University of Edinburgh
Division of Informatics

Gesture Recognition Using Human Skin Classification, Zernike Moments as Image Descriptors and HMMs to Model Gestures

4th Year Project Report
Artificial Intelligence and Computer Science

Ebenezer Ademisoye

March 2, 2005

Abstract: The deaf and hard of hearing account for 8.6 million people in the U.K. This coupled with the fact that the mouse and the keyboard are not natural input methods for humans encourages us to look for better interfaces for everybody and more specifically for the hard of hearing. A system that could use gesture recognition as an interface would be more natural. Sign language exists in various forms and is a valid choice for human computer interaction because of its already rich vocabulary, syntax and grammar. Furthermore, there are already many who use sign language such as the estimated 500 000 users in America and canada where American Sign language (ASL) is the third most used language in the US. Such a system would already have a number of users who need little training to use it. Sign language is much more than gesturing and so a sign language recognition system could have greater significance that just a computer interface. Such a system could allow fuller integration of the hard of hearing into the hearing community as it would break down the language barrier or at least ease communication between the sub culture and the hearing community. This project will be based on gesture recognition using a colour video camera to capture human beings with little restriction on the background and clothing. Zernike moments are used for image description and classification of the 10 poses with 94.6% accuracy. Hidden Markov Models are used to gesture recognition with a rate of 84% accuracy for 10 poses and 93% accuracy when modified data is used.

Contents Page

1. Introduction

A number of countries have their own sign language. American Sign Language (ASL) is the most popular with an estimated 500 000 members of the signing community in America and Canada. However, the hearing community and the signing community are still distinct in day to day life. Tasks that are trivial for the hearing community, such as passing through metal detectors at airports, can be difficult for the hard of hearing if the airport staff have no training or familiarity in signing. Any difficulty in communication can be resolved by having a translator accompany the hard of hearing during an excursion. However, this is impractical for 3 reasons:

1. It may be inappropriate to have a third person in a private or sensitive meeting such as an appointment with a medical doctor or consultation with a criminal lawyer.
2. The cost of having an accompanying translator on every public appearance would be too great.
3. Translators are in high demand and one may have to arrange an appointment two months in advance just to be guaranteed the translator's services.

However, an automatic sign language recognition and translator system could solve these problems. The user could own the system and so any data sent or received by the system would be under the control of the user and so confidentiality can be controlled. Also, widespread use of the system could mean a reasonably priced system. However, the final problem of mobility is more difficult to solve. There exist mobile speech recognition systems that can accurate extract the user's voice from all the background noise. A mobile sign language recognition system would be similar as it would have to extract the user's movements from all the information available in the 3D environment.

This project presents a mobile gesture recognition system with a vocabulary of 4 poses and 10 gestures described by a sequence of the poses. The system is able to extract the user from the background by selecting the pixels human skin pixels with minimal constraints on the background or the user's clothing to make the system very portable. This is very different to the majority of previous gesture language recognition systems that require signers to have markers placed on specific parts of their body and have highly constrained backgrounds. Zernike moments and Gaussian distribution are used to classify each pose. This is also a different approach to past projects that have relied heavily on tracking to recognise the state of the gesture from one time step to another. Finally, Hidden Markov Models are used to recognise the sequence of poses.

There are 4 main sections to the system:

1. Capturing the sign.
2. Background removal.
3. Pose recognition.
4. Gesture recognition.

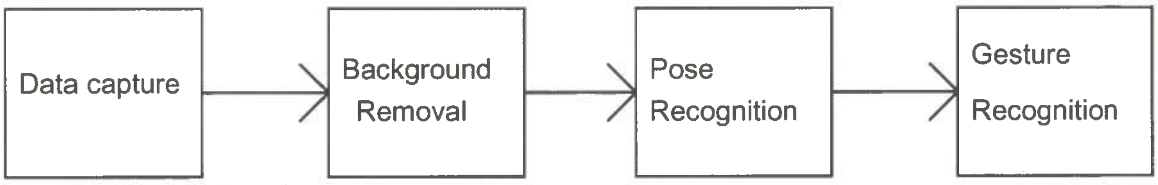
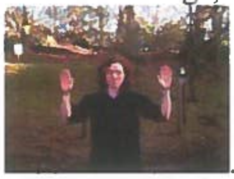
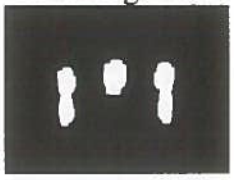


Figure 1.A. Shows the ordering of the 4 major subsystems.

1. In the first stage, a gesture is captured as a sequence of frames.
   . This is discussed in chapter 2.
2. The background is removed from the frame to produce a binary image
    This is discussed in chapter 2.
3. The binary image is then classified to produce a label '2'. This is discussed in chapter 3.
4. The sequence of labels e.g. '121', are classified as a gesture, e.g. 'gesture 1'. This is discussed in chapter 4.

1.2 Background

1.2.1 Related work

There have been previous successful attempts at recognising human skin pixels from images. There are a number of colour spaces where the skin pixels can be modelled but [1] concluded that the HSV space performed better than all the 5 that were compared. This was also reported by [6] in their work on automatic face recognition. It has been found [10, 4] that human skin pixels, even those from people from different races, lie in compact region of the colour space. There are a number of tools used to model the cluster of skin pixels such the simple Gaussian, a variation of the Gaussian, or a histogram. [6] However, [8] have demonstrated that the histogram model is superior to the Gaussian models. This model is used by [6] and [7] who also implement a face recognition system.

