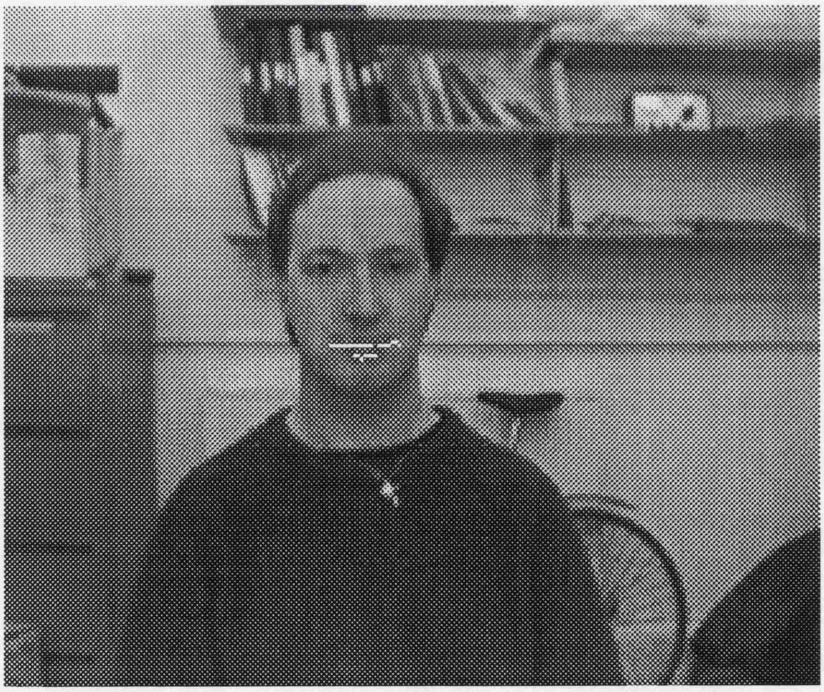


Figure 33: *Image showing mouth with three H-lines*

The upper H-lines indicates the actual mouth. The lower H-line is a distraction just below the lower lip. This 'dimple' occurs on many images and, as in this image, can cause a break in the upper H-line. The break occurs when the part of the dimple below the trough is of a much lower intensity, and so after the filter is applied the upper trough is lost.

Figure 34 shows that the trough preprocessor works with smiling faces (image type 16).

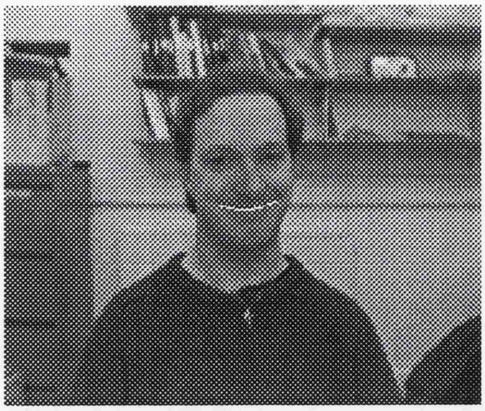


Figure 34: *Smiling face subjected to the trough preprocessor*

However, problems occur for images with cheesy grins (image type 17) as shown in figure 35.

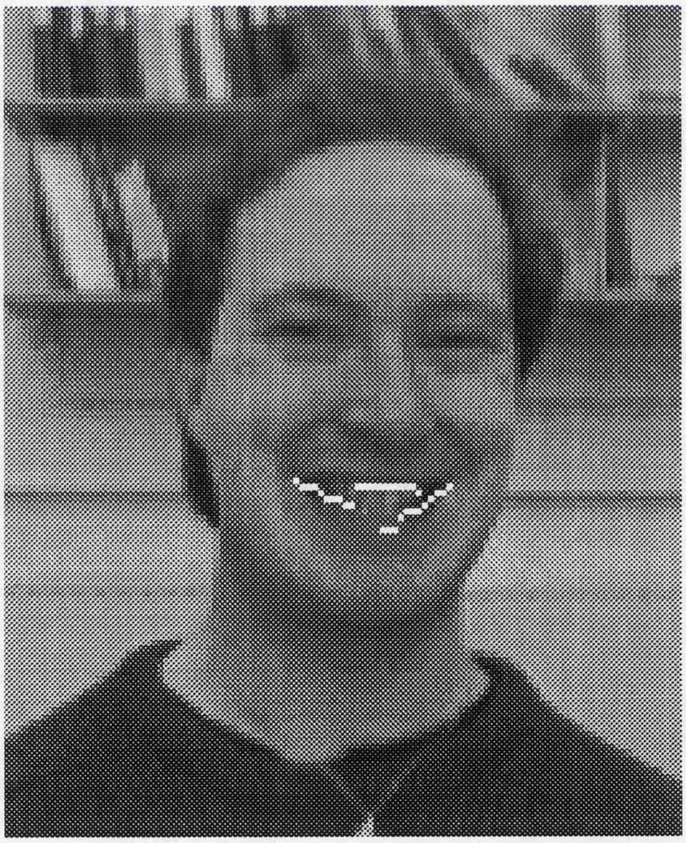


Figure 35 *Image of a cheesy grin subjected to the tough preprocessor*

Although most of the width of the mouth on the cheesy grin face is marked, because the image shows teeth, the H-line is split into numerous sections. This demonstrates some of the limits of the trough preprocessor that occur because they were based on observations of closed mouths.

In summary, the trough preprocessor produces H-lines over most of the mouth. H-lines are also produced below the lower lip which in some cases causes a break up of the real mouth's H-line. The trough preprocessor works on images with various expressions if the mouth is kept closed. For open mouths the H-lines still exist but are broken up into many sections.

3.2.4 Peak and trough threshold levels

Threshold variables were described in section 3.1.1. Threshold values between 1 and 7 have been tested on all the faces in the face bank. Tables 4 and 5 show the results of the tests which measured the effect on the performance of locating noses and mouths using the performance measures from section 3.2. The tables also show the number of false features found.

Threshold value	1	2	3	4	5	6	7
V-line on nose / nose length	96.2	95.5	94.8	94.2	93.9	92.7	91.5
V-line / nose length	108.7	107.1	105.7	103.8	102.7	100.3	98.6
number of sections found on nose	1.06	1.06	1.05	1.05	1.05	1.01	1.03
number of false noses found on an image	133	118	110	103	97	91	86

Table 4: *Effect of various threshold values on the peak preprocessor*

Threshold value	1	2	3	4	5	6	7
percentage of the mouth marked by H-line	93.6	93.4	92.9	91.7	90.4	88.2	86.4
success rate of trough preprocessor	97.4	97.4	97.4	96.8	96.8	96.1	95.5
number of sections found on mouth	1.90	1.77	1.77	1.71	1.73	1.72	1.73
number of false noses found on an image	237	194	172	156	146	136	129

Table 5: *Effect of various threshold values on the trough preprocessor*

These tables show that for both the peak and trough preprocessors lower threshold levels increase the sensitivity of the preprocessors. This improves the performance but causes the generation of many more false features and slightly more sections. These extra false features are caused by local peaks of lower intensity. This means that it will take longer to process the features and also gives a higher chance of the PRODIGY (see chapter 4) finding a false feature.

3.3 Peak and trough preprocessors with uneven lighting conditions

The performance and robustness of the peak processor for nose location drop if the subject is illuminated from the side. Most face location methods such as the one specified by Kanade (1977) state that no attempt has been made to compensate for illumination changes. Despite this, observing many images of typical scenes shows that subjects are often unevenly lit. e.g. light coming through a window will illuminate one side of the subject's face. Although this type of image was not catered for in the research face bank, some images were available. A typical image is shown in figure 36 complete with an intensity graph of a raster line along the nose.

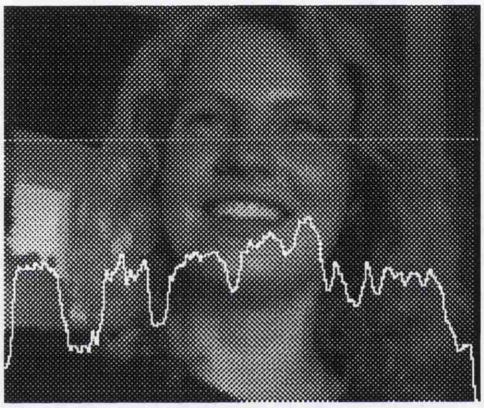


Figure 36: *Image with face lit from the right*

This image causes problems because the nose does not produce a clear peak. Examination of figure 36 reveals that when scanning the image from left to right the intensity rises fast on the left side of the nose but does not drop again on the right side. If the image was illuminated from the left then scanning the image from left to right would show flat intensity as the left side nose is approached followed by a sudden drop in intensity as the right side is reached. In neither case is there a peak.

In summary an image that is illuminated from the side has an exaggerated change in intensity on the opposite side of the nose to which it is illuminated because of a shadow. The peak preprocessor was modified to take this into account and two further versions of the peak processors were designed. One to propose noses on images illuminated from the right and one to propose noses on images illuminated from the left.

Figure 37 shows the algorithm for images illuminated from the right.

```
select one horizontal raster line;
set THRESHOLD variable;
if intensity is rising over the first few pixels of the raster then FLAG='rising' or else FLAG='falling';
set POINT to the first pixel in the raster;
scan each pixel on the raster from left to right {
    if FLAG = 'rising' {
        if the intensity of the current pixel > intensity of POINT then set POINT to the current pixel;
        else if the intensity of the current pixel = intensity of POINT then set PLATPOINT = POINT and set FLAG = 'plateau';
        else if the intensity of the current pixel < intensity of POINT then mark the POINT as a peak and set FLAG = 'falling'
    }
    if FLAG = 'falling' {
        if the intensity of current pixel < intensity of POINT then set POINT to the current pixel;
        if intensity of current pixel > intensity of (POINT) + THRESHOLD mark the POINT as a trough and set FLAG = 'rising';
    }
    if FLAG = 'plateau' {
        if intensity of current pixel > intensity of POINT then set POINT to the current pixel and set FLAG = 'kept rising';
        if intensity of current pixel < intensity of POINT then mark PLATPOINT as a peak and set FLAG = 'falling';
    }
    if FLAG = 'kept rising' {
        if intensity of current pixel > intensity of POINT then set POINT to the current pixel;
        if intensity of current pixel < intensity of (POINT) - 3 {
            if the intensity of current pixel < intensity of (PLATPOINT) - 3 then mark PLATPOINT as a peak;
            else mark POINT as a peak;
            set FLAG = 'falling';
        }
    }
} until reached the end of the raster;
```

Figure 37: *Algorithm for finding peak and troughs with an image illuminated from the right*

The algorithm in figure 37 not only marks the peaks in intensity but the beginning of plateaus as well. If a nose is illuminated from the right the intensity will rise as the nose is approached from the left. At the top of the nose the intensity will plateau instead of falling due to the light from the right. When the cheek is reached the intensity then rises again. The algorithm is designed to compensate for these phenomena. The algorithm also ignores the THRESHOLD value when finding the peaks. This allows the left side of small peaks to be marked.

3.3.1 Detecting the direction of illumination

The previous section explained how the peak preprocessor was adapted to cope with illumination from the left or right. This implies that it is already known from which direction the light is coming.

A simple method has been devised which gives the direction of illumination. This method is based on observations and will only detect substantial changes in illumination.

```
set THRESHOLD value;
set U=0;
set D=0;
for (each pixel in a block of pixels in the centre of the image) {
    if the intensity of current pixel is > intensity of the pixel to
    the right of the current pixel then set U=U+1;
    if the intensity of the current pixel is < intensity of the pixel
    to the right of the current pixel then set D=D+1;
}
if U>(D+THRESHOLD) then the illumination is from the left;
else if D<(U+THRESHOLD) then the illumination is from the right;
else the illumination is neither significantly from the left or right;
```

Figure 38: *Algorithm for detection the direction of illumination of an scene*

The algorithm in figure 38 examines an area of the image to evaluate if the intensity change from left to right is more often upwards than downwards. For an image illuminated from the right the intensity rises tend to be sharper and faster whereas the intensity falls tend to be slower and longer due to shadowing. This means that for most points on an image illuminated from the right the intensity will be falling. The THRESHOLD variable sets the sensitivity of the algorithm. As the peak preprocessor only fails on images with significantly high off centre illumination then this variable can be set relatively high.

3.4 Trough Preprocessor for eye location

Due to the iris being a circular shape it is not obvious how they can be located using the methods described for locating the nose and mouth. It is not as simple as looking for a dark circle in an image because the circle could be of any size. However, examining figures 11 and 39 shows that there are troughs in intensity in both the horizontal and vertical direction.

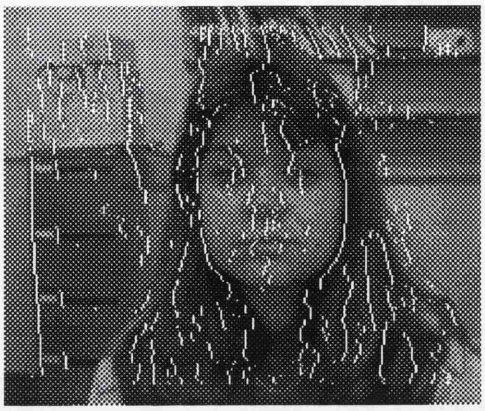


Figure 39: *Vertical troughs*

If the image is rotated as shown in figure 40 then troughs still appear on the eyes.

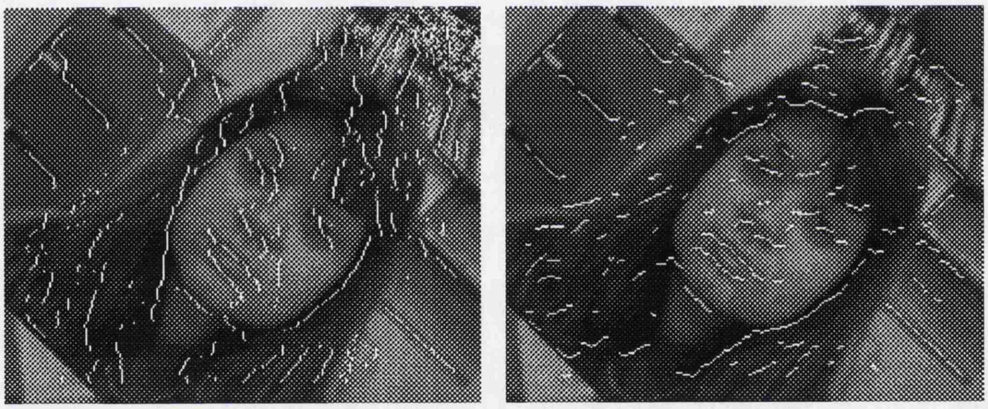


Figure 40: *Vertical and horizontal troughs on a rotated image*

The eye produces troughs irrespective of the angle the image is scanned at because the intensity of the image tends to rise in all directions from the pupil of the eye. The method used to propose eyes locates points in an image where the intensity rises or stays flat at all adjacent points. This is called the black hole method and the output of this technique is shown in figure 41.

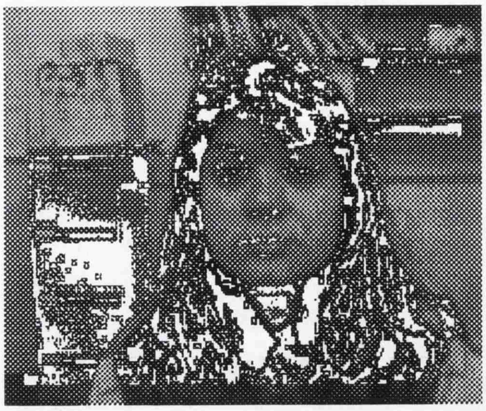


Figure 41: *Black hole eye detector*

There are many edges around the eyes. These edges are inconsistently positioned from image to image but are always present near the eye. The number of false eyes can be reduced by searching for edges around the proposed eyes. The method simply sends out edge detectors in several directions from each proposed eye point. If there is a significant edge in each direction then the proposed eye is kept, if not it is rejected.

This output of this method is shown in figure 42

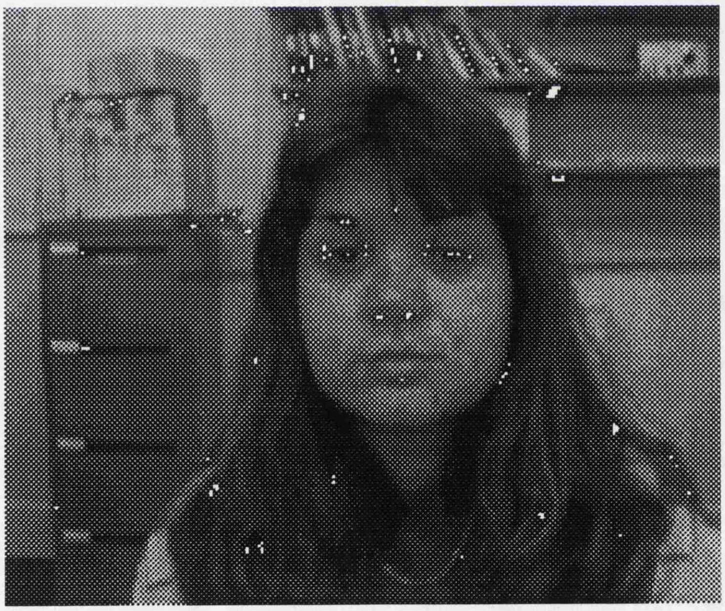


Figure 42: *Black hole eye detector followed by edge checking*

The eyes in figure 42 have many proposed points on them. To reduce the number of points on each eye and to move the point as close to the centre of the eye as possible an iterative method was designed called the pupil finder. The pupil finder algorithm shown in figure 43 attempts to find the darkest spot close to a proposed eye.

```
for (each proposed eye) {
    repeat (four times) {
        for (each point around the eye within a radius of four pixels) {
            Convolve each point with the mask $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 6 & 1 \\ 1 & 1 & 1 \end{bmatrix}$;
        }
        select the point which produced the lowest value as the new proposed eye location;
    }
}
```

Figure 43: *Algorithm for finding the centre of the eye pupil*

With each iteration, this algorithm moves the proposed eye closer towards the centre of the pupil of the eye. This assumes the centre of the eye is actually the darkest point. The output of this process is shown in figure 44.

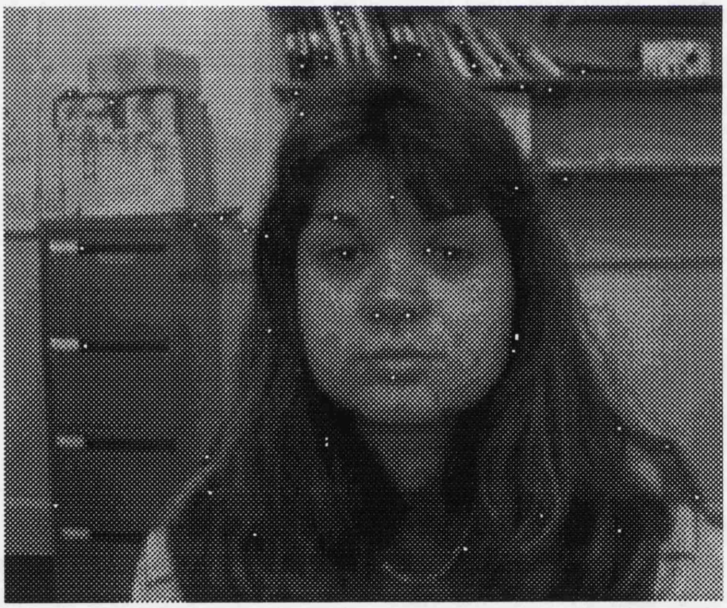


Figure 44: *Figure 42 after the pupil finding algorithm has been applied*

The final output of the preprocessor for eye location is gives a single point in the iris, but does not give any indication of the possible size of the eye. This will obviously put constraints on performance of any subsequent processing.

3.4.1 Performance of trough preprocessor for eye location

The preprocessor has been tested on the face database and it works on all the eyes with images of type *full* to *small*. On tiny and minute faces the eye is very small and can become a single pixel, which often causes errors. A medium size filter usually improves the performance on tiny faces because it merges the eyebrow with the eye making the dark spot bigger.

The eye preprocessor described in this section locates a point close to the centre of the iris. Tests show that the point deviates from the hand coded centre of the eye by 0 to 8% of the total width of the iris. This confidence level can be utilised by any subsequent processing of the eyes.

4. Intensity Gradient Techniques

The output of the peak preprocessor from chapter 3 is a list of points that are possible positions of the nose or mouth in a scene. Any subsequent processing must either eliminate points that are definitely not desired feature or for each point give a figure pertaining to the likelihood of a feature being at that point. To achieve this an intensity gradient technique called PRODIGY[11] was developed. The PRODIGY technique analyses the *Gradient Directions* around the nose or mouth.

This chapter describes what a Gradient Direction is and gives an example of some previous work in this area. Section 4.1.1 justifies the choice of the Gradient Analysis technique. The PRODIGY technique is then presented. This technique is designed from a statistical analysis of the spread of gradient directions on all the faces in the face bank and uses this information for locating facial features. The analysis is an extension to the peak and trough techniques. It analyses the surfaces of the facial features proposed by the preprocessors and checks if they fit a model built up from many sample features. The method consists of two parts; a zone location algorithm and a likelihood function. The zone location algorithm locates the edges of proposed features and partitions the feature into zones that are known to have a bias of gradient directions of particular type. The likelihood function compares a proposed feature with known versions of that feature and false versions of that feature. Although the PRODIGY could be classed as a general purpose object locator (Robertson 1992) it is used in this research as part of a face location system. It is therefore is only described as applied to nose and mouth location.

4.1 Gradient Direction Analysis

Gradient Direction analysis entails using the direction of the maximum rise in intensity to locate objects.

The use of the Sobel (Ballard et al, 1982, pp 6-81) operator is a common method for determining the direction of intensity change on an image. The Sobel detector shown in figure 45 finds the direction of maximum intensity rise at each point on the image.

[11] Proportions of the Direction of Intensity Gradient

$$Ix_{i,j} = \sum_{a=1}^{3}\sum_{b=1}^{3} X_{a,b} I_{i+a-2,j+b-2} \qquad Iy_{i,j} = \sum_{a=1}^{3}\sum_{b=1}^{3} Y_{a,b} I_{i+a-2,j+b-2}$$

where $X = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ and $Y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

Then the magnitude ($M_{i,j}$) of each point on the image is

For all i,j $M_{i,j} = \sqrt{(Ix_{i,j})^2 + (Iy_{i,j})^2}$

To calculate the direction ($D_{i,j}$) an intermediate variable ($W_{i,j}$) is introduced

For all i,j $W_{i,j} = \tan^{-1}\left[Ix_{i,j} \big/ Iy_{i,j} \right]$

Then the direction can be found for all i,j :

If $W_{i,j} < \tan^{-1}(1/3)$ then $D_{i,j} = W_{i,j}$

else If $W_{i,j} \geq \tan^{-1}(1/3)$ then $D_{i,j} = \tan^{-1}\left[\dfrac{7\tan^2 W_{i,j} + 6\tan W_{i,j} - 1}{-9\tan^2 W_{i,j} + 22\tan W_{i,j} - 1} \right]$

Figure 45: *The Sobel edge detector (Ballard et al, 1982, pp 6-81)*

It is common to quantify the angle of maximum intensity rise into one of 8 gradient directions (GDs) to simplify analysis. An intensity rise with an angle of 0-45 degrees becomes a GD of 1, and an angle of 45-90 degrees becomes a GD of 2 and so forth as shown in figure 46.

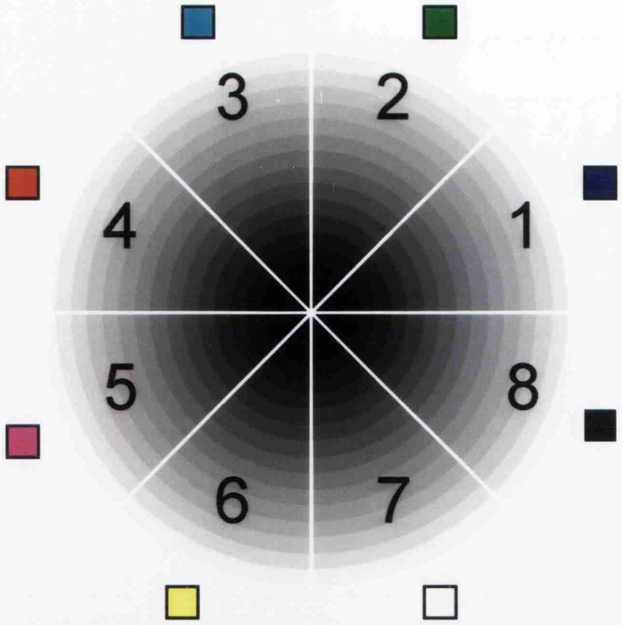


Figure 46: *The eight gradient directions (GDs)*

Using these eight GD's a program has been developed to generate a false colour image. Each pixel on the false colour image represents one pixel from the original image. One such false colour image is shown in figure 47.

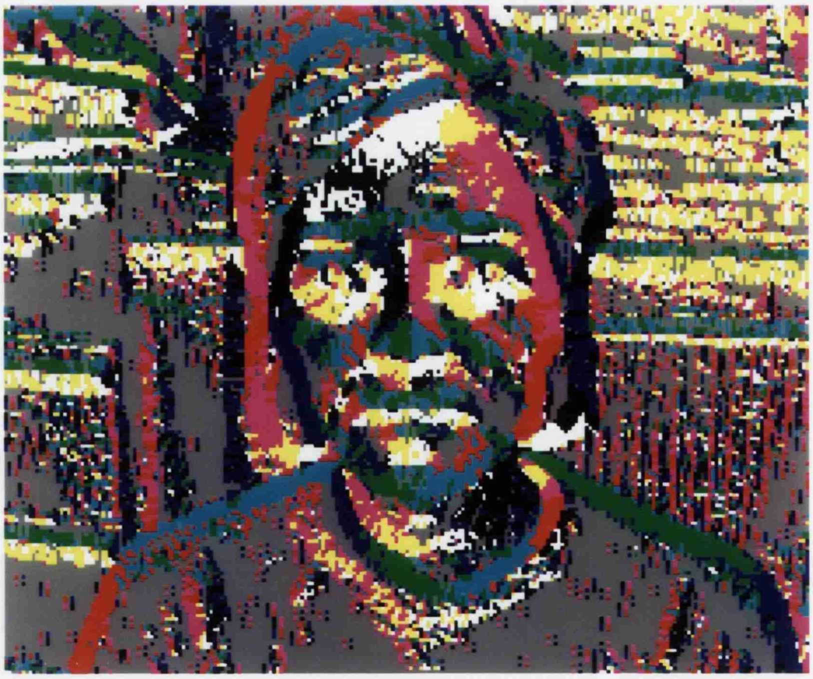


Figure 47: *A false colour image of JOY1 showing all the GDs across the scene*

A close examination of the image shows that the various facial features have definite GD characteristics, i.e. the top lip is characterised by GDs of 2 and 3.

Some parts of the image do not have a GD. Instead these are given a gradient direction of 9. GDs of 9 can be seen on the false colour image as fuzzy coloured pixels.

Craw et al (1987) implemented gradient direction analysis to locate the lips on an image of a face. The method they describe requires the faces to be presented in the centre of the image with the face taking up most of the space. Their lip locator searched for the lips in the lower middle section of the image. Figure 48 shows the algorithm that was described.

```
for (each row of pixels on the image) {
    send out a line follower from the centre of the image to the left
    {
        if a line of consecutive pixels with GDs of {2, 3} is found
        then send out a line follower from the centre of the image to
        the right {
            if a line of consecutive pixels with GDs of {2, 3} is
            found then mark an upper lip
        }
        if a line of consecutive pixels with GDs of {6, 7} is found
        then send out a line follower from the centre of the image to
        the right {
            if a line of consecutive pixels with GDs of {6, 7} is
            found then mark an lower lip
        }
    }
}
if the upper and lower lip are marked and are not separated by
excessive distance and they fit in a long thin box then record a mouth
```

Figure 48: *Craw et al's (1987) GD analysis lip finding algorithm*

4.1.1 Justification

Gradient direction analysis is chosen because

- it reduces the data that is to be analysed (256 grey levels to 8 GD's).
- clusters of similar GD's are found on parts of the facial features.
- the GDs are independent of the absolute lighting level.

Independence to absolute lighting can be proved as follows:

From figure 45, $Ix_{i,j} = \sum_{a=1}^{3}\sum_{b=1}^{3} X_{a,b} I_{i+a-2,j+b-2}$

If the overall intensity of the image is increased by the addition of a constant c the new value of $Ix_{i,j}$ which will be denoted as $I'x_{i,j}$ is

$$I'x_{i,j} = \sum_{a=1}^{3}\sum_{b=1}^{3} X_{a,b}\left(I_{i+a-2,j+b-2} + c\right)$$

$$\Rightarrow I'x_{i,j} = \sum_{a=1}^{3}\sum_{b=1}^{3} X_{a,b} I_{i+a-2,j+b-2} + \sum_{a=1}^{3}\sum_{b=1}^{3} c.X_{a,b}$$

$$\Rightarrow I'x_{i,j} = I'x_{i,j} + c\sum_{a=1}^{3}\sum_{b=1}^{3} X_{a,b}$$

As $\sum_{a=1}^{3}\sum_{b=1}^{3} X_{a,b} = 0$ then $Ix_{i,j} = I'x_{i,j}$ and similarly $Iy_{i,j} = I'y_{i,j}$.

Therefore the intensity increase 'c' has no effect on the Gradient Directions.

4.2 The PRODIGY Algorithm

The PRODIGY algorithm developed for this research combines gradient direction analysis with statistical techniques in order to locate objects. Before the PRODIGY can be used a training set of objects is statistically analysed to effectively find the average and standard deviation of the object (Statistical model). To then locate an object in a scene the PRODIGY searches for a part of the image that falls within the boundaries of the statistical model. The object characteristics statistically analysed are the gradient directions.

The general PRODIGY algorithm is given in figure 49. The various parts of it will be explained in the following sections.

```
Locate a set of points on the image that could be the object required[12];
Select a set of zones on the possible objects[13]. Each zone should have a small variation of Gradient Directions;
Calculate the percentage of each GD in each zone;
Find the likelihood that each set of zones is the desired object[14];
Select the point which has the set of zones with the maximum likelihood;
```

Figure 49: *The general PRODIGY technique*

4.3 The PRODIGY Algorithm for nose location

To apply the PRODIGY to locate noses a statistical model of the nose was first created. This section describes the model that was designed and shows several ways of implementing the PRODIGY technique.

4.3.1 Nose zones

Careful analysis of figure 47 reveals that the surface of the nose falls into several distinct sections. For the purpose of this research five zones were chosen as shown in figure 50 and table 6.

[12] In the case of this research this usually means the peak and trough algorithm.

[13] Each point located by the peak and trough algorithm is a 'possible object'.

[14] Prior statistical knowledge of the 'object required' enabled construction of a likelihood function.

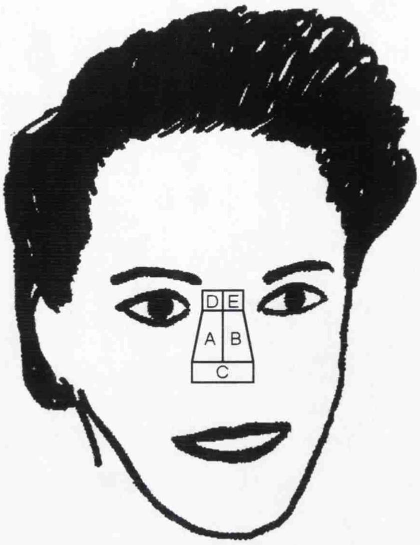


Figure 50: *Position of the five nose zones*

ZONE	AREA
A	The left side of the nose
B	The right side of the nose
C	The nostril region of the nose
D	Above the bridge of the nose to the left
E	Above the bridge of the nose to the right

Table 6: *Description of Nose Zone Areas*

From figure 47 is can be seen that each zone has a bias of points in one or two GDs. Table 7 shows the general bias observed from figure 47.

ZONE	COMMON GDs
A	Mostly 8 with some 1 and 7
B	Mostly 5 with some 4 and 6
C	A mixture of 2 and 3 with some 1 and 4
D	Mostly 2 with some 1 and 3
E	Mostly 3 with some 2 and 4

Table 7: *Common Gradient Directions in each zone*

The PRODIGY technique takes account of this information when locating objects in a scene although the data in table 7 is not sufficiently accurate to enable the algorithm to locate noses. To obtain a more accurate description of the zones it was necessary to analyse all the noses in the whole image bank to determine the average and standard deviation of each GD in each zone. Before doing this it was necessary to define the boundaries of each zone more accurately so that the computer can predict zones consistently for each image.

The zone predictor was based on the output of the peak preprocessor for proposing nose locations. From section 3.1.1 it is known that the output of the peak preprocessor is a line down the centre of the nose[15]. The zone predictor uses the knowledge of this line, which effectively gives the separation between the A&B zones and follows on using the algorithm in figure 51.

[15] The peak preprocessor produces many V-lines on an image. Each V-line would be down the centre of the nose if one existed at that location. For the purpose of the statistical analysis the V-line on the real nose was manually located.

```
proc to locate the boundary between the AB zone and DE zone {
    Mark an area around the top of the V-line[16];
    Select the bottom horizontal line of pixels in the area;
    while there are more points with GDs of {5, 6, 7, 8) than of GDs
    of {2, 3} on the line {
        select the next horizontal line above the last;
    }
    mark the boundary at the line;
}
proc to locate the boundary between the AB zone and C zone {
    Mark and area around the bottom of the V-line[17];
    Select the top horizontal line of pixels in the area;
    while the number points on the line with GDs of {5, 6, 7, 8) +
    half the number of points on the line with GDs of {1, 4} is
    greater than the number of points on the line with GDs of {2, 3} {
        select the next horizontal line below the last;
    }
    mark the boundary at the line;
}
proc to locate the right edge of the B zone {
    Assume that the B side of the nose is 12 degrees from the
    vertical;
    Select a line of pixels 12 degrees from the V-line and the height
    of the nose;
    while the number of points on the line with GDs {1, 8} is greater
    than the number of points on the line with GDs {2, 3, 4, 5, 6, 7}
    {
        Select a line one pixel to the right of the last
    }
    mark the edge at the current line;
}
proc to locate the left edge of the A zone {
    Assume that the A side of the nose is -12 degrees from the
    vertical;
    Select a line of pixels -12 degrees from the V-line and the height
    of the nose;
    while the number of points on the line with GDs {4, 5} is greater
    than the number of points on the line with GDs {1, 2, 3, 6, 7, 8}
    {
```

[16] If the top of the Vline is at location i,j with length l the area marked is the rectangle enclosed by (j-l/2,i-l/2), (j+l/2, i-l/2), (j+l/2, i+l/2), (j-l/2, i+l/2)

[17] The area marked is the rectangle enclosed by (j-l/2, i+l/2), (j+l/2, i+l/2), (j+l/2, i+3l/2), (j-l/2, i+3l/2)

```
            Select a line one pixel to the left of the last
    }
    mark the edge at the current line;
}
```

Figure 51: *Algorithm for locating the nose zones from a given V-line*

The algorithm in figure 51 is based on the following observation. From figure 47 it can be seen that at the boundary of each zone there is a transition from certain types of GDs to others. For example on the boundary between zones A and B there is a transition from GDs {1, 8} to GDs of {4,5}. The transition is not abrupt but there is a point where there is more of one GD than the other. The boundary is between the zones is marked at this point.