To create a histogram for the training data, the hue and saturation hyper planes are each portioned into a number, n, of equal portions. This is necessary as the hue and saturation hyper planes range from 0 to 1 inclusive. The greater the number of divisions, the greater the accuracy that can be obtained from the model. N is typically 100. For each pixel in the training data, the hue and saturation is extracted and the bin corresponding to that hue and saturation value is incremented by 1. The histogram is then normalised and viewed to give the regions where human skin pixels most frequently appear. The height of the bin is proportional to the probability that the colour encountered is a human skin pixel. To classify a colour as human skin, its corresponding bin is located to produce the probability. The probability is then used to classify the colour as skin if the probability is greater than a certain threshold. If the probability is not, then the skin is classified as background. If the probability is greater than the threshold, then the colour is classified as human skin. This is threshold along with the histogram are commonly referred to as the skin filter.

Often, there will be two aspects to classifying human skin. The skin filter is the first and edge information is used for the second stage. An edge map of the image is obtained using some well tested edge detecting algorithm such as Sobel, which is used by [6] and [7]. The edge at each pixel to be classified is then compared with an edge threshold that has been obtained empirically, much in the same way as the histogram was obtained. The edge at the pixel has to be smaller than the edge threshold to be classified as human skin. To be classified as human skin in the final binary image, a pixel has to satisfy both tests.

1.2.1 Data capture

There have been several attempts to classify gestures automatically. The most promising have been the sign language recognition systems. These systems can be divided into two sets based on the data devices used to capture the movements. The first set uses video cameras to capture the different movements. The other set use motion capture devices commonly known as data gloves.

1.2.2 Image

Traditionally, fixed video cameras have been used to capture the relevant details of an image. Markers are placed on the body to help better distinguish between the signer and the background. [2] uses coloured markers to better distinguish between dominant and non-dominant parts of the hand. An algorithm that detects the colours of these specific markers is used for automatic background removal. There are restrictions regarding the signer's clothing and the background must be of a uniform colour in order for the background removal algorithm to work correctly. The entire system, including the background removal and feature classification works at a rate of 13 frames per second on a Pentium III 300 computer. Such a system does not use information about the user's head which is important in sign language classification.

The system built by [12] allowed direct recognition of the head by detecting human skin pixels and the resulting shape of classified regions were considered before any region was classified as the head. However, the use of coloured gloves was retained. The background removal then worked in a similar fashion to [2]. [12] system allowed greater flexibility in the background as the location of the classified head and the gloves were considered after the initial classification stage, so that noisy regions could be eliminated. This worked particular well as it meant that other people could be visible in the colour image and as long as they were a reasonable distance from the centre of the image, the person would be classified as noise. Features such as pose and orientation are used to describe each frame. This systems works at 15 frames per second.

A more ambitious system with the camera being mounted onto a hat that the signer wears is successfully implemented by [5]. Here, the captured image is scanned from left to right until a pixel is found that has a colour that falls within the range of human skin colour.

The 8 neighbours of this pixel are then checked to see if they also are human skin pixels and this continues recursively. Although not implemented, the system is able to use the nose to automatically calibrate the camera in the case of changing background light. Regions are grown from the recursive process. The labels "right hand" and "left hand" are assigned to each region corresponding to the relative position of the two major regions in the image. Unfortunately, one hand often occludes another and so there may not be two distinct regions but one large region with a great mass. This is especially bad since the system tracks the individual hands and occluded hands are difficult to track. The information obtained from tracking is used for the feature vector. This information includes the x and y position of each hand, the change of the x and y position of each hand from one frame to the next, the mass of each hand, the eccentricity of the bounding eclipse and other information obtained using the PCA tool. When a hand is occluded, this is even recorded and so is modelled both in the training data and test data. This system works at 10 frames a second and achieves at 96% classification rates on an independent set.

1.2.3 Data Gloves
Data glove is the generic name for the motion capture device that is worn as a glove and provides real-time information on the various angles of the joints which can be used to calculate more information. [9] used data gloves to facilitate sign language recognition. The data glove uses optical fibres to detect joint angles and magnetic sensors to determine position of the hand. Data obtained about the hands were sampled 30 times a second and had to be compressed because this was deemed too much information. The system has a recognition rate of 100% among 17 words when the uncompressed data is used but it takes 1.23 seconds to recognise each word when used on a HP9000/720 UNIX workstation. The recognition rate drops to 97.3% when the compressed data is used and the system only requires 0.076 seconds per word.

[3] used data gloves and position trackers to obtain six features: right hand shape, right hand position, right hand orientation, left hand shape, left had position, left hand orientation. This system was tested with a vocabulary of 5177 signs from Chinese sign language. The system can recognise up to 95.1% isolated word recognition and 91.7% word recognition for continuous sign recognition. However, the system was trained by a Chinese sign language teacher where 5 words of each sign were used for training and another instance of sign performed by the very same teacher was used to test the performance of the system.