4.3.2 Statistical Analysis of the real nose zones

Each zone was statistically analysed separately over all the real noses in the face bank and the results in table 8 show the average percentage of each GD in each zone.

ZONE	GRADIENT DIRECTION							
	1	2	3	4	5	6	7	8
A	25.6	4.7	2.0	4.5	3.3	4.1	12.2	43.6
B	4.7	1.8	2.7	20.8	48.9	11.3	5.2	4.6
C	12.8	29.8	26.2	10.6	6.1	2.8	3.3	8.4
D	12.0	47.7	23.3	3.5	4.0	2.9	3.1	3.5
E	3.4	15.4	44.3	24.4	6.0	3.5	1.1	1.9

Table 8: *Average percentage of GDs in each zone for real noses*

The standard deviations of the values in table 8 were also calculated and are shown in table 9.

ZONE	GRADIENT DIRECTION							
	1	2	3	4	5	6	7	8
A	10.3	2.7	2.0	3.5	3.1	3.3	7.1	9.4
B	4.2	1.7	2.2	6.7	9.0	6.2	4.1	3.2
C	6.1	8.5	7.1	5.4	4.7	3.3	4.8	5.0
D	15.2	25.4	20.1	6.8	10.6	9.7	7.7	7.1
E	5.3	17.9	26.3	24.1	15.9	12.9	3.5	8.1

Table 9: *Standard deviation of GDs in each zone, for real noses*

Examining table 8 confirms the observations in table 7. This with the fact that the figures in table 9 are low (as compared to table 12) indicates that it is possible to use these values for locating noses. This is because they show a consistency in GDs from nose to nose.

The covariance matrix for each zone was also calculated and table 10 shows the covariance matrix for zone A.

GD	1	2	3	4	5	6	7	8
1	106.7	-2.2	-0.1	15.3	15.9	-7.0	-51.0	-77.6
2	-2.2	7.0	0.2	-1.3	-0.3	3.0	-2.3	-4.1
3	-0.1	0.2	3.8	-0.8	0.01	0.1	-1.0	-2.2
4	15.3	-1.3	-0.8	12.0	2.1	-0.9	-14.3	-12.1
5	15.9	-0.3	0.01	2.1	9.3	0.1	-12.1	-15.1
6	-7.0	3.0	0.1	-0.9	0.1	10.8	0.2	-6.4
7	-51.0	-2.3	-1.0	-14.3	-12.1	0.2	51.1	29.5
8	-77.6	-4.1	-2.2	-12.1	-15.1	-6.4	29.5	88.0

Table 10: *Covariance matrix for zone A of real noses*

Table 10 shows the interrelationship between the various gradient directions of zone A. The table shows that if GD 7 increases then it is likely that GD 8 will also increases. It shows that if GD 8 increases it is likely that GD 1 will decrease. It also shows that GDs like 2,3, and 6 are not related as much as GD's 1, 7 and 8.

4.3.3 Statistical Analysis of the false nose zones

So far the analysis has been constrained to 'real' noses. The statistical analysis of these noses was performed on the faces from the bank. To perform the analysis the 'real' nose was manually selected from the list of V-lines given by the peak preprocessor. Recall that the algorithm for locating the zones depends on the GDs, which is exactly the same as is being statistically analysed. Therefore V-lines that do not fall on 'real' noses (i.e. they fall on 'false' noses) will also be statistically biased. If the bias for 'false' noses is the same as for 'real' noses then it would be impossible to distinguish 'real' noses from 'false' noses.

To evaluate the bias for 'false' noses all the V-lines that are not 'real' noses were statistically analysed in the same way as for 'real' noses. The results are shown in tables 11 and 12.

ZONE	GRADIENT DIRECTION							
	1	2	3	4	5	6	7	8
A	27.8	5.5	3.8	7.7	10.5	7.2	15.3	22.2
B	13.1	4.7	5.7	16.8	25.6	13.6	9.2	11.3
C	17.8	11.6	13.5	14.6	14.5	8.0	8.4	11.6
D	18.8	14.4	12.5	8.1	11.0	9.4	11.1	14.7
E	14.6	12.4	17.2	11.7	13.4	9.6	9.8	11.3

Table 11: *Average percentage of GDs in each zone for false noses*

ZONE	GRADIENT DIRECTION							
	1	2	3	4	5	6	7	8
A	22.1	8.5	8.9	13.4	13.2	11.3	18.4	17.9
B	17.0	8.3	9.7	17.6	21.1	18.2	14.4	13.4
C	18.9	13.6	16.2	17.2	17.3	13.4	14.6	13.8
D	27.4	24.0	21.9	18.4	22.8	20.8	23.0	24.8
E	24.2	22.2	28.3	22.8	24.9	21.1	22.0	21.7

Table 12: *Standard deviation of GDs in each zone for false noses*

The data in tables 11 and 12 show that although the bias for false noses is similar to real noses it is not as great. The standard deviations in table 12 are much higher than in table 9. This means that many of the false noses must fall well outside the range of real noses and are therefore distinguishable from real noses.

4.3.4 Likelihood Ratios

The output of the peak preprocessor is a few hundred V-lines. For faces in the bank only one of these V-lines is the real nose and the rest are false. To find out which are true or false each V-line is tested with the following hypotheses.

H_0 : The V-line is a real nose

H_1 : The V-line is a false nose

Likelihood ratios can give a figure that assists in the selection of the correct hypothesis. Edwards (1972) defines the likelihood ratio and when applied to the hypothesis above is given in equation 4.

$$L(x) = \frac{R(x)}{F(x)} \qquad (4)$$

where *R(x)* and *F(x)* are the Probability Distribution Functions of real and false noses. *x* is the value of a proposed nose.

4.3.5 Statistical Assumptions

It is assumed that the probability distributions of the GDs are normal distributions. As the value for each GD is between 0%-100% and the definition of the normal distribution is unbounded then the distribution cannot be normal. However, the distribution can be approximated as 'normal' and incur only small errors if the average is not close to the bounds of the function (and the standard deviation is appropriate). Some of the GDs have very low averages (4-6). These are the GDs not expected to be found in a zone. As the standard deviations are around (8-10) then approximating the function to a normal distribution will clearly give significant errors. The errors caused by approximating the PDFs as normal have not yet been analysed but would be appropriate to consider in future research.

Section 4.3.7 introduces the use of a covariance matrix method, which is aimed at reducing the dependence between each GD. The PDFs of the vectors used to create these covariance matrices are approximated as multi-dimensional normal distributions. This approximation can create even larger errors that outweigh the positive effects of using covariance matrices. These errors can be reduced by removing the elements that cause the distortion from the normal distributions as does the compromise method described in section 4.3.8.

4.3.6 Likelihood ratio from individual zones and GD's : standard method

The likelihood ratio in equation 4 cannot be found directly because a PDF does not exist. However, the PDF's for each individual GD of each zone the likelihood ratio for a GD of a zone can be found as shown in equation 5.

$$L(x_{zd}) = \frac{R(x_{zd})}{F(x_{zd})} \qquad (5)$$

where $R(x_{zd})$ and $F(x_{zd})$ are the PDFs of a particular GD, *d*, of zone *z* for real and false noses respectively.

Assuming the PDFs are normal then from the equation of the normal standard deviation (Milton and Arnold, 1986)

$$R(x_{zd}) = \frac{1}{\sqrt{2\pi}\sigma^R_{zd}} \exp\left[-\frac{1}{2}\left(\frac{x_{zd} - \mu^R_{zd}}{\sigma^R_{zd}} \right)^2 \right] \tag{6}$$

where σ^R_{zd} is the standard deviation of real noses for a particular zone and gradient direction. μ^R_{zd} is the average value of real noses for a particular zone and GD as found in tables 8 and 9.

Similarly

$$F(x_{zd}) = \frac{1}{\sqrt{2\pi}\sigma^F_{zd}} \exp\left[-\frac{1}{2}\left(\frac{x_{zd} - \mu^F_{zd}}{\sigma^F_{zd}} \right)^2 \right] \tag{7}$$

Therefore

$$L(x_{zd}) = \frac{R(x_{zd})}{F(x_{zd})} = \frac{\sigma^F_{zd}}{\sigma^R_{zd}} \exp\left[\frac{1}{2}\left(\left[\frac{x_{zd} - \mu^F_{zd}}{\sigma^F_{zd}} \right]^2 - \left[\frac{x_{zd} - \mu^R_{zd}}{\sigma^R_{zd}} \right]^2 \right) \right] \tag{8}$$

Equation 8 gives the likelihood for a specific GD of a specific zone. The PRODIGY requires that the likelihood of the whole nose is found (eqn 4). If each of the 'zone, GD' likelihoods were independent then the nose likelihood ratio would be a product of all the 'zone, GD' likelihoods as shown in equation 9.

$$L(x) = \prod_{z=1}^{5} \prod_{d=1}^{8} L(x_{zd}) \tag{9}$$

To simplify calculation the log of the likelihood was derived as shown in equation 10.

$$\log(L(x)) = \sum_{z=1}^{5} \sum_{d=1}^{8} \Psi_{zd} + \frac{1}{2}\left(\left[\frac{x_{zd} - \mu^F_{zd}}{\sigma^F_{zd}} \right]^2 - \left[\frac{x_{zd} - \mu^R_{zd}}{\sigma^R_{zd}} \right]^2 \right) \tag{10}$$

where $\Psi_{zd} = \log\left(\frac{\sigma^F_{zd}}{\sigma^R_{zd}} \right)$

Taking the log of the likelihood function does not change the relative[18] values of the likelihoods of various noses. Therefore, the log likelihood function is also a likelihood function. Indeed as Ψ_{zd} is a constant and the (1/2) is a constant multiplier they can also be removed without affecting the relative values of the function and so the likelihood function is simplified to equation 11.

$$L(x) = \sum_{z=1}^{5}\sum_{d=1}^{8}\left[\frac{x_{zd}-\mu_{zd}^{F}}{\sigma_{zd}^{F}}\right]^{2} - \left[\frac{x_{zd}-\mu_{zd}^{R}}{\sigma_{zd}^{R}}\right]^{2} \qquad (11)$$

4.3.7 Likelihood Ratio using zone vectors : vector method

Section 4.3.5 states that both the following assumptions cause errors: (1) that the PDF's are normal, (2) that the likelihoods of each zone are independent.

Assumption (2) can be overcome by calculating the likelihood of a whole zone (equation 12) by using a vector and covariance matrix.

$$L(x_z) = \frac{R(x_z)}{F(x_z)} \qquad (12)$$

where x_z is a the vector for zone z (all 8 GD for a zone in a vector).

$R(x_z)$ is the PDF of zone z. Again the assumption is made that the PDF is a normal distribution. This multidimensional normal distribution is given in equation 13 (Krzanowski, 1988).

$$R(x_z) = \frac{1}{\sqrt{(2\pi)^{8}\left|\Sigma_{z}^{R}\right|}}\, e^{-\frac{1}{2}\left[\left(x_z-\mu_z^R\right)^{T}\left(\Sigma_z^R\right)^{-1}\left(x_z-\mu_z^R\right)\right]} \qquad (13)$$

where μ_z^R is the average vector for a given zone z for real noses and Σ_z^R is the covariance matrix for zone z.

[18] Note that a likelihood function has a higher result if the input is more 'likely'. Therefore for two values a and b where a is more likely than b then $L(a) > L(b)$ is true. This inequality still holds for $(\log(L(a))-3)*2 > (\log(L(b))-3)*2$.

The normal distribution for false noses is similar to equation 13. Therefore, the likelihood of a zone derived from equations 12&13 is as given in equation 14.

$$L(x_z) = \frac{\sqrt{\left|\Sigma_z^F\right|}}{\sqrt{\left|\Sigma_z^R\right|}} e^{\frac{1}{2}\left[(x_z-\mu_z^F)^T(\Sigma_z^F)^{-1}(x_z-\mu_z^F)-(x_z-\mu_z^R)^T(\Sigma_z^R)^{-1}(x_z-\mu_z^R)\right]} \tag{14}$$

The log likelihood of the whole nose, assuming independence between zones, is the sum of the each zone likelihood as given in equation 15.

$$\begin{gathered} L(x) = \prod_{z=1}^{5} L(x_z) \Rightarrow \log(L(x)) = \\ \sum_{z=1}^{5} \Psi_z + \frac{1}{2}\left[(x_z-\mu_z^F)^T(\Sigma_z^F)^{-1}(x_z-\mu_z^F)-(x_z-\mu_z^R)^T(\Sigma_z^R)^{-1}(x_z-\mu_z^R)\right] \end{gathered} \tag{15}$$

where $\Psi_z = \log\left(\frac{\sqrt{\left|\Sigma_z^F\right|}}{\sqrt{\left|\Sigma_z^R\right|}}\right)$

Equation 15 can be simplified in a similar way to equation 11 to give equation 16.

$$L(x) = \sum_{z=1}^{5} (x_z-\mu_z^F)^T(\Sigma_z^F)^{-1}(x_z-\mu_z^F)-(x_z-\mu_z^R)^T(\Sigma_z^R)^{-1}(x_z-\mu_z^R) \tag{16}$$

4.3.8 Likelihood Ratios using reduced zone vectors : compromise method

From an examination of the covariance matrix used in the vector method although clearly some GD's are dependent (e.g. 7 & 8) others have virtually no dependence (3 & 5). This causes the covariance matrix to be near singular and near singular matrices are difficult to invert using the available numerical techniques.

This problem, together with the assumption of a normal distribution, causes a significant amount of error. This is apparent from the results in section 4.5. To reduce the effect of these errors a compromise method combining the standard method and the vector method was designed. The eight GDs are split into two; 4 of them combined into a vector (the compromise vector C_z), and the other 4 used separately (y_{zn} where z is the zone and n is the GD of the new subset of the original eight GDs). The likelihood of the compromise vector is found using the vector method and the likelihood of the subset is found by the standard method.

Figure 52 shows the relationship between the original zone and the compromise variables.

old zone = $\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}$

compromise vector $C = \begin{bmatrix} x_a \\ x_b \\ x_c \\ x_d \end{bmatrix}$

and new zone subset = $\{y_1, y_2, y_3, y_4\}$ where $y_1 = x_e, y_2 = x_f, y_1 = x_g, y_2 = x_h$

a≠b≠c≠d≠e≠f≠g≠h

Figure 52: *Relationship between compromise 'vector, zone value' and original zone*

The values of {a,b,c,d,e,f,g,h} determine which GDs are included in the compromise vector and the new zone subset. The GDs that are selected for the compromise vector in each zone are the ones that have the highest correlation. For zone A this turns out to be {a=1,b=5,c=7,d=8,e=2,f=3,g=4,h=6}.

From table 8 and 10 the compromise vector for zone A and its covariance matrix are

$$C = \begin{bmatrix} 25.6 \\ 3.3 \\ 12.2 \\ 43.6 \end{bmatrix}, \text{ with covariance } \Sigma = \begin{bmatrix} 106.6 & 15.9 & -51.0 & -77.6 \\ 15.9 & 9.3 & -12.1 & -15.1 \\ -51.0 & -12.1 & 51.1 & 29.5 \\ -77.6 & -15.1 & 29.5 & 88.3 \end{bmatrix}$$

The likelihood function for the whole zone is a combination of equation 11 and 16 giving equation 17

$$L(Y) = \sum_{z=1}^{5} \begin{bmatrix} (C_z - \mu_z^F)^T (\Sigma_z^F)^{-1} (Y_z - \mu_z^F) - (Y_z - \mu_z^R)^T (\Sigma_z^R)^{-1} (Y_z - \mu_z^R) \\ + \left(\sum_{n=1}^{4} \left[\frac{y_{zn} - \mu_{zn}^F}{\sigma_{zn}^F} \right]^2 - \left[\frac{y_{zn} - \mu_{zn}^R}{\sigma_{zn}^R} \right]^2 \right) \end{bmatrix} \qquad (17)$$

The results of tests carried out on the compromise vector are given in section 4.5.

4.3.9 Summary of Likelihood Ratios

The previous three sections describe different ways of finding a likelihood function of a possible nose all which make different assumptions. These methods are summarised in equations 18, 19 and 20 below. These equations are more general solutions not specifically for noses and therefore include extra variables denoting the number of zones.

- The standard method

$$L(x) = \sum_{z=1}^{p} \sum_{d=1}^{8} \left[\frac{x_{zd} - \mu_{zd}^{F}}{\sigma_{zd}^{F}} \right]^2 - \left[\frac{x_{zd} - \mu_{zd}^{R}}{\sigma_{zd}^{R}} \right]^2 \qquad (18)$$

where

x = a proposed object

p = number of zones on object

x_{zd} = value for a particular zone and direction of the proposed object

μ_{zd}^{R} = average value for a particular zone and direction of known object specimens

σ_{zx}^{R} = standard deviation of μ_{zd}^{R}

μ_{zd}^{F} = average value for a particular zone and direction of false object specimens

σ_{zd}^{F} = standard deviation of μ_{zd}^{F}

- The vector method

$$L(x) = \sum_{z=1}^{p} \left(x_z - \mu_z^F\right)^T \left(\Sigma_z^F\right)^{-1} \left(x_z - \mu_z^F\right) - \left(x_z - \mu_z^R\right)^T \left(\Sigma_z^R\right)^{-1} \left(x_z - \mu_z^R\right) \qquad (19)$$

where
x = a proposed object
p = number of zones on object
x_z = vector of a particular zone of the proposed object
μ_z^R = average vector for a particular zone of known object specimens
Σ_z^R = Covariance of μ_z^R
μ_z^F = average vector for a particular zone of false object specimens
Σ_z^F = Covariance of μ_z^F

- The compromise method

$$L(Y) = \sum_{z=1}^{p} \left[\begin{array}{l} \left(C_z - \mu_z^F\right)^T \left(\Sigma_z^F\right)^{-1} \left(Y_z - \mu_z^F\right) - \left(Y_z - \mu_z^R\right)^T \left(\Sigma_z^R\right)^{-1} \left(Y_z - \mu_z^R\right) \\ + \left(\sum_{n=1}^{4} \left[\frac{y_{zn} - \mu_{zn}^F}{\sigma_{zn}^F} \right]^2 - \left[\frac{y_{zn} - \mu_{zn}^R}{\sigma_{zn}^R} \right]^2 \right) \end{array} \right] \tag{20}$$

where
Y = a proposed object
p = number of zones on object
C_z = compromise vector of a particular zone of the proposed object
μ_z^R = average compromise vector for a particular zone of known object specimens
Σ_z^R = Covariance of μ_z^R
μ_z^F = average compromise vector for a particular zone of false object specimens
Σ_z^F = Covariance of μ_z^F
y_{zn} = value of element n of the zone subset z of the proposed object
μ_{zn}^R = average of element n of the zone subset z of known object specimens
σ_{zn}^R = standard deviation of μ_{zn}^R
μ_{zn}^F = average of element n of the zone subset z of false object specimens
σ_{zn}^F = standard deviation of μ_{zn}^R

These three methods have been tested and are presented in section 4.5, which gives the relative merits of each technique.

4.4 The PRODIGY Algorithm for mouth location

The general technique for locating the mouth using the PRODIGY technique is similar to the one for locating the nose. The main difference is in the way in which the mouth zones are located.

4.4.1 Mouth zones

Figure 47 reveals that the mouth, like the nose, falls into several sections. For the mouth four zones were chosen as shown in figure 53 and described in table 13.

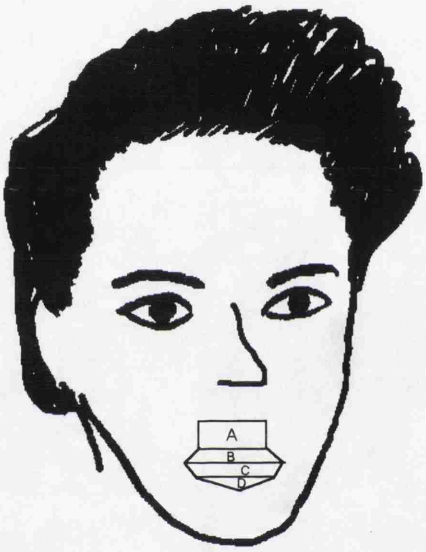


Figure 53: *Position of the four mouth zones*

ZONE	AREA
A	Above the mouth
B	The upper lip
C	The lower lip
D	Below the mouth

Table 13: *Description of the mouth zone areas*

Observations from figure 47 shows that the bias of GDs displayed in table 14.

A	Mostly 6 and 7
B	Mostly 2 and 3
C	Mostly 6 and 7
D	Mostly 2 and 3 with some 1 and 4

Table 14: *Common Gradient Directions in each mouth zone*

The zone locator for mouths is more ad hoc and-complex than for the nose zone locator. An outline of the mouth zone locator is given in figure 54.

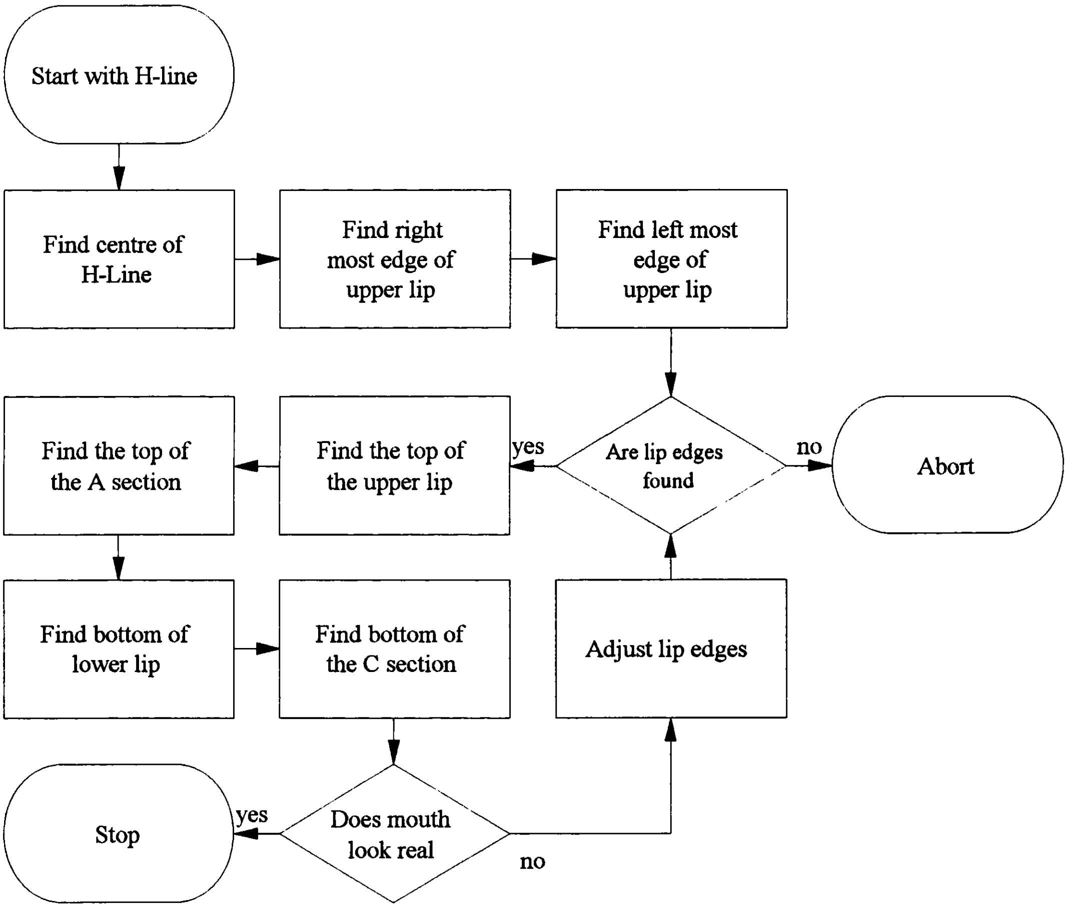

Start with H-line
Find centre of H-Line
Find right most edge of upper lip
Find left most edge of upper lip
Are lip edges found
no
Abort
yes
Find the top of the upper lip
Find the top of the A section
Find bottom of lower lip
Find bottom of the C section
Adjust lip edges
Does mouth look real
yes
Stop
no

Figure 54: *Outline of the mouth zone locator*

The edges of the lip are found by tracing the gradient directions {2, 3} or following the H-line, or following a vertical transition between gradient directions {2, 3, 4, 5} and {5, 6, 7, 8}.

The A, B, C, D zones are located by the algorithm shown in figure 55.

```
proc to locate the boundary between the A and B zone and the top of
the A zone {
    Mark and area above H-line[19];
    Select the bottom line of pixels in the area, (line = 1);
    while there are more points with GDs of {2, 3) than of GDs of {6,
    7} on the line {
        select the next line of pixels (line = line - 1);
    }
    mark the boundary between the A and B zone at the line;
    while there are more points with GDs of {6, 7) than of GDs of {2,
    3} on the line {
        select the next line of pixels (line = line - 1);
    }
    mark the edge of the A zone at the line;
proc to locate the boundary between the B and C zone and the bottom of
the C zone {
    Mark and area below H-line[20];
    Select the top line of pixels in the area, (line = 1);
    while there are more points with GDs of {6, 7) than of GDs of {2,
    3} on the line {
        select the next line of pixels (line = line + 1);
    }
    mark the boundary between the A and B zone at the line;
    while there are more points with GDs of {2, 3) than of GDs of {6,
    7} on the line {
        select the next line of pixels (line = line +1 );
    }
    mark the edge of the C zone at the line;
```

Figure 55: *Algorithm for locating the zone boundaries for the mouth*

[19] If r and l are the x coordinates of the left and right edges and m_x are the y coordinates of the H-line extended to the lip edges. The area searched is the points m_{x-t} for all x; $l+(r-l)/2 < x < r-(r-l)/2$, $t>0$.

[20] If r and l are the x coordinates of the left and right edges and m_x are the y coordinates of the H-line extended to the lip edges. The area search is the points m_{x+t} for all x; $l+(r-l)/2 < x < r-(r-l)/2$, $t>0$.

4.5 Performance of the PRODIGY algorithm

4.5.1 Zone location algorithms

The zone location algorithms were tested on subsets of faces from the face bank. The nose zone locator showed a success rate of around 98% on the noses tested (Images of size *full* to *small*) and figure 56 shows an example of a successful result.

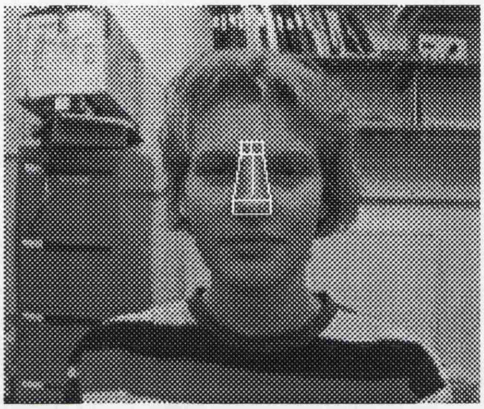


Figure 56: *The successful result of the nose zone location algorithm*

The nose zone locator was tested on face images that were slightly rotated and was found to work on many of these images as demonstrated in figures 57 and 58. The edges of the zones A and B are in error by a few degrees because the algorithm assumes that the angle of the edge of the nose is fixed.

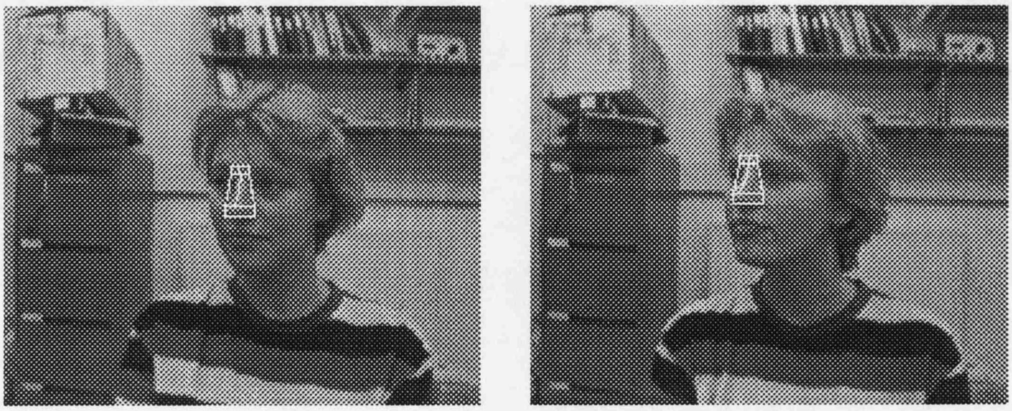


Figure 57 & 58: *Successful nose zone location of rotated faces*

The nose zone location algorithm was also tested on many faces with glasses. The algorithm still functions although the edges of zones A and B (left and right edges of the nose) are sometimes closer together, falling on the edge of the glasses. Figure 59 demonstrates the algorithm being successful on a face with glasses.

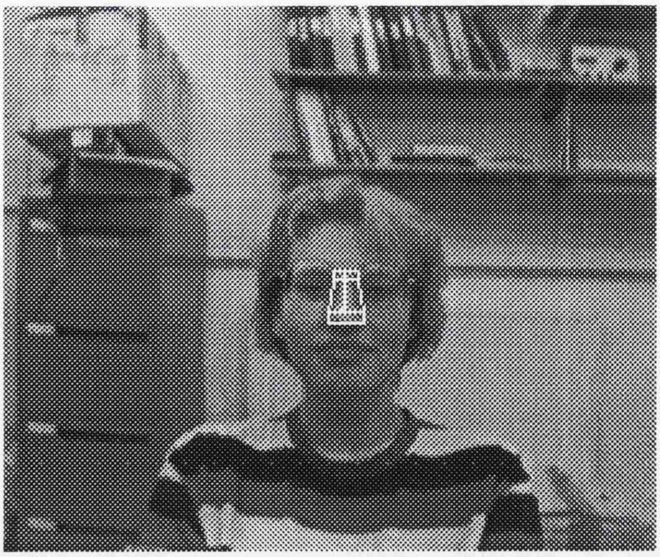


Figure 59: *Successful nose zone location on a face with glasses*

Some of the failures of the nose zone locator are shown in figures 60-62. These failures come from the test database of faces.

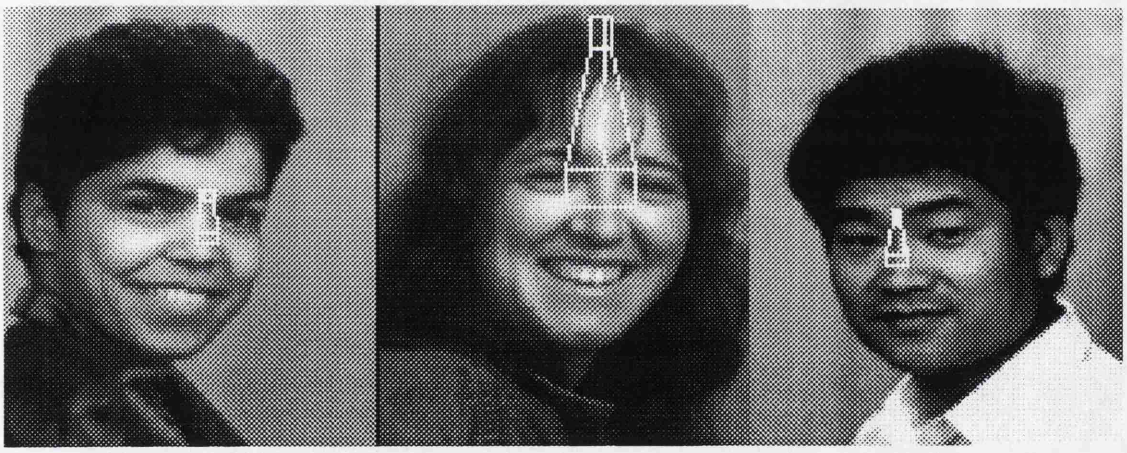


Figure 60, 61 & 62: *Some of the failures of the nose zone location algorithm*

The pictures highlight two sources of failure.

- In figures 60 and 62 the assumptions made by one of the zone edge finders was incorrect and the edge was missed. The failure in figure 62 may be because the face is from a different race than those in the original face bank and the nose is wider.
- In figure 61 the forehead is very bright. Because of this the original V-line extended right up the forehead. This confused the nose zone locator, which believed the nose to be high up on the forehead.

The mouth zone locator had a success rate of about 95% on faces from the face bank but only about 56% on the test database. This poor success rate on the test set is mainly due to moustaches and unconstrained expressions. Figure 63 shows an example of a successful result.

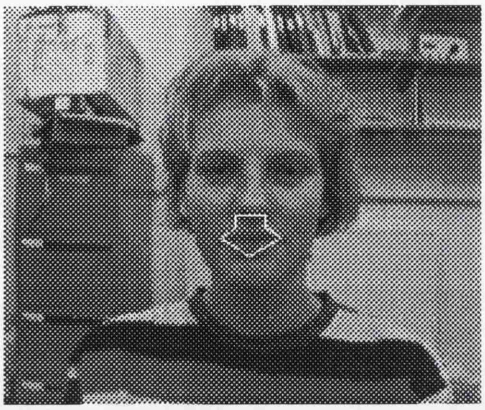


Figure 63: *A successful mouth zone location*

Although the mouth zone locator was not designed to work with mouths showing alternative expressions, in many cases it still succeeds (figure 64).

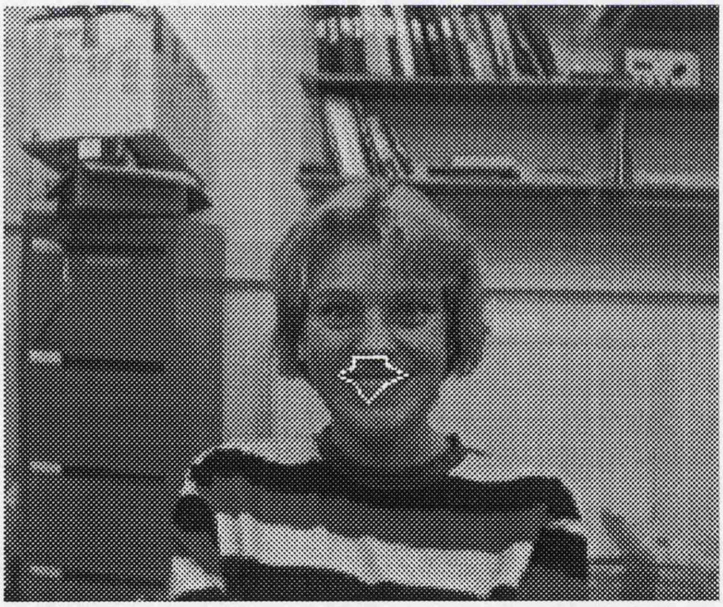


Figure 64: *A successful mouth zone location on a face with a cheesy grin*

There are many cases where the mouth zone locator fails and some of these are shown in figures 65-69.