1.3 Sequence Classification Models
1.3.1 Introduction
Past systems for accurate continuous sign language recognition first recognised isolated words to form a sequence of labels and then classified the sequence. There have been two major approaches to modelling a sentence as a sequence of individual signs. The most popular model is the Hidden Markov Model; and the other is the neural network

1.3.2 HMM

[12] used linear forward Hidden Markov Models to train a German sign language recognition system with a vocabulary of 16 signs and achieves 94% accuracy. Only 7 examples of each sign were used for training the models and another 3 were used for testing. [2] used Hidden Markov Models to train a signer dependent German sign language recognition with a vocabulary of 52 signs to achieve 95% accuracy. 3.5 hours of video were used to train the models and a further 0.5 hours were used to test the models.

1.3.3. Neural Network

[5] initially trained the system to continuously recognise a finger alphabet using a three-layered recurrent neural network that uses the back propagation. There are 16 nodes in the input layer, 150 nodes in the hidden layer, and 10 nodes in the output layer. This system took 4 days to train on 10 words with a SUN/4 workstation with a learning rate of 0.01 and momentum of 0.05 and achieved up to 96% accuracy. [13] also used a neural network to train an individual word recogniser with 99% accuracy with a vocabulary of 203 words and adaptive control of speaking rate and word stress is available.

2 Background removal

2.1 Aim

Given a frame from a video, the background removal subsystem attempts to accurately create a black and white image, where the white pixels are the human's face, hand, and forearm region; the black pixels represent the background. Figure 2.A is a typical frame from a video and Figure 2.B shows what the image should look like after background removal.

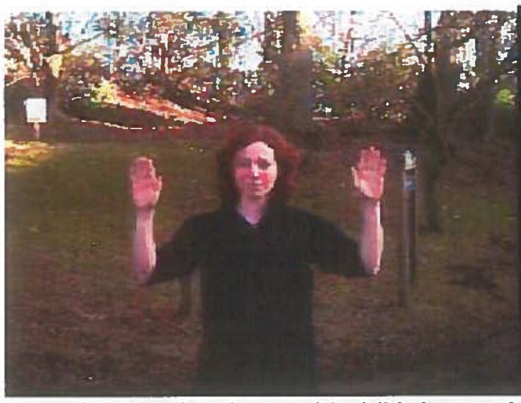


Figure 2.1.A a colour image with visible human skin

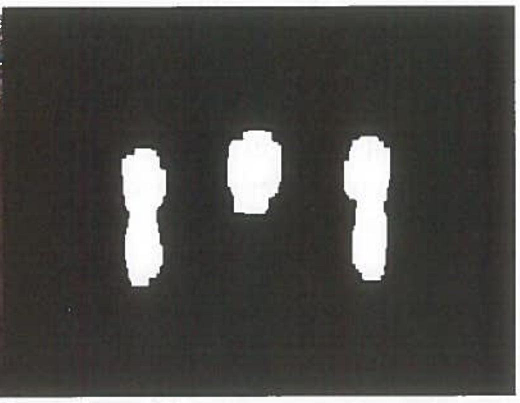


Figure 2.1.B

There are 2 major stages involved in background removal:

1. Identify the human skin region.
2. Remove any remaining noise.

2.2 Modelling human skin pixels

2.2.1 Modelling difficulties

Differentiating between human skin and the general background is a difficult task given that the appearance of skin is influenced by many factors such as ambient light and object movement. Also, different cameras produce different colour values even for the same person under the same lighting condition and skin colours differ from person to person. A model that resolves those problems is possible in the Red, Green, and Blue (RGB) space but this space is very sensitive to illumination intensity. The three values in the RGB change given an illumination change. However, the Hue, Saturation, and Value (HSV) space exists where only the variable Value will change given an illumination change.

2.2.2 Red Green Blue space

All colours used in the RGB can be specified in terms of the intensities of the three primary colours - Red, Green and Blue [16]. Each colour is represented as a three element vector [R, G, B]. The intensity of the colour is given by the length of the vector and the colour is given by 2 angles describing the orientation of the vector in the RGB colour space [16]. It is clear to see that if the illumination changes, then the intensity changes and so the length of the vector used to describe the colour in RGB space must change.

2.2.3 Hue Saturation Value

The HSV space is also a three dimensional space described by a vector [H, S, V]. The hue component specifies the average wavelength; the saturation specifies the amount of white light; and the value component specifies the brightness or the illumination [16].

The HSV space is circular. This is easy to show if one considers that 0 =H =1 and H = mod(H + 1, mod 1).

2.2.4 Building the skin classification model

2.2.4.1 Collecting data samples

Quality photographs were taken of humans in good light. The humans would change their pose from one photograph to another. This was done so that any shadows that formed on the humans should change from one photograph to another. Varying the shadows would mean any machine learning technique that modelled the human skin would not fit the noise created by the shadows. Also, the venue where the photographs were taken was varied in order to further minimise any noise. Below follows 4 of the 43 photographs collected to extract skin samples.

These are labelled Figure **2.2.4.A-2.2.4.H:** These are digital photographs at a 1MegaPixel resolution and in the format JPEG.