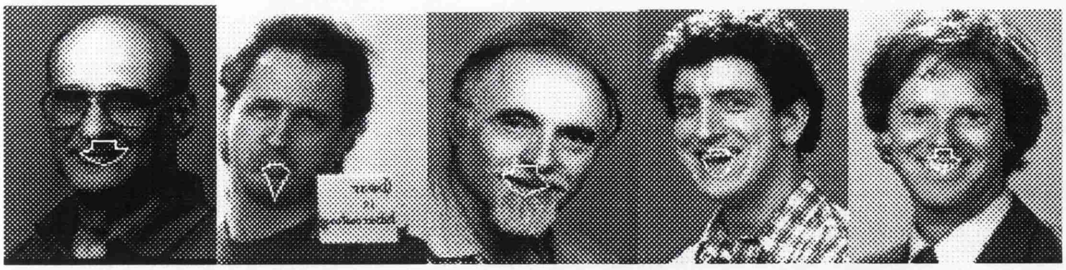


Figure 65, 66, 67, 68 & 69: *The types of failures that can occur with the mouth zone locator*

The failures seen in figures 65 and 67 are caused by a failure to locate one of the edges on the mouth, in figure 67 this is mainly due to the moustache. In figure 66 there is an excessively dark portion below the mouth and the algorithm has mistakenly assumed this is the mouth. In figures 68 and 69 the faces have prominent smiles, which confused the zone locator.

4.5.2 Likelihood Ratios

In this chapter three different ways of calculating likelihood were presented; the standard, vector or compromise method. The three methods were implemented using all the data from the face bank to establish the values of the variables (averages and standard deviations).

The three methods were then tested on all the faces in the bank and on a set of false faces. The purpose was to evaluate the difference between the likelihood for real noses and false noses.

PDFs (Probability Density Function) for real and false noses were found and are shown in figures 70, 71, 72.

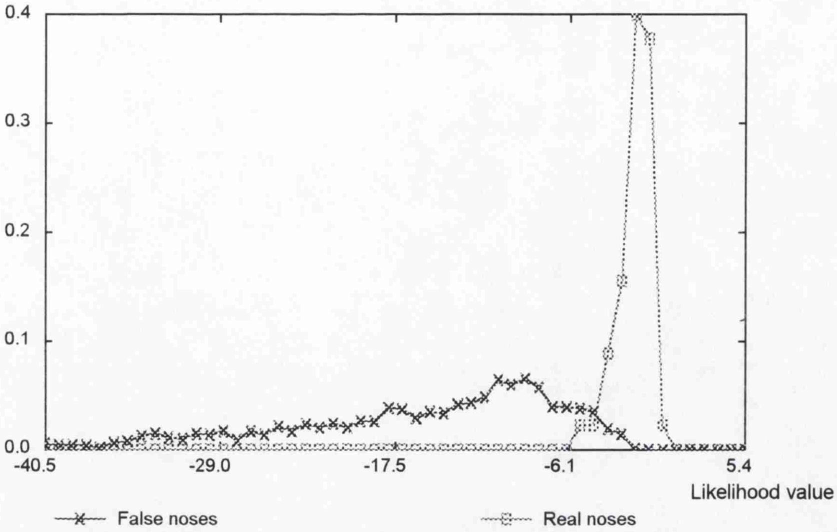


Figure 70: *PDFs of likelihoods of real and false noses using standard method*

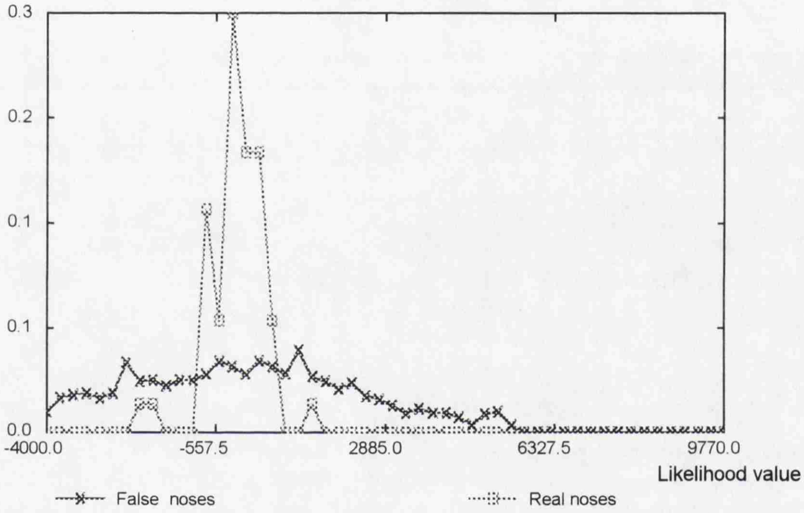


Figure 71: *PDFs of likelihoods of real and false noses using vector method*

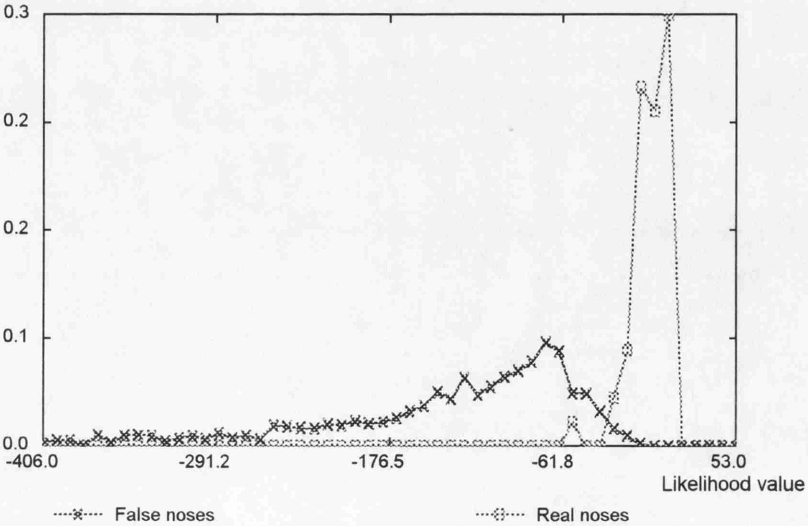


Figure 72: *PDFs of likelihoods of real and false noses using compromise method*

Figure 70 and 72 show that the standard method and compromise method have a reasonable separation between real noses and false noses. The two PDFs overlap which means that the method will not always manage to resolve the difference between real and false noses. Observations show that the real noses at the lower end of the PDF are usually due to poor zone location.

Figure 71 shows that the vector method does not resolve the difference between real and false noses. There is, however, a narrow band of values where it is more likely that noses are real rather than false.

It is not sufficient to only test the likelihood ratios on the training data alone. Therefore the small and compromise methods were tested on the independent test set of data and the PDFs are shown in figures 73 and 74.

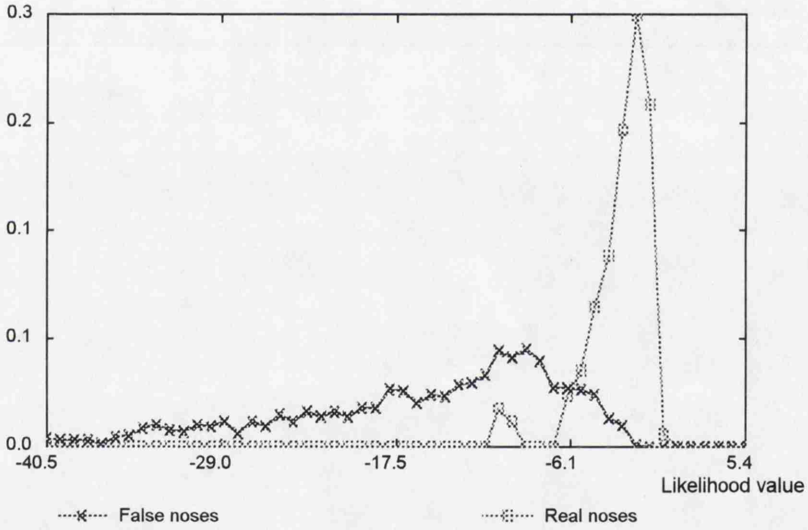


Figure 73: *Independent PDFs of likelihoods of real and false noses using the standard method*

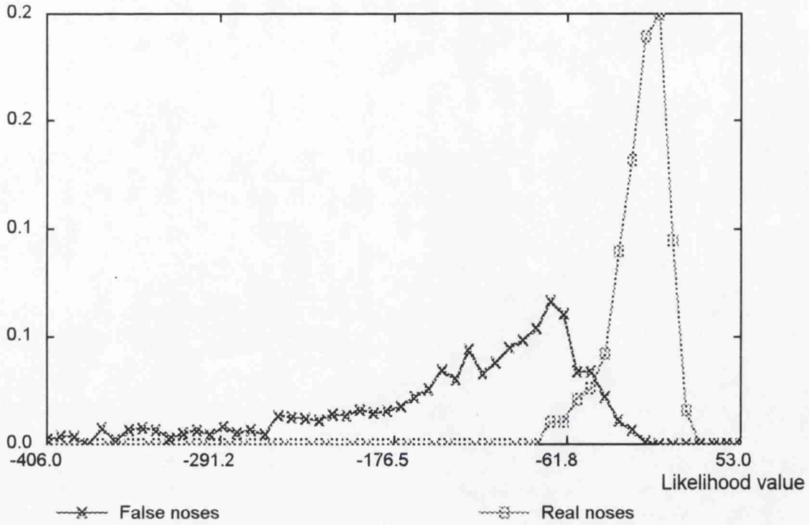


Figure 74: *Independent PDFs of likelihoods of real and false noses using the compromise method*

These graphs show that there is a slightly larger overlap between real and false noses. However, there is still a definite resolution between the PDFs. It is clear that all stages of this technique require improvements. However, as shown in the next section, poor results for one feature can be compensated for by good results for another feature (By use of a control structure, see chapter 5).

The PDFs of the likelihoods for real and false mouths was also calculated. The PDFs of the mouths in the face bank are shown in figures 75 and 76.

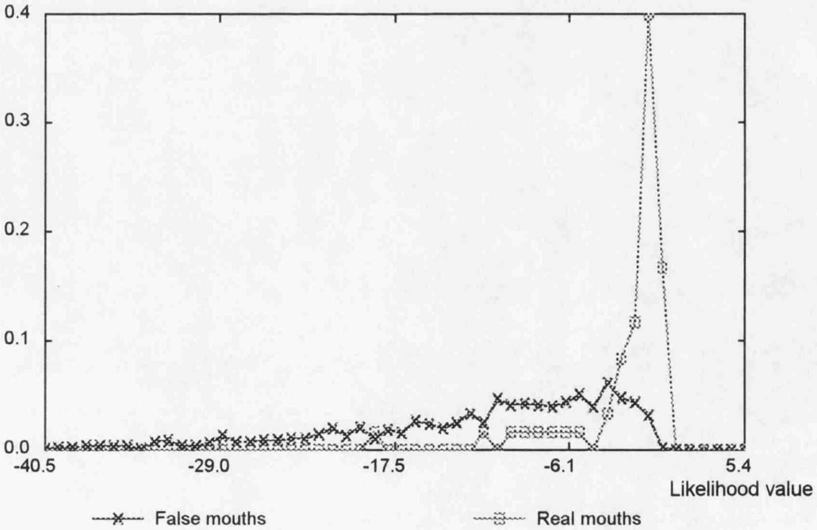


Figure 75: *PDFs of likelihoods of real and false mouths using the standard method*

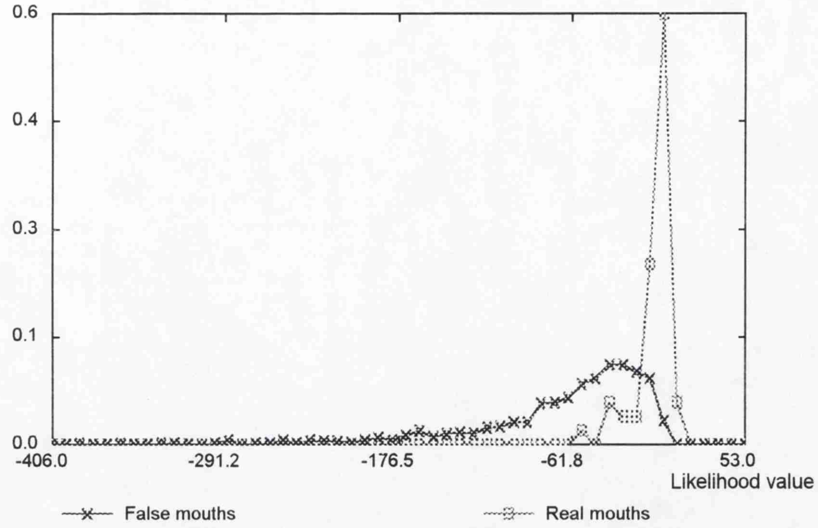


Figure 76: *PDFs of likelihoods of real and false mouths using compromise method*

The separation between real and false mouths is smaller than for real and false noses. This is mainly due to the poor zone locator. Chapter 6 discusses improvements that should be made to this technique in the future.

The mouth likelihood function was also tested with the independent face bank. The results of these tests are shown in figures 77 and 78.

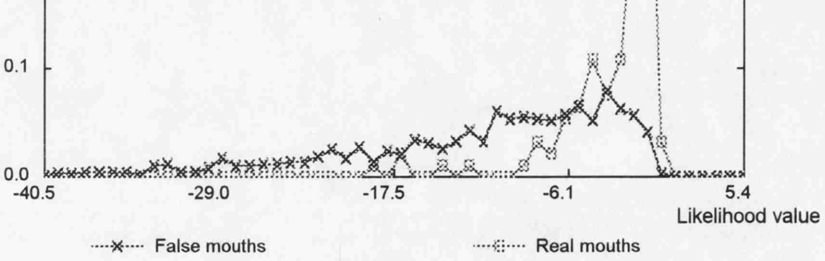


Figure 77: *Independent PDFs of likelihoods of real and false mouths using the standard method*

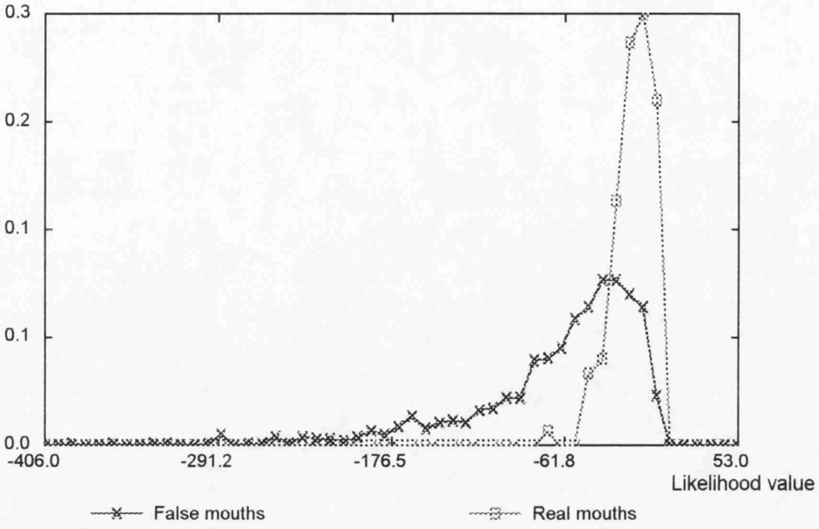


Figure 78: *Independent PDFs of likelihoods of real and false mouths using the compromise method*

Figures 77 and 78 have an even poorer separation between real and false mouths, although the algorithm still provides useful information. The independent database of faces consists of unconstrained faces, (i.e. there are many faces that have moustaches, smiles, grins and many mouths that have teeth showing), hence the poor result.

4.5.3 PRODIGY Algorithm

A simple test was devised to examine the PRODIGY algorithm. An image is preprocessed, scanned for possible features, a likelihood for each possible feature is found, and the feature with the highest likelihood is selected.

The performance was measured by carrying out this test on the set of independent images to find out in how many images the correct feature is found.

Tests show that, for the nose location algorithm using the standard method, 86% of noses were successfully located and with the compromise method, 88%.

For the mouth location algorithm the standard method successfully located 35% of mouths and the compromise method located 43% of the mouths.

These results cannot be compared realistically with other work because results have not been quoted in respect to individual independent location routines. Tock (1992) does however quote facial feature location results within the context of the complete system, he quotes 76% for the nose and 59% for the mouth. This shows that the nose locator described in this thesis is an improvement, but it is not possible to tell whether the mouth locator shows an improvement without putting it into Tock's (1992) control system. Both sets of results serve to confirm that locating the mouth is a more complex task than locating the nose. This is undoubtedly due mainly to the placidity of the mouth.

These results need to be put in context as although the required features do not always give the highest likelihood, they are often ranked in the top four or five. Further processing on these features either by combining face features (see chapter 5) or using further observations about the facial features such as colour will result in improved performance.

5. Face Location System

This Chapter draws together the work described in previous chapters to present a complete face location system. It combines the peak and trough preprocessors and the PRODIGY techniques using further statistical methods similar to those used in the PRODIGY. A simple face recognition system is then presented which compares cue faces with people who have been previously photographed.

5.1 Control System

Several control systems were described in section 2.5. The discussion mentions two types of control structures; the serial approach, such as Kanade (1977), and the independent feature location approach, such as Craw et al's (1991) blackboard based system.

The approach chosen uses the independent approach and is demonstrated in Figure 79. This is a tightly defined approach like the serial approach, as compared to the blackboard system, which is a loosely defined system and allows multiple execution paths.