The image editing program GIMP was used to select the skin regions of the photographs. Skin samples from the head region, the hand region, and the forearm region were collected. The background removal subsystem had the goal of selecting these regions given a frame from a video. Therefore, only skin samples from these regions were used to model the human skin. Collectively, 86000 individual pixels, corresponding to human skin, were collected. Furthermore, only 80000 were distinct pixels. These pixels were then mapped from the RGB space to the HSV space. Given that an HSV pixel is 3-dimensional with the first dimension corresponding to the hue, the second to the saturation, and the third to the value, this data could be visualised in 3D.Initially there are two distinct clusters. Section **2.2.3** shows how the HSV space is circular. However, the natural 3D space is not circular. Any machine learning tool would perform significantly better if the 2 distinct clumps where just one large clump. The 2 distinct clusters are made one cluster by modifying the hue component of the vector representing each data point. That is, H=mod(H*256 + 128,256)

Figure **2.2.4.J** below shows how the data points appear after the transformation.

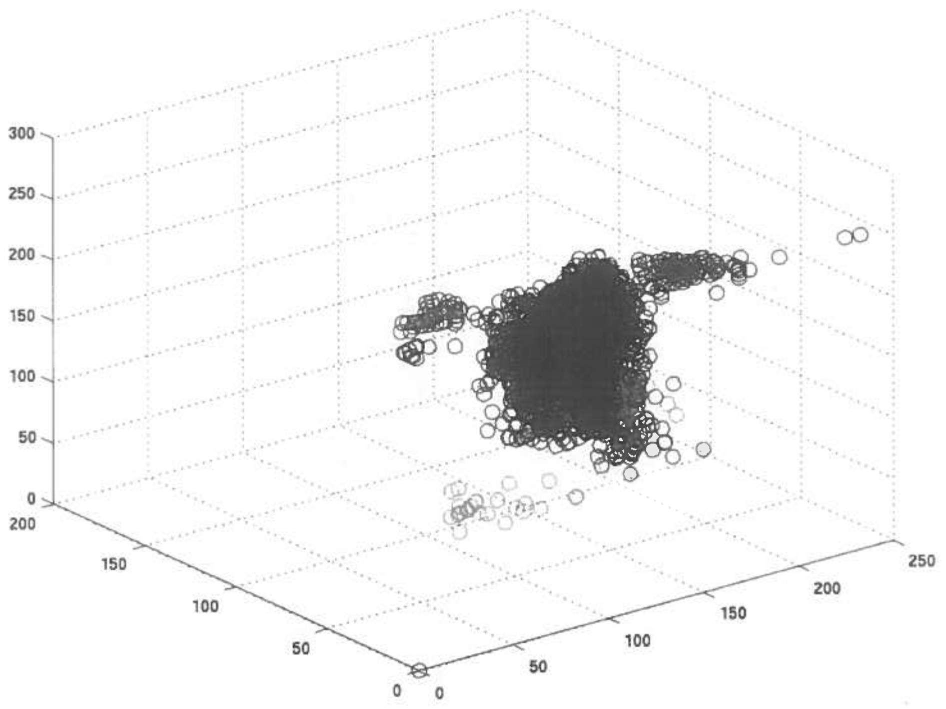


Figure **2.2.4.J.** The dark blue region corresponds to pixels in HSV space where the hue has been modified to make the clusters corresponding to human skin pixels merge into one cluster. The saturation component has been modified so that it can be viewed over the same range as the hue and value axis. The red region corresponds to data points that were obtained as noise from multiple images after the first set of images were processed by the background removal system. The light blue region corresponds to data points that were obtained as noise from multiple images after the second set of images were processed by the background removal system.

The data has now been viewed and found to be in a state where a machine learning technique can be applied to it to produce a solution. A solution would take a pixel and put it into one of two sets. Let the two sets be labelled A and B for simplicity sake. Let set A contain pixels of human skin and just that. Let set B contain pixels of the background. If the solution considered each pixel from an image, then it should be fast. It should be fast as images typically contain many pixels. Slow but accurate solutions are to be avoided as they make this sub systems infeasible considering there will be many frames for each video and there will be many videos. If the solution is slow for one frame, then one would have to wait a large amount of time for the many videos to be classified. Also, as will be shown over the following subsections, a fast and fairly accurate solution may achieve a comparable rate of correct frame classification in a fraction of the time. Therefore, a fast but simple classifying system is used here.

Given the data as presented in Figure **2.2.4.J.** straight edges where drawn to encapsulate much of the human skin data points. This is shown in Figure 2.2.4.K. and Figure **2.2.4.L**

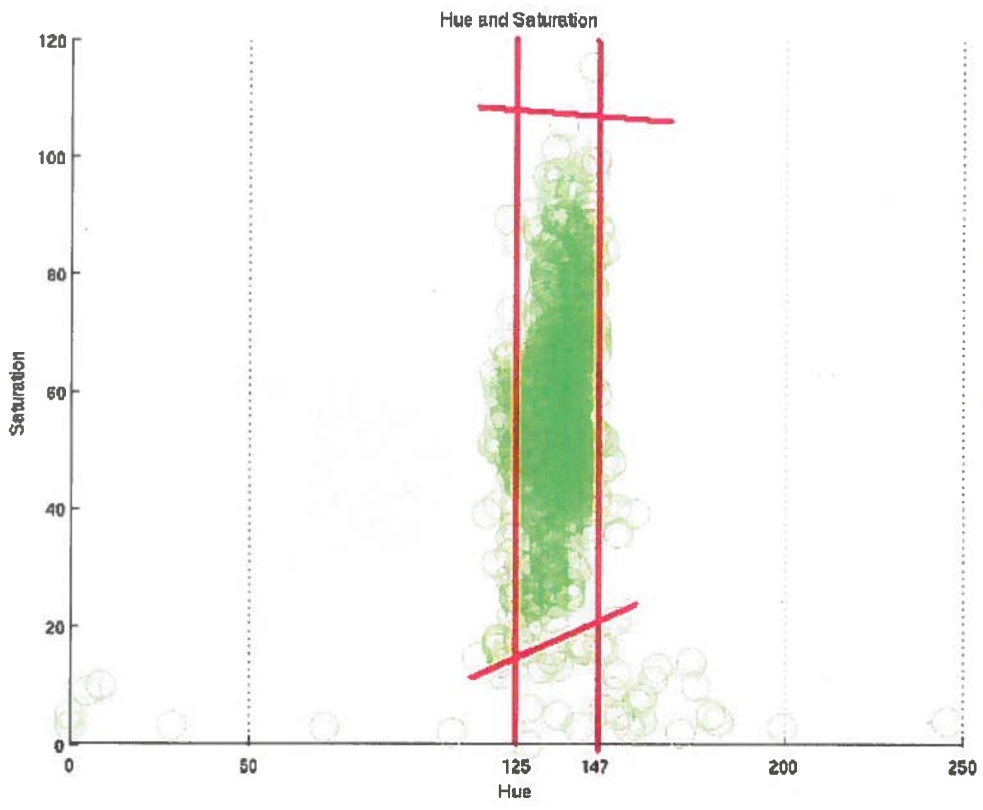


Figure 2.2.4.k The hue and saturation space.

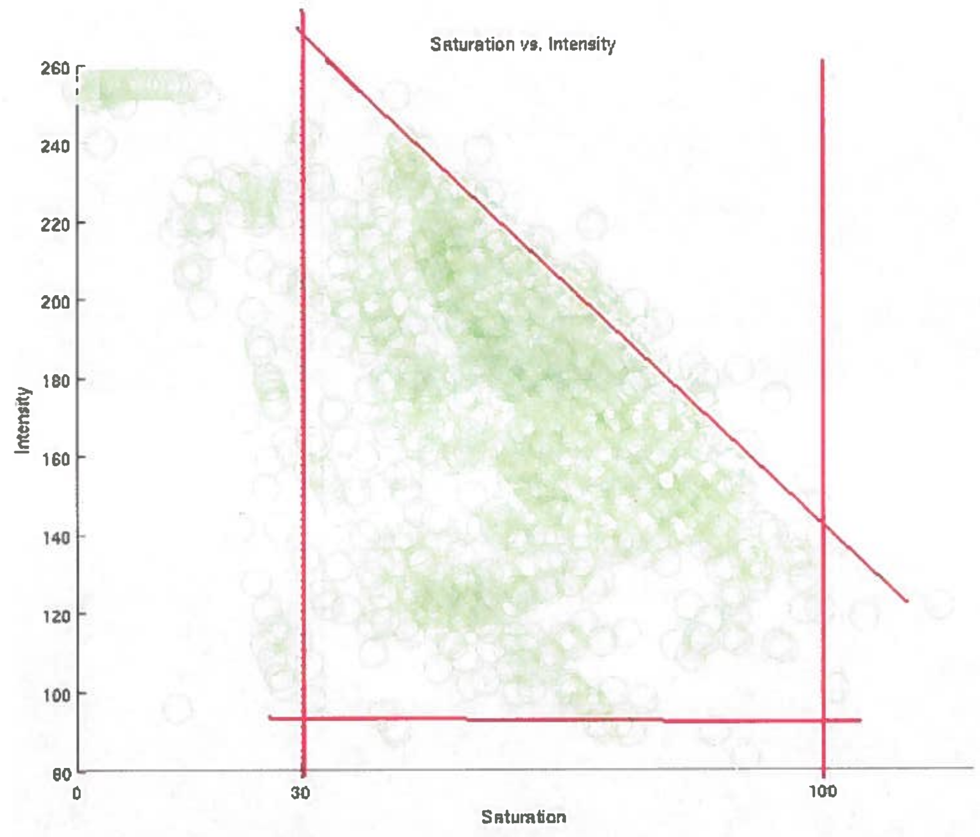


Figure 2.2.4.m The saturation and intensity space.

The classification solution is a simple one. If the novel pixel lies within the lines drawn, then classify the pixel as human skin; that is, state that the pixel belongs in set A. If the pixel does not lie within the lines drawn, then classify the pixel as background; that is, state that the pixel belongs in set B.

2.2.4.2 Results

At this stage of a presentation on a classification solution, a confusion matrix of results would be provide. This matrix would summarise the performance of the solution. However, since the solution given by the classification system is an image, displaying typical examples of the images obtained would seem like a more natural way to show the performance of the system. Furthermore, one such criteria often used in the typical confusion matrices used would be number of test points correctly classified. However, for this specific task, such a measure is difficult to obtain given the number of pixels in a typical image.

Figure 2.2.4.2.A. Sample of the classification system performing very well. Most of the human skin pixels were correctly classified and there are a small number of background pixels incorrectly classified.