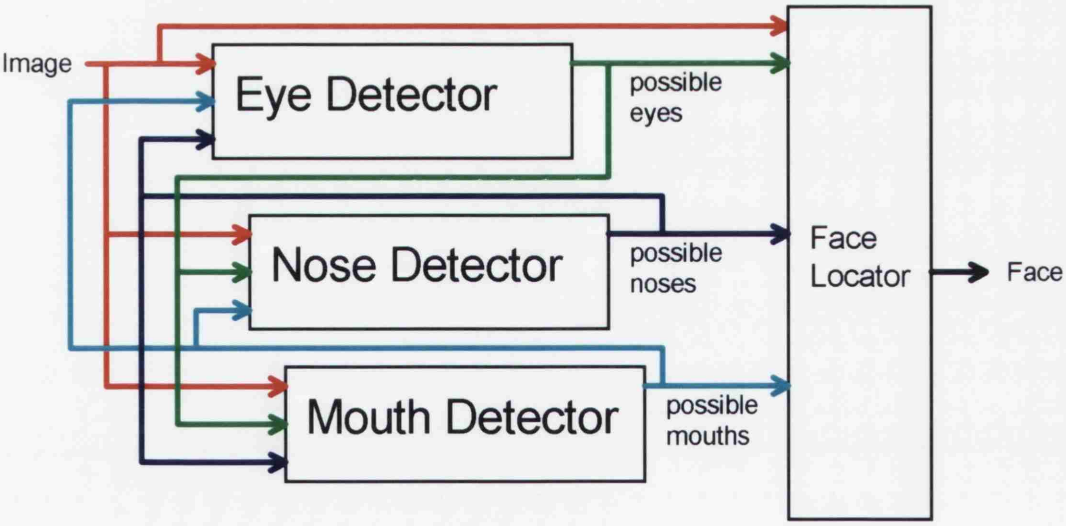


Figure 79: *Block diagram of a system for locating faces*

The system above shows several independent facial feature detectors that produce a list of possible features along with a confidence value for each possible feature. The list is not only passed onto the face locator but to other feature detectors, which enable the search to be directed to appropriate areas of the image. The nose and mouth detectors that are used in the system are those described in the previous chapter, i.e. those using the PRODIGY technique. The face location part combines the output of these detectors using statistical methods similar to the PRODIGY and is described in section 5.4.

5.2 Feedback Loop

The actual implementation of the feedback loop in figure 79 is described by the following:

> The nose detector supplies the eye detector with information that it has found '*x*' possible noses. The eye detector then checks its possible eyes to test if they are close to the top of a nose. Each eye that is not close to the top of the nose is removed. The eye detector then passes its possible eyes to the nose detector and the nose detector then checks its possible noses for nearby eyes.

This process is repeated several times between the eye, nose and mouth. This feedback reduces system failures and computation time. This is because the number of 'false' eyes, noses and mouths are cut and hence the time taken to check each of the remaining possible features, or combinations of them, is lower.

5.3 Face Model

The face location part of the system in figure 79 has similarities to Fischler and Elschlarger's (1973) templates and springs method. In their method they designed a model of the whole face as shown in figure 80. To locate a face, first a predicted location of each facial feature is given. A cost is calculated as to how much each predicted feature's shape and texture deviates from the model features. Then another cost is calculated which shows how much the relative locations of the predicted features deviate from the model. The predicted facial feature locations are then adjusted until the cost is minimised.

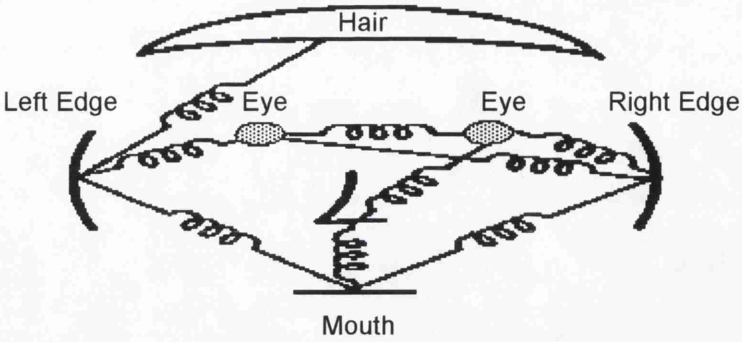


Figure 80: *Fischler and Elschlager's springs*

The total cost of this model for a given location is shown by equation 21. The cost function is minimised to find the face on an image.

$$\sum \text{Template costs} + \sum \text{Spring cost} \qquad (21)$$

A template cost is a figure relating to the amount of mismatch of the template and any part of the image. The spring cost is calculated by how much it deviates from the average length springs. i.e. it is a measure of the effort needed to stretch or compress the spring.

This method is not unlike the deformable templates proposed by Yuille et al (1988) in that the model is deformed until it fits the shape of the actual face in the image.

The implementation of the 'springs' method used in this thesis differs somewhat. Fischler and Elschlarger (1973) system was designed to refine the location of objects, whereas the PRODIGY has already specified facial feature locations. We use the springs only to check combinations of facial features to test if they are structurally compatible.

5.4 Implementation of the 'spring' control system.

The 'spring' method that was developed uses likelihood ratios in a similar manner to the PRODIGY technique. This diverges from the Fischler and Elschlarger (1973) 'springs method' which uses cost functions. To locate a face the likelihood function given in equation 26 is maximised.

$$\sum_{a=1}^{\text{no of features}} lr(a,n_a) + \sum \text{spring likelihood ratios} \qquad (22)$$

Function *lr(a,n$_a$)* is the likelihood ratio for n_a^{th} proposed facial feature of type *a*. The likelihood ratio is found by the PRODIGY algorithm (or similar algorithm) for that feature as described in section 5.4.1.

Type *a*=1 represents a **nose**, *a*=2 represents a **mouth**, *a*=3 represents the **left eye**, *a*=4 represents the **right eye**.

A spring likelihood ratio is calculated from the distance between each and every feature. It is assumed that the distance between each 'real' and each 'false' feature has a normal distribution and hence simplifies the calculation of the likelihood ratio function.

5.4.1 Feature likelihood ratio functions

The likelihood of the mouth and nose features are found by dividing each value from the real PDF by the corresponding values of the false PDFs. The values of the PDFs are the output of the PRODIGY functions and are called Nvalues for noses and Mvalues for mouths. The PDFs from the compromise method were used as shown in figures 72 and 74. The graph in figure 81 shows the likelihoods of noses.

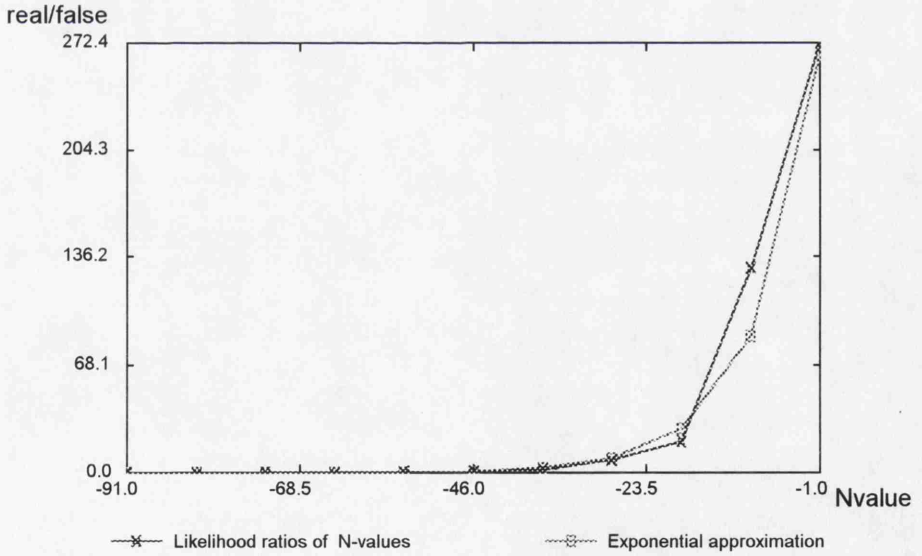


Figure 81: *Likelihood ratio of Nvalues between -91 and -1*

The curve is exponential in shape, so an approximation to the exponential curve was estimated. The approximation shown in figure 81 is given in equation 23.

$$L(Nvalue) = \exp(Nvalue / 8) * 300 \qquad (23)$$

Similarly the likelihood ratios of the Mvalues are shown in the graph in figure 82.

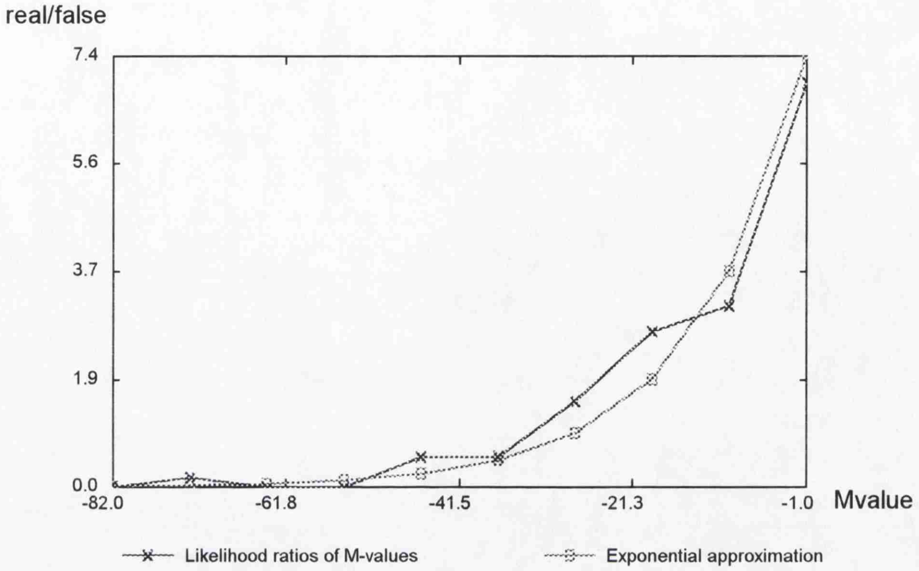


Figure 82: *Likelihood values of Mvalues between -82 and -1*

and an exponential approximation was devised as shown in equation 24.

$$L(Mvalue) = \exp(Mvalue / 13) * 8 \tag{24}$$

5.4.2 Spring likelihood ratio functions

The likelihood ratio function introduced in equation 4 (page 63) is used for calculating the spring likelihood values. A simplified version derived from equation 8 (page 64) is given in equation 25 and shows the normal distribution of real spring lengths divided by the normal distribution of false spring lengths.

$$L(x) = \exp(ax^2 + bx + c) \tag{25}$$

where x is the proposed spring length and

$$a = \frac{\sigma_r^2 - \sigma_f^2}{2\sigma_r^2\sigma_f^2},\ b = \frac{\mu_r\sigma_f^2 - \mu_f\sigma_r^2}{\sigma_f^2\sigma_r^2},\ c = \frac{\mu_f^2\sigma_r^2 - \mu_r^2\sigma_f^2}{\sigma_f^2\sigma_r^2} + \log\left(\frac{\sigma_f}{\sigma_r}\right) \text{ and}$$

μ_r = average length of real spring

σ_r = standard deviation of real spring length

μ_f = average length of false spring
σ_f = standard deviation of false spring length

Therefore, the likelihood ratio can easily be determined by finding the average and standard deviation of real and false springs. To calculate the false spring averages a random sample of false features was selected and the distances to other false features were calculated.

In practice the distance between the nose and the mouth is not an appropriate measure because it varies with the scale of the image. To ensure scale independence relative values were calculated using length ratios. The ratio chosen for the nose to mouth distance was:

ratio = (*nose length/distance between the nose and mouth*)

The average and standard deviation of the ratio were found by analysing all the faces in the face bank of type 1-4 (*large* to *small*). The average and standard deviation of the ratio of false faces were found by analysing thousands of random false faces[21].

5.4.3 Face likelihood

A version of the likelihood 'springs' methods was constructed combining the nose likelihood ratio, mouth likelihood ratio and the spring likelihood ratio. The eyes are not included in this likelihood function as the eye preprocessor does not give a confidence value. The likelihood function derived is given equation 26.

$$L(Nvalue, Mvalue, x) = \exp(Nvalue/8) * 300 * \exp(Mvalue/13) * 8 * \exp(ax^2 + bx + c) \qquad (26)$$

where *Nvalue* is the output of the nose PRODIGY, *Mvalue* is the output of the mouth PRODIGY and *x* is the spring ratio.

Taking the log likelihood and removing constants gives equation 27.

[21]The analyses showed that $\mu_r = 0.57$, $\sigma_r = 0.10$, $\mu_f = 0.77$, $\sigma_f = 2.41$. Therefore a = -47.5, b = 54.2 and c = -12.3

$$L(Nvalue, Mvalue, x) = Nvalue / 8 + Mvalue / 13 + ax^2 + bx \qquad (27)$$

Note that this equation includes weighted versions of the Nvalue and Mvalue. This is because Nvalues and Mvalues were originally log likelihoods. The weights give precedence to the relative reliability of the values produced by the PRODIGY algorithm. In this case as the nose locator is more reliable it is given a higher weight.

5.5 Face location

The likelihood function in equation 27 can be used directly to locate a face on an image. This is done by finding the value of the function for every combination of proposed features. The combination with the maximum likelihood is chosen as the real face.

This involves little processing for the system proposed (with only two features). e.g. if there are 40 proposed mouths and 40 proposed noses then the function is calculated 40x40 = 1600 times[22]. If the system is expanded later by adding 40 proposed right eyes and 40 proposed left eyes this would increase to 2.5million iterations, which would take a significantly[23] longer time. Therefore, an alternative method for testing all the combinations would be necessary such as gradient descent, genetic algorithm (Robertson and Sharman, 1990) or by implementing the process in parallel.

5.5.1 Shape free faces and position/scale free faces

Many have used face location as a preprocessor to face recognition (Gallery et al, 1992, Craw et al, 1992, Jia and Nixon, 1992). Where face recognition has been tested on unlocated faces the results have been poor. For example Turk and Pentland (1991) show recognition of around 30% for unlocated faces.

[22] The likelihood function has 5 multiplications and 4 additions. Total for 1600 iterations = 8000 multiplications and 6400 additions. If a computer can do 1million multiplications per second and 10million additions this would take 8 msec.

[23] 15.36million multiplications and 10.24 million additions. Using the same assumptions as the previous footnote this would take 16.384 sec.

Applications, such as face recognition, that use ‘located faces’ usually require an extracted face as and input (Gallery et al, 1992, Craw et al, 1992, Jia and Nixon, 1992). This is easily done by geometrically transforming the image so that the face is moved to the centre of the image and is scaled to a fixed size. This is called a position/scale free (ps-normalised) image and the transformation is given in equation 28.

$$i', j' = f(i, j) \text{ for all } i, j \in \{C\} \qquad (28)$$

where *i, j* are coordinates on the original image mapped to new coordinates on the position/scale free image by function *f*. *C* represents all possible image coordinates. Similar transformations have been used by Craw et al's (1992) (Shape free image) and Shackleton and Welsh's (1991) (Geometrically normalised image).

The process for producing the position/scale free follows the stages given in figure 83.

```
Locate the face;
Centre the face in the image (translation);
Rotate the face so that the eyes are on a horizontal line (rotation);
Scale the image of the face vertically so that the mouth falls on a fixed point (vertical scaling);
Scale the image of the face horizontally so that the eyes fall on fixed points (horizontal scaling);
```

Figure 83: *Process for scaling and translating the face to fixed place on an image*

This system has been implemented and the output of such a transformation is shown in figures 84-87. Note that this transformation does not remove expression information, lighting effects or horizontal turning of the face (face tilt).

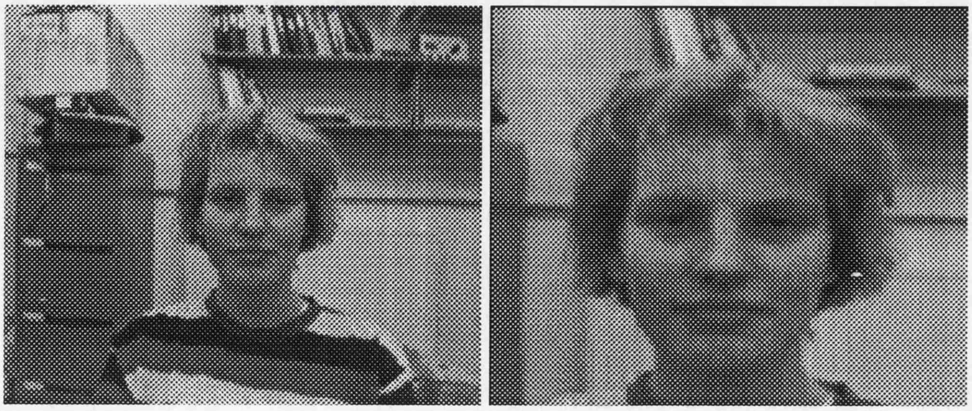


Figure 84: a) TRACY3 b) *PS-Normalised image of TRACY3*

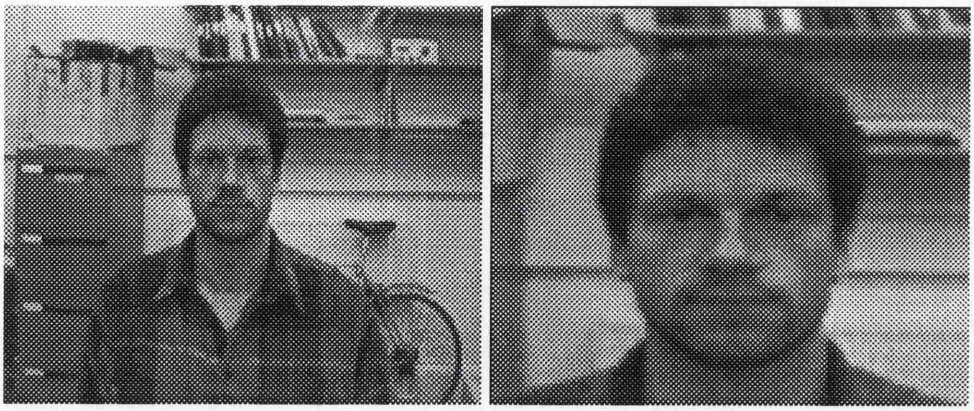


Figure 85: *a*) THOMAS3 b) *PS-Normalised image of THOMAS3*

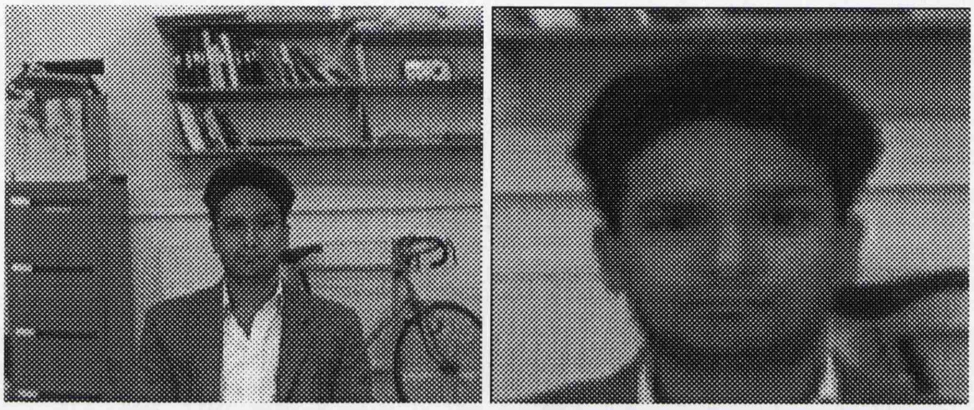


Figure 86: *a*) DEEP4 b) *PS-Normalised image of DEEP4*

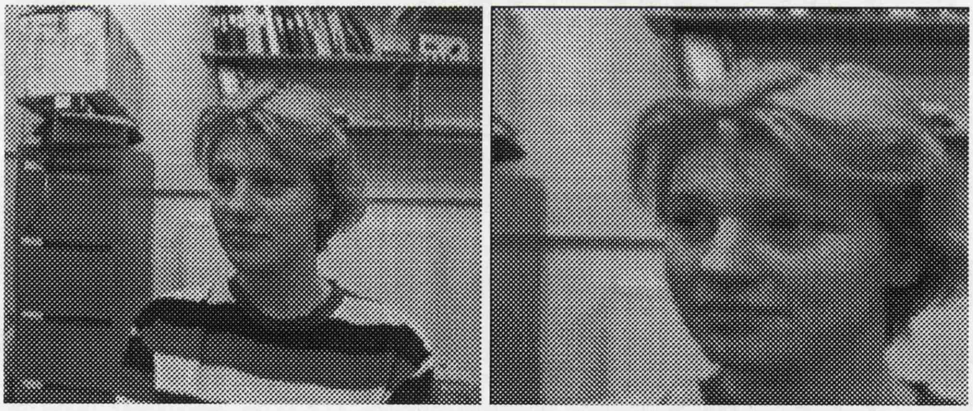


Figure 87: *a*) TRACY8 b) *PS-Normalised image of TRACY8*

Figure 87 shows a rotated face. The face location algorithm still works and the ps-normaliser still places the eyes, nose and mouth in fixed positions.

Note that the ps-normaliser is different from shape freeing faces in that the shape information is still retained[24]. The only shape information removed is the aspect ratio of the face, which is fixed after the transformation.

5.5.2 Face location performance

A face location algorithm was implemented using the likelihood function presented in section 5.4.3. The face location algorithm was tested on all the faces in the face bank and on all the test faces.

Of the original face bank the program located 86% of the 'non rotated' faces correctly. The system assumed no knowledge of the position of the face and the size of the face could be between 20-60 pixels wide. Of the test set of independent images (which had faces with glasses and beards among them) 58% of the faces were located correctly.

Kanade (1977) quoted a figure of 75% location (20 people); but this system also had prior knowledge of the size of the face, orientation and possible location of the face - if his system was tested on the face bank it would find none of the faces. Tock (1992) quotes a figure of 86% on his set of faces which are all full face and of a reasonable size. Tock's program is a blackboard system and is expandable; for example by adding the feature detectors presented in this thesis. His system will choose the most reliable feature locators in its control structure. This compares with the likelihood equation for locating faces that weights reliable locators, although the weights are not modified automatically. Waite and Welsh (1990) achieved face location results of between 54-85% using a snakes method for locating the boundary of the head. This method requires the head to be on a white background because is searches for edges. None of these other systems described were tested on images as diverse as those in the independent test set.

The results of the PRODIGY based system above show that there is more development needed before the system is robust. However, as compared to the other systems cited above the technique shows a gain in performance. The system does not require the input faces to be in the middle of the image or on a plain background or of a fixed scale. The results on the independent test database show that the system has flexibility and is not constrained to certain types of face images.

[24] As the shape information is retained it is not necessary to have a separate shape vector.

5.6 Face recognition

As mentioned in the introduction and background to this thesis automatic face recognition by machine is one common application of face research. To test the usefulness of the face location techniques described in this thesis a simple face recognition program was devised. This section describes the face recognition process.

5.6.1 Face recognition : a simple technique

To compare face images a simple correlation technique was designed. Rectangular portions ($\Re$) of each face enclosing the eyes, nose cheeks and chin are correlated. As in Craw et al's (1992) method it was decided that the hair was too variable to be included.

If *C(i,j)* is the cued image and *I(i,j)* is a face in the training set then the correlation between the images is given in equation 29. This is derived from the standard correlation methods such as found in Milton and Arnold (1986, p157).

$$\frac{n\sum_{i,j\in\Re}C(i,j)I(i,j)+\sum_{i,j\in\Re}C(i,j)\sum_{i,j\in\Re}I(i,j)}{\sqrt{\left(n\sum_{i,j\in\Re}C(i,j)^2-\left[\sum_{i,j\in\Re}C(i,j)\right]^2\right)\left(n\sum_{i,j\in\Re}I(i,j)^2-\left[\sum_{i,j\in\Re}I(i,j)\right]^2\right)}} \qquad (29)$$

The correlation between the cue image and all the images in the pool is found and the image with the highest correlation is assumed to be a picture of the same person.

5.6.2 Face recognition performance

The face recognition program described above was tested by placing someone in front of the camera, digitising the face, locating the face, and comparing it to a set of previously located faces in a pool (candidates). The pool consists of several pictures of about eight people.

Tests showed that whenever the face was located correctly then the face was also successfully recognised. It is expected that failures would occur if the set of possible candidates is increased.

The fact that ‘face recognition’ was demonstrated successfully, using a simple program, adds weight to the theory in section 2.2 that face location is an essential preprocessor to face recognition.

6. Observations and Conclusions

This Chapter examines the experimental results and draws together some conclusions and inferences from the results. Facial vision engineering is a growing subject, core methods and techniques are still emerging and difficulties in the unconstrained face location problem require further research. This study has, however, made advances in the field in reducing image constraints while retaining computational efficiency. In this chapter an examination of each technique is made. Their properties, advantages and disadvantages are discussed. Finally, future research on each technique is suggested.

6.1 Peak and Trough Preprocessor

The peak and trough preprocessors propose a number of locations on an image likely to be a particular facial feature. This is done by examining intensities on an image to produce V-lines, which are vertically connected peaks in the image, and H-lines, which are horizontally connected troughs in the image. The V-lines are proposed noses and the H-lines are proposed mouths. Eyes are proposed at points where a trough in intensity is found in all directions. This technique is based on observations that have been made on numerous face images.

The implementation of the peak and trough preprocessors is computationally efficient. They require approximately 11 instructions per pixel. For a 256x256 image this would account for 720896 instructions per image. On an 10 MIP machine this would take 70 msecs per image. This compares with the morphological methods for finding peak and trough fields as implemented by Yuille et al (1988), which are by nature less efficient. These preprocessors are robust for face images between 20 and 64 pixels wide (see table 3, page 23). This is supported by the fact that there were no true rejects of noses, or mouths on all face images of size *large* to *small* when the technique was tested on the face bank.

The performance of the peak and trough algorithms is affected by the filters and the filter parameters and the threshold value. Larger filter parameters increase the performance of the V-line proposer, but too large a value merges the nose into the face. The size of the filter parameter for proposing H-lines has a minimal effect on the performance of the trough preprocessor. Lower threshold parameters increase the number of false features that are found which decreases the likelihood of the PRODIGY locating the correct feature. Higher thresholds shorten the length of the H/V-lines, which increases the probability of the PRODIGY failing to define nose or mouth zones.

Future work on the preprocessors would focus on developing a technique that maximises the length of the H/V lines without increasing the number of false features. One method is to start with a large threshold and then slowly allow the threshold to decrease without allowing new proposed facial features to appear. Different types of filters, such as matched filters, should be tried with the aim of enhancing the facial features that the preprocessors are proposing.

6.2 The PRODIGY algorithm

The PRODIGY algorithm takes as its input the proposed features from the peak and trough preprocessors and gives each a confidence value. This value is a likelihood of the line being a real rather than a false feature. Producing the likelihood values involves first locating zones around each V/H-line, then calculating the proportions of eight gradient directions within each zone. The system then compares the proportions with the average and standard deviations of real and false facial features, producing the likelihood value, which are called N-values and M-values for noses and mouths respectively.

The PRODIGY technique is scale independent because the proportion of gradient directions in each zone is found rather than the absolute number. Because gradient directions are based on the direction of change in intensity, rather than magnitude, the technique is independent of absolute image intensity. The use of statistics in the PRODIGY technique performs two functions: first, they allow the removal of statistically impossible proposed noses; and second, they provide confidence information about the proposed facial features without introducing 'true rejects'. This confidence information is subsequently used in the face location control structure.

The performance of the PRODIGY algorithm is affected mainly by the zone location algorithm. Unlike other parts of the system, failure of the zone location section can cause 'true rejects'. For the mouth zone locator it has been shown to cause up to 5% 'true rejects' and hence causes a weak link in the chain (the nose locator causes up to 2% 'true rejects'). The zone location algorithms were based on observations but in this case these observations proved inadequate. The choice of the statistical method affects the amount of resolution between real and false facial features. Of the methods tested, the compromise method proved to be the best (see section 4.3.9). This method is a combination of a multi and single dimensional analysis. The results show that this combination approach, which uses cross correlation information, reduces the impact of any assumptions that are made during the statistical analysis.

Future work on the PRODIGY must tackle the failures in the zone location algorithms. Areas of research will examine the use of statistics to define zone boundaries, introduce checks on the credibility of the zones found and generate multiple zone sets for each proposed feature. Currently the zone locators generate one zone set for each facial feature, which means that the algorithm must be able to cope with multiple expressions and classes of feature. If a zone locator is made to produce several zone sets representing different classes and expressions, then the definition of the boundaries on each zone can be tightened. The likelihood technique performed on gradient directions can also be applied to image grey levels, colour levels, edges, and other image operators. Future research will analyse the use of these other indicators with the aim of increasing the resolution between real and false likelihood values.

6.3 Face location control system

The face location control system first removes proposed facial features that cannot be combined with any other proposed facial features to produce a structurally feasible feature set. This is performed using a multiple feedback loop. The system uses a likelihood function based on structural feasibility and the confidence of the proposed facial features to find the most feasible combination of facial features. The proposed features are generated by the peak and trough algorithms and the confidence values are generated by the PRODIGY. A feature of the control system is that the real mouth and nose do not have to be ranked first in the list of proposed features for the combination to produce the most likely face. This supports the use of the statistical methods to govern a face location control system.

Many face location techniques (see chapter 2) require the constrained face images to be presented to the system. These constraints are listed in chapter 1. The face location algorithm designed in this research reduces the level of constraints required. Individual comments on each constraint category are given in table 15. These comments are based on performance tests which were reported at the end of chapters 3-5.

CATEGORY	SUB-CATEGORY	COMMENT
Translation	scale	Independent to scale for face images greater than 25 pixels wide
	position	The face image can be in any position in the image
	orientation	The face must be approximately within 15 degrees of the vertical
Pose	occlusion	The eyes, nose and mouth must not be occluded
	rotation	Head can rotate by a small amount[25]
	face tilt	Eyes should level with the camera
Lighting	level	Independent to absolute image intensity level
	direction	There is a small tolerance to lighting direction[26]
Noise	camera	The technique has been tested successfully on many cameras
	intensity resolution	The images should have 16 grey levels or more
	clutter	Independent to background clutter
Artefacts	moustaches	Not tolerant
	beards	Not tolerant
	glasses	Tolerant
	sex	Independent
Expression		Requires a straight, expressionless face

Table 15: *Comments and tolerances on the categories of constraints applicable to facial images*

Table 15 shows that the face locator is independent of scale, position, some head movement, absolute lighting level, camera, background clutter, digitising equipment, gender and faces with glasses. This shows a significant improvement in independence to constraints to the face location techniques used by Kanade (1977), Brunelli and Poggio (1992) and Jia and Nixon (1992) which require fixed scale and plain background.

This method was tested successfully on 86% of the 'non-rotated' faces in the face bank and 58% of the faces in the test database. As described in section 5.5.2 these results compare favourably with other reported face location results. This suggests that the

[25] The amount of has not been quantatively determined

[26] The amount of has not been quantatively determined

number of constraints required of face images has been reduced without resulting in decreased peformance

The face location system has been tested on several computer systems. On a 386 20Mhz PC the program takes about 1 minute. On a 486 33Mhz PC it takes about 15 seconds. And on a SUN SparcStation LX it takes about 25 seconds. This is an improvement on the deformable template eye locator reported by Yuille et al (1988) which takes about 5 minutes on a SUN4.

Future work on the face locator will be in improving the zone locators as previously described and increasing the number of face features that are included in the control structure. The eye proposers do not as yet provide a confidence value. For this reason they are not included in the statistical part of the control structure. The face locator can also be improved by adding extra feedback to reject output that is clearly not a face. Work also needs to be carried out to reduce more of the image constraints, especially pose, facial expressions, moustaches and beards. The face location system fails with faces smaller than 20 pixels wide. This is because the features used to locate the face are too small to be detected with the existing methods. A new technique needs to be developed to find these small faces. This could be done by locating the body of a subject as well as the face.

Specific enhancements of this face location technique can be made depending on the application. If multiple images of a subject are available, as when video cameras are used, the success rate can be improved by choosing the image that was located most accurately. In a security entry application the lighting conditions can be controlled so as to allow the program to be tuned within that limited environment.

6.4 Face recognition

A simple face recognition algorithm has been evaluated and found to work in a number of tests. Although the face recognition results of various researchers cannot easily be compared, the simple algorithm described has similar success rates to other reported methods that do not use face location. Despite the fact that some of these algorithms use neural networks, the template matching face recognition algorithm described produces comparable results. It can be concluded from this and the evidence listed in chapter 2 that the face location algorithm is a necessary step in face recognition. Without knowing exactly where all the features are on a face, knowing where the edges of the features are, or knowing the shape of the features it is impossible to compare one face with another.

6.5 Final remarks

This research has tackled many problems in designing a reliable unconstrained face location system. The techniques that have been demonstrated reduce limits on the size, scale and background clutter of the images containing faces to be located. A method for locating the surfaces of objects has been proposed and successfully implemented, especially for nose location. At the heart of the method are feature proposers looking for local peaks and troughs in intensity on a grey scale image, and a statistical analyser, called PRODIGY, that determines characteristics of the reflective surfaces on human facial features.

The basis of a robust face location system has been designed and demonstrated to perform well on a random set of face images. The value of this system has been clearly demonstrated in the development of a simple but effective face recognition system. The research leaves open the path for many more research opportunities which will hopefully succeed in fulfilling the long term goal of building an fully unconstrained face location system.

7. References

Akamatsu S., Sasaki T, Fukamachi H., Masui N. and Suenaga Y. An Accurate and Robust Face Identification Scheme, *International Conference on Pattern Recognition,* 1992.

Allinson N.M., Ellis A.W., Flude B.M. and Luckman A.J. A Connectionist Model of Familiar Face Recognition, *IEE Colloquim on 'Machine Storage and Recognition of Faces',* January 1992, 5/1-5/10.

Ballard D.H. and Brown C.M. Computer Vision, *Englewood Cliffs, New Jersey: Prentice-Hall, Inc.,* 1982.

Brunas J., Young A.W., and Ellis A.W. Repetition priming from incomplete faces: Evidence for part to whole completion, *British Pschological Society,* 1990, 81, 43-56.

Brunelli R. and Poggio T. Face Recognition through Geometrical Features, *European Conference on Computer Vision,* 1992, 792-800.

Burtons M.A., Bruce V. and Johnston A. Understanding face recognition with an interactive activation model, *Journal of Psychology,* 1990, 81, 361-380.

CCITT Recomendations H.261. Code For Audiovisual Services at nx385kbits/s, *CCITT IXth Plenary Assembly,* 1989.

Compuserve Incorporated. A standard defining a mechanism for the storage and transmission of raster-based graphics information, *Compuserve, 5000 Arlington Centre Blvd. Columbus, Ohio 43220,* June 1987.