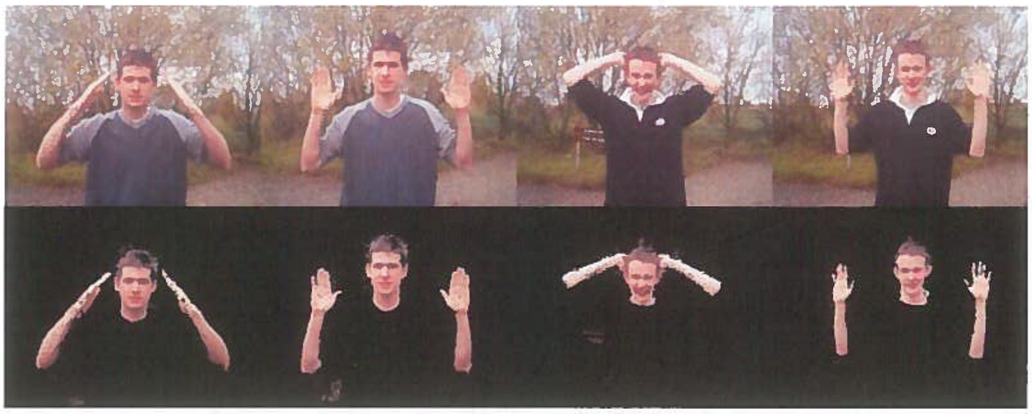
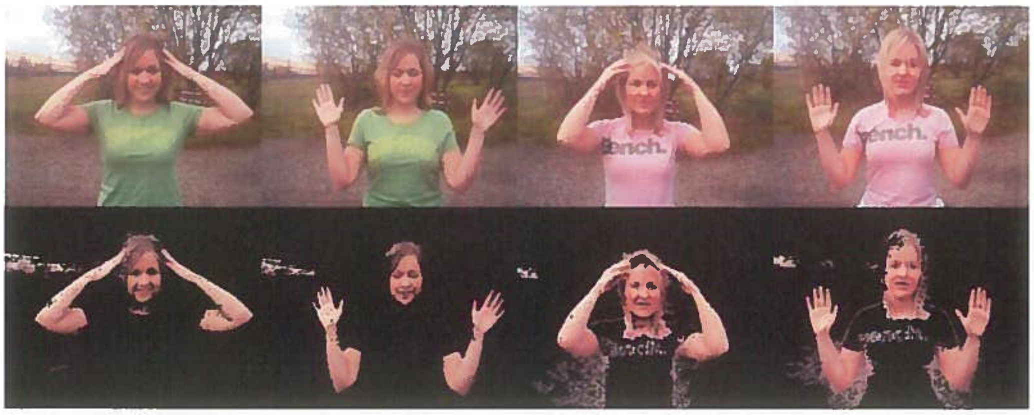


Figure 2.2.4.2.B **Sample of the classification system retaining noise. Most of the human skin pixels were correctly classified and sections of the background pixels are incorrectly classified.**



The typical classified image tends to be more like those found in Figure 2.2.4.2.B. Out of 3940 images that were classified for human skin pixels, 40 used in testing and another 3900 obtained from videos of gestures, 2989 contained noisy regions. This means 76% of all images contained noisy regions. As will be shown in section 2.3, the vast majority of noisy regions can be removed.

2.2.4.2 Analysis

2.2.4.2.1 Explanation for performance

As was shown from the results above, the process of identifying human skin pixels from an image often results in noise being identified as well. This is to be expected for three main reasons:

1. Image capture problems.
2. The classification rule is crude.
3. The presence of pixels that are very similar to human skin pixels in nature.

2.2.4.2.2 Image capture problems

Often, images will have features that are natural but inhibit their suitability for classification problems. Specularities are a problem for this recognition task. When a light source emits light and it hits a diffuse surface, the diffuse surface reflects light that is coloured by both the source colour and the surface colour. Natural sunlight appears as white light. When too much light is reflected from the skin, then the entire colour reflected from the particular region of the skin becomes saturated and appear as white in the image.

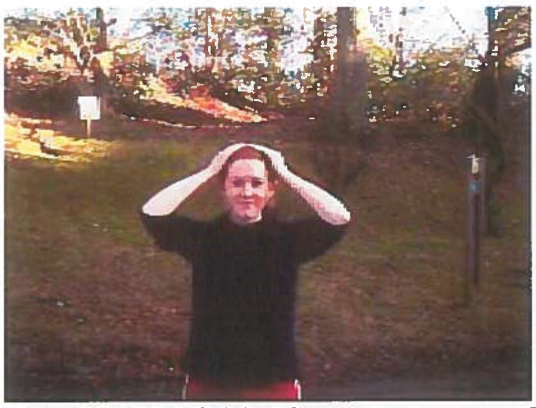


Figure 2.2.4.2.2.A Original frame

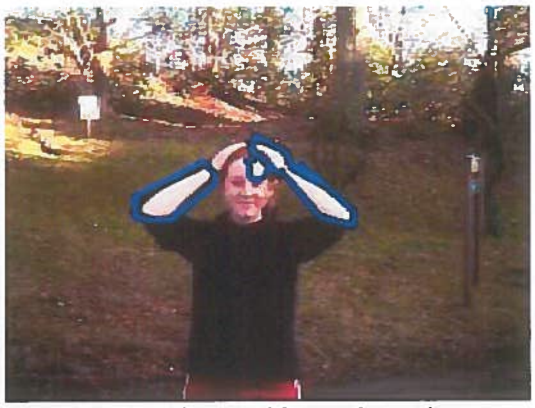


Figure 2.2.4.2.2.B Frame with specular regions highlighted.

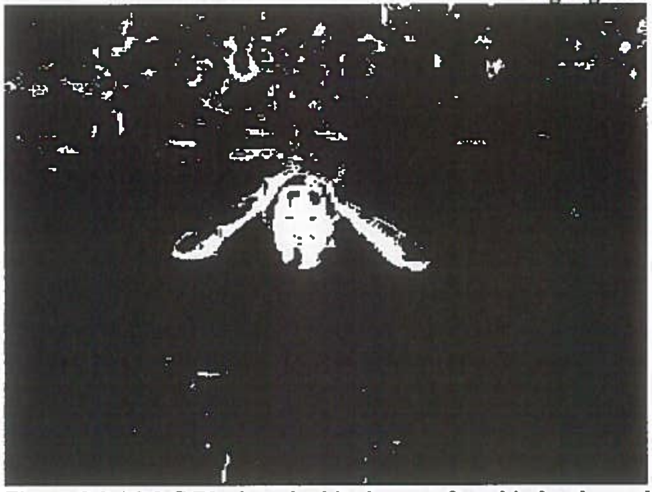


Figure 2.2.4.2.2.C Black and white image after skin has been classified

The classification rule is such, that it will reject pixels that are white. Therefore, human skin pixels altered to appear white will be rejected. Figure 2.2.4.2.2.B, shows regions of skin that have been altered. As can be seen in Figure 2.2.4.2.2.C, the pixels highlighted in 2.2.4.2.2.B have primarily been rejected by the classifying rule. As can been seen, this is a major problem and the main reason regions of skin will be rejected as background in an image.

2.2.4.2.3 Crude classification rule

Section **2.2.2.4.1** explains the need for a simple classification rule. However, a slightly more complicated rule that examines a number of pixels at one time may perform better both in time and classification rate. Such a rule may consider a square neighbourhood of pixels for example. The rule could find the mean of that neighbourhood and base classification of the entire neighbourhood on this mean. This small addition to the original classification rule may reduce the effects of many image capture issues such as the one discussed in Section 2.2.4.2.2. This new rule is just one of the many that may be tested to improve classification. It is worth noting that other systems that do employ more complicated models with a significantly greater number of training data still classify noise with the human skin [16]

2.2.4.2.4 Desired human skin in nature

The two criteria for a good classification rule for this task are that it is fast and has a reasonable classification rate. This may mean that any successful classification rule will only consider pixels individually or a group of pixels in some neighbourhood window. Regardless of what is being considered, let us refer to it as a data point. That is, if it is an individual pixel, let us refer to this pixel as a data point. If it is some group of pixels found in a neighbourhood, let us refer to this neighbourhood as a data point. If data points are only considered individually, then data points that lie within the constrained range and correspond to noise will be classified as if the data point were human skin that lies with the constrained space. Given the collective structure of all data points that are skin pixels and are classified as skin pixels, most pixels that are noise and classified as skin can be easily identified. Skin data points tend to be positioned adjacent other skin data points. Collectively, these data points form a curvy arm or a round face. However, noisy data points that are classified as human skin will be scattered randomly forming no discernable shape or pattern. More complex rules can exploit the position of a data point in relation to the other data points to decide if it should be classified as a skin pixel.

These noisy data points will almost certainly exist in the vast majority of images classified using simple rules. This is because it is unreasonable to expect pixels that are similar to human skin pixels, to be absent in any natural environment. That is, one can expect to find human skin coloured pixels in the background of any natural environment. To illustrate this, consider this simple example. A photograph of a human face is taken using a reasonably priced digital camera. This photograph is then printed onto a t-shirt using a reasonably priced printer. If the pixels of a photograph of a person wearing this t-shirt were then to be classified into human skin and background, it is likely that some pixels corresponding to the printed t-shirt would be classified as human skin. Incidentally, the noisy data points on the t-shirt may still form a round face and so would elude even the complicated classification rules. The previous example essentially embedded what would be classified as human skin into the background. It is a crude example, but it illustrates that what would appear as human skin can often be found in the background. This partially accounts for the extra pixels that were classified as human skin when they were in fact noise.

2.3 Removal of Noise

2.3.1 Motivation

As will be explained further below, images are to be classified into 1 of 4 classes. This classification will be based on the orientation of the visible body parts. The skin classification procedure described previously obtains the visible human body parts from an image fairly well. Much of the human skin is correctly classified. However, noise is often classified as human skin. The noise in the classified images presents a problem for classification systems that rely on the skin classification procedure. Removing the noise would greatly aid these classification systems.

2.3.2.Aim

When the result of the classification is viewed as an image, it is an easy task for humans to discern what is noise and what is correctly classified; Figure 2.2.2.A below shows such an image. The noise is very distinct from the arms and head. As explained previously, much of this is due to the noise, collectively, not having a discernable shape or pattern. Techniques exploiting this could be used to also decide what is noise.