Coombes A., Richards R., Linney A., Hanna E. and Bruce V. Shape-Based Description of the Facial Surface, *IEE Colloquim on 'Machine Storage and Recognition of Faces'*, January 1992, 9/1-9/4.

Craw I. and Cameron P. Face Recognition by Computer, *Proceedings of British Machine Vision Conference,* September 1992.

Craw I., Ellis H. and Lishman J.R. Automatic Extraction of face features, *Pattern recognition Letters,* February 1987, 5, 183-187.

Craw I., Tock D. and Bennett A. Finding Face Features, *European conference on computer vision, Italy,* May 1992.

Dawson B.M. Introduction to Image Processing Algorithms, *BYTE,* March 1987, 169-186.

Edwards A. W. F. Likelihood, *Cambridge University Press,* 1972, 8-11.

Elber G. GIF Library document, *Public Domain Software Library, Lancaster,* August 1989.

Fischler M.A. and Elschagler R.A. The Representation and Matching of Pictorial Structures, *IEEE Transactions on Computers,* January 1973, 22:1, 67-92.

Forchheimer R. and Fahlander O. Low Bit Rate Coding through Animation, *Abstracts of the Picture Coding Symposium,* 1983, 113-114.

Gallery R. and Trew T.I.P. An Architecture for Face Classification, *IEE Colloquim on 'Machine Storage and Recognition of Faces'*, January 1992, 2/1-2/6.

Goldberg D.E. Genetic Algorithms in Search, Optimization, and Machine Learning, *The University of Alabama, Addison-Wesley,* 1989.

Goldstein A.J., Harmon L.D. and Lesk A.B. Man-machine interaction in human face identification, *Bell System Technical Journal,* February 1972, 51:2, 339-427.

Goldstein A.J., Harmon L.D. and Lesk A.B. Identification of Human Faces, *Proceedings of the IEE,* May 1971, 59:5, 748-760.

Gonzalez R.C. and Wintz P. Digital Image Processing, *Addison Wesley,* 1987, 185-186.

Hancock P.J.B. GANNET: Design of a neural network for face recognition by Genetic Algorithm, *Genetic Algorithm, Neural Networks and Simulated Annealing workshop, Glasgow,* March 1990.

Huang J.S., Wu C.J. Human Face Profile Recognition by Computer, *Pattern Recognition,* 1990, 23:3, 255-259.

Jia X. and Nixon M.S. On Developing an Extended Feature Set for Automatic Face Recognition, *IEE Colloquim on 'Machine Storage and Recognition of Faces',* January 1992, 8/1-8/4.

Kanade T. Computer Recognition of Human Faces, *Interdisciplinary Systems Research, Birkhause, Basel, Stuttgart,* 1977.

Kaya Y. and Kobayashi K. A Basic Study on Human Face Recognition, *Frontiers of Pattern Recognition : Academic Press,* 1972, 265-289.

Kirby M. and Sirovich L. Application of the Karhunen-Loeve Procedure for the Characterization of Human Faceds, *IEEE Trans of Pattern Analysis and Machine Intelligence,* January 1990, 12:1, 103-108.

Krzanowski W.J. Principles of Multivariate Analysis, *Oxford University Press,* 1988, 205.

Milton J.S. and Arnold J.C. Introduction to Probability and Statistics, *McGraw-Hill International Editions,* 1986, 103.

Nakamura O., Mathur S. and Minami T. Identification of Human Faces based on Isodensity Maps, *Pattern Recognition,* 1991, 24:3, 263-272.

Nixon M. Eye Spacing Measurement for Facial Recognition, *Proceedings of International Society for Optical Engineering,* August 1985.

Reisfeld D. and Yeshurun Y. Robust Detection of Facial Features by Generalised Symmetry, *International Conference on Pattern Recognition,* September 1992, 11, 117-120.

Rickman R.M. and Stonham T.J. Coding Facial Images for Database Retrieval using a Self Organising Neural Network, *IEE Colloquim on 'Machine Storage and Recognition of Faces',* January 1992, 3/1-3/4.

Robertson G.J.S. Image Processing at the Hairdressers, *Strathclyde University,* 1989.

Robertson G.J.S. and Sharman K.C. Object Location Using Proportions of the Direction of Intensity Gradient (PRODIGY), *Vision Interface 92', Canadian Image Processing and Pattern Society,* 1992, 189-195.

Robertson, G.J.S. and Sharman K. POLGA - The persistence of Life Genetic Algorithm and its parallel implementation on transputers, *Workshop on Neural Networks, Genetic Algorithms and Simulated Annealing,* 1990.

Chellappa R. and Rosenfeld A. Computer Vision: Attitudes, Barriers, Counseling, *Vision Interface '92,* May 1992, 1-7.

Rvdfalk M. Candide, a Parameterised Face, *Linkoping University, Internal Report,* 1987.

Samal A. and Iyengar P. A. Automatic Recognition and Analysis of Human Faces and Facial Expressions: A Survey, *Pattern Recognition,* 1992, 25:1, 65-77.

Schalkoff R. J. Digital Image Processing and Computer Vision, *New York, John Wiley and Sons,* 1989.

Seeling G.C. Tracking 3-D Moving Objects, *Philips research labs U.k. internal report,* September 1990.

Sension. IPLIB - Transputer Processing Library, *Denton Drive, Northwich, Chesire CW9 7LU,* 1990.

Serra J. Image Analysis and Mathematical Morphology, *NY Academic Press,* 1982.

Shackleton M.A. and Welsh W.J. Classification of facial features for recognition, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVIP-91),* 1991, 573-579.

Sheperd J.W. An Interactive Computer System for Retrieving Faces, *Aspects of Face Processing, NATO ASI Series,* 1986, 398-409.

Sleigh A.C. The extraction of boundaries using local measures driven by rules, *Pattern Recognition Letters,* September 1986, 4, 247-258.

Somerville I. Information Unlimited, *London: Addison-Wesley,* 1983, 4, .

Stonham J. Practical face recognition and verification with WISARD, *Aspects of Face Processing,* 1986.

Sutherland K., Renshaw D. and Denyer P.B. A Novel Automatic Face Recognition Algorithm Employing Vector Quantization, *Colloquim on 'Machine Storage and Recognition of Faces', IEE,* January 1992, 4/1-4/4.

Tock D. Findface: Finding Facial Features by Computer, *Aberdeen University, PHD Thesis,* 1992.

Trew T.I.P, Gallery R.D., Thanassas D. and Badique E. Automatic Face Location to Enhance Videophone Picture Quality, *British Machine Vision Conference,* 1992.

Tubbs, J.D. A note on Binary Template Matching, *Pattern Recognition,* 1989, 359-365.

Turk M. and Pentland A. Eigenfaces for Recognition, *Journal of Cognitive Neuroscience,* 1991, 1, 71-86.

Waite J.B. and Welsh W.J. An Application of Active Contour Models to Head Boundary Location, *British Machine Vision Conference,* 1990, 3, 407-412.

Webster R.W., Wei Y. A Robot Golfing System Using Binocular Stereo Vision, *Vision Interface 92',* May 1992, 195-202.

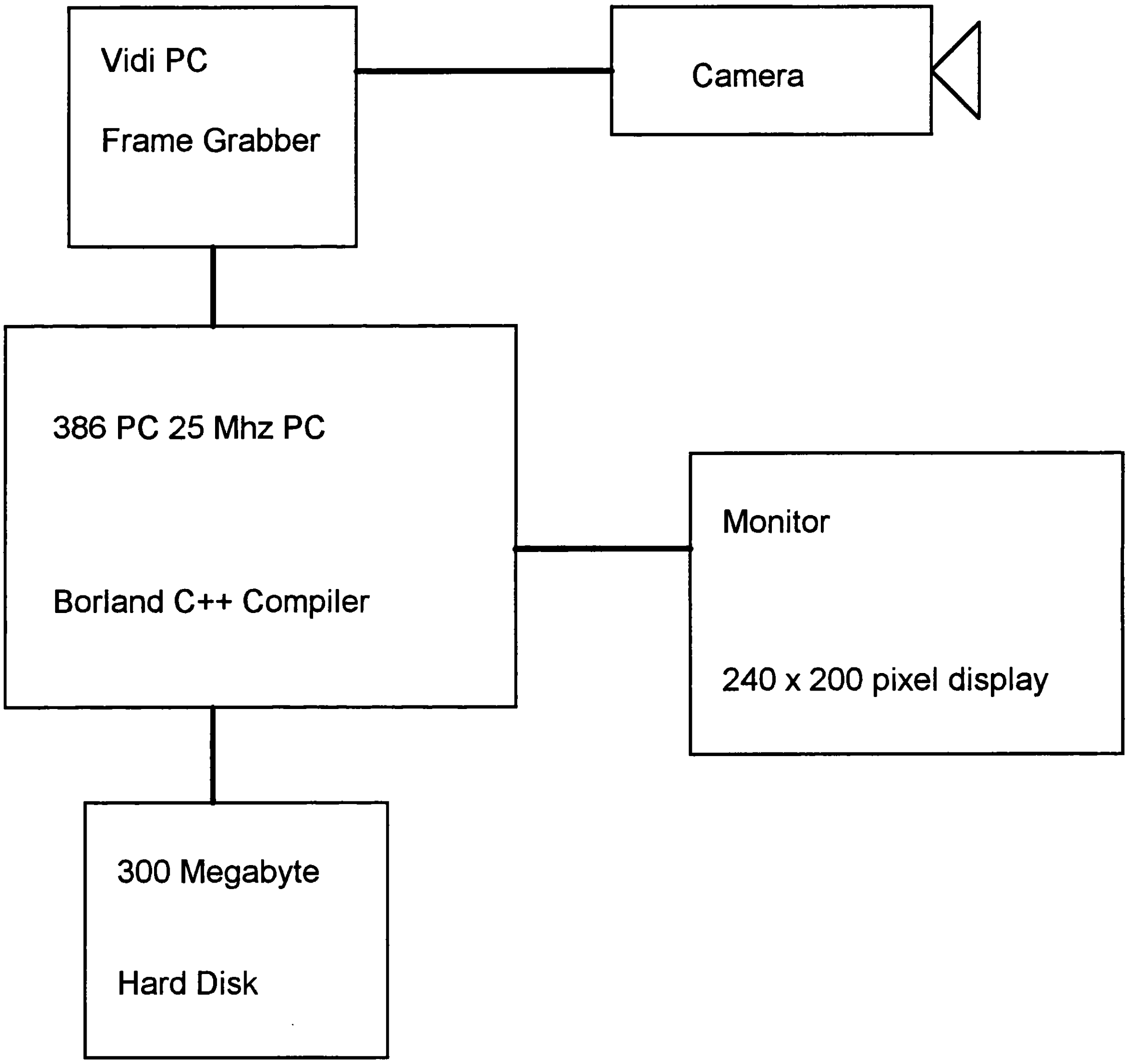
Wilson K. Vidi-PC Manual, *Rombo Productions, 6 Fairbairn Road, Kirkton North, Livingston, Scotland,* March 1990.

Yuille A., Cohen D. and Hallinan P. Facial Feature Extraction by Deformable Templates, *Technical Report CICS-P-124, Centre for Intelligent Control Systems, Cambridge, MA 02139,* March 1988.

Zhu Q. and Poh L.K. A Transformation-Invariant Recursive Subdivision for Shape Analysis, *9th International Conference on Pattern Recognition,* 1983, 833-835.

8. Appendix

8.1 Equipment Set Up



8.1.1 Computing Equipment

The computer made available consisted of:

- A 80386 based IBM PC compatible computer running at 20Mhz.

- A 14 inch colour monitor capable of displaying images in 64 gray levels to a resolution of 320x200 pixels.
- A 300 Megabyte hard disk.

As the PC can only display images up to a size of 320x200 pixels, which is a fairly low resolution, it was decided that all the images used in this research would be 240x200 pixels[27]. A 240x200 pixel picture, although an awkward size to process, gave a square image on the screen. The architecture of the IBM PC and the compiler also limited the image size because only arrays of up 64Kbytes were possible. A 240x200 pixel image with 8 bits per pixel is 48000 bytes in size, which fits within these requirements. Because the size of these images is small the computer could store many of them on the hard disk.

8.1.2 Imaging hardware

The imaging hardware available for this research consists of:

- Black and white camera with a composite video output, focus and aperture.
- Green monitor to view the image from the camera.
- Vidi-PC frame grabber (Wilson, 1990). The frame grabber can grab images up to 1024x512 pixels with 16 grey levels.

The Vidi-PC frame grabber connected to the PC was simple one that only grabs images in 16 grey levels, however, an algorithm scales the 1024x512 images to 240x200 images and simulates more grey levels by averaging pixel intensities[28]. The overall imaging hardware was of poor quality but suffices for much of the research.

Although the frame grabber can grab images at video rates the accompanying software is slow and does not transfer the images to the computer at video rates. The actual frame rate is about 2 frames per sec.

[27] The images consist of 256 grey levels (8 bits/pixel) even though the computer could only display 64 grey level.

[28] The image was first cropped to 960x400 pixels. Each block of four pixels was averaged to produce a 240x200 pixel image with 64 grey levels.

8.2 Face bank created for the research

The face bank is a database of faces selected from the general public. The bank includes a range of people as shown in table 16.

Name of face	Sex	Comment
Joy	Female	Poor quality
Morag	Female	Poor contrast
Tracy	Female	
Alas	Male	
Musty	Male	
Jim	Male	
Deep	Male	Dark skinned
Kate	Female	
Thomas	Male	
Danny	Male	
Tony	Male	
Sheila	Female	
Steve	Male	
Andy	Male	

Table 16: *List of faces in the facebank*

There are eighteen pictures of each subject in the face bank. Each picture was stored with a a name and number.

e.g. JOY1, JOY2,..., JOY18, MORAG1, etc.

On each face image 37 landmarks were located. Table 17 lists the chosen landmarks.

1	nose:	left of bridge
2		middle of bridge
3		right of bridge
4		bottom left
5		bottom middle
6		end of nose
7		bottom right
8	mouth:	left
9		top lip, top middle
10		top lip, bottom middle
11		bottom lip, top middle
12		bottom lip, bottom middle
13		right
14	left eyebrow:	left
15		top middle
16		bottom middle
17		right
18	left eye:	left
19		left of iris
20		top of iris
21		middle of iris
22		bottom of iris
23		right of iris
24		right
25	right eyebrow:	left
26		top middle
27		bottom middle
28		right
29	right eye:	left
30		left of iris
31		top of iris
32		middle of iris
33		bottom of iris
34		right of iris
35		right
36	head:	left at mouth level
37		right at mouth level

Table 17: *The 37 points manually located for each image in the face bank*

The 37 points were marked by hand with a mouse pointing device and the location recorded in a file (See example in section 8.3). Where the landmarks were occluded due to rotation or closed eyes, then they were approximated. For the TINY and MINUTE images it is difficult to resolve all the landmarks around the eyes so they were just positioned roughly.

8.3 GFF image descriptor

To enable the exchange of images between various graphics platforms a graphics file descriptor (GFF) was designed, which encompasses all the present graphics file formats currently available. GFF primitives create a description file for each image in which is stored the history of the image. The design of the GFF image loading and saving primitives was such that they could determine the graphics format of the image file and save or load in that format. The GFF primitives also updates the description file if any changes are made to the image.

Initially all the images attached to the GFF descriptors were GIF files[29]. The GIF primitives compress the images while saving and loading. Although this compression saved considerable space the code to perform this compression was slow. Therefore, later on in the research, the face processing algorithms worked with RAW[30] image files, which despite taking up more space, load and save quickly[31]!

The follow text is the sample contents of a GFF image file descriptor. A face data descriptor is also attached to all the image files in the face database.

```
#%Date of creation of GFF file image descriptor
17/03/91
#Last modification to image - 8 character code
Original
#60 Characters of description      ----> up to this bracket]
Joy 4
#One line description
```

[29] GIF stands for Graphics Interchange Format. The C software routines to handle these files are public domain (Elber, 1989)(Compuserve, 1987).

[30] RAW image files contain only the image data with no header files. To read a RAW image file the computer must know the size of the image. A 240x200 pixel image with 8 bit/pixel is exactly 48000 bytes in size.

[31] In about 0.25 secs as averse to 1.5 secs

```
#Brief three line description
#Description of the image - maximum 6 lines
#History of the image after creation
#Comments about the image
@@End of GFF file image descriptor
{ facedata
nose: left of bridge
120 75
nose: middle of bridge
124 73
nose: right of bridge
127 74
nose: bottom left
117 92
nose: bottom middle
125 94
nose: end of nose
125 91
nose: bottom right
133 91
mouth: left
116 104
mouth: top lip, top middle
125 101
mouth: top lip, bottom middle
125 103
mouth: bottom lip, top middle
125 104
mouth: bottom lip, bottom middle
125 107
mouth: right
137 104
left eyebrow: left
102 69
left eyebrow: top middle
110 66
left eyebrow: bottom middle
110 69
left eyebrow: right
118 67
left eye: left
104 77
left eye: left of iris
108 75
left eye: top of iris
111 72
left eye: middle of iris
111 74
left eye: bottom of iris
111 76
left eye: right of iris
113 74
left eye: right
117 74
right eyebrow: left
129 67
right eyebrow: top middle
135 65
right eyebrow: bottom middle
136 68
right eyebrow: right
145 67
right eye: left
131 74
right eye: left of iris
135 74
right eye: top of iris
137 72
right eye: middle of iris
138 74
right eye: bottom of iris
138 76
right eye: right of iris
140 74
right eye: right
145 75
head: left at mouth level
104 104
head: right at mouth level
152 103
}
```

8.4 Paper 1

This paper was presented at a Workshop called Neural Networks, Genetic Algorithms and Simulated Annealing, Glasgow 1990.