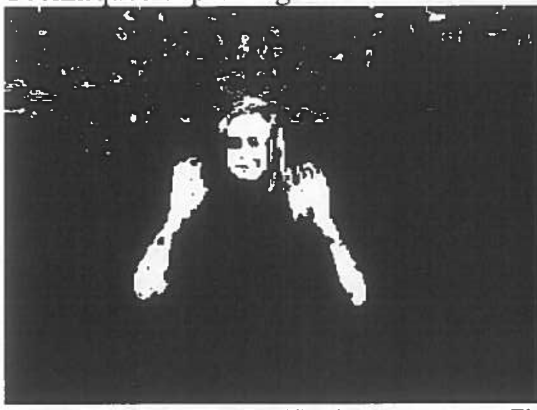


Figure 2.3.2.A. A skin classified image

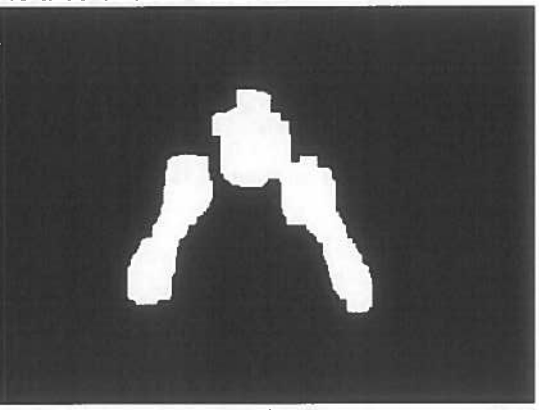


Figure 2.3.2.B. The cleaned image.

However, such techniques are avoided for two reasons:

1. Such methods would rely on knowing the various shapes that the arm and head can take.
2. Such methods would require analysing the patterns formed by noise.

Any further rules or techniques that solved both problems for each frame would most likely be very costly in time. Also, there appeared to be better methods of reducing noise in a skin classified image, without having to know much about each frame. Techniques which take an image such as Figure 2.3.2.A. and produce the image Figure 2.3.2.B. which is significantly cleaner. These methods are presented below and the results obtained are presented.

2.3.3 Noise removing strategies

2.3.3.1 The strategy

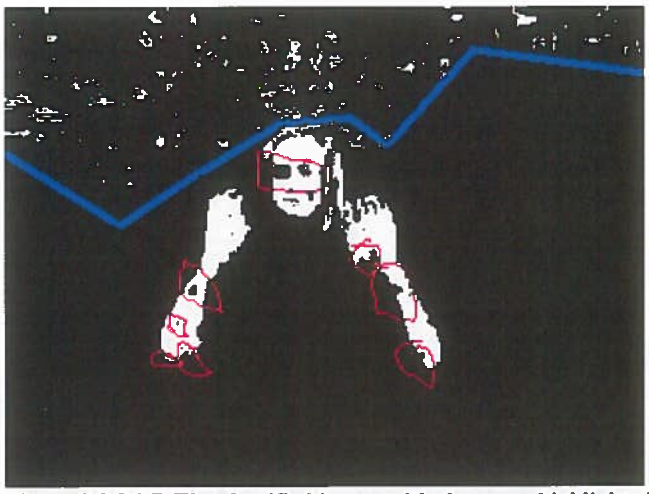


Figure 2.3.3.2.B The classified image with the error highlighted in blue and gaps in the skin highlighted in red.

The noise found in Figure 2.3.2.A is typical of noise produced by the skin classification procedure. Figure 2.3.2.2.B highlights the error in blue. The noise tends to be sparse which happens to be a very desirable feature. Also, the noise tends to be spatially separable from the object; this is also a desirable feature. However, consider the regions highlighted in red. These regions are gaps. All of the lower red regions are gaps that should have been classified as human skin but were incorrectly classified as background. The upper most red highlighted gap corresponds to what would have been the person's eyes. There are other regions on the face that were not highlighted that correspond to the person's lips and a dimple. Clearly, the eyes were correctly classified as background because people's eyes are very different to their skin. However, it would be desirable if the head region was void of any noticeable expression. The types of gestures that will be classified do not rely on any facial expression. Therefore, the gaps that correspond to the eyes, the lips, and the dimples are all to be filled to transform the gap filled head into a solid head shaped figure like that found in Figure 2.3.2.B. Therefore, this strategy attempts to remove the noise, whilst filling many of the gaps found within the skin and on the edge of the skin.

This strategy has five steps shown in Figure 2.3.2.2.C:

$$\begin{tikzcd}
\boxed{\text{Erode x1}} \arrow[r] & \boxed{\text{Dilate x3}} \arrow[r] & \boxed{\text{Erode x3}} \arrow[r] & \boxed{\begin{array}{l}\text{Keep 3}\\ \text{Largest connected}\\ \text{Regions}\end{array}} \arrow[r] & \boxed{\text{Dilate x1}}
\end{tikzcd}$$

Figure 2.3.2.2.C The five steps of the cleaning strategy.

2.3.3.2.2 Dilating and Eroding

Dilating and eroding are two techniques used to alter images on the pixel level. Dilating causes particular pixel regions to "grow" and eroding causes them to "shrink". An object is sometimes the black pixels and the background the white pixels in a black and white image. For the following examples, let the object be the black pixels as this simplifies the examples. Also, a pixel can only have one of 2 colour values. It can either be white or it can be black. Other colours are used for highlighting black pixels.

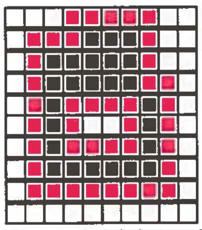


Figure 2.3.2.2.A Edge Pixels

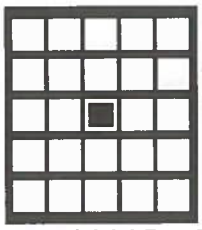


Figure 2.3.2.2.B. Single black pixel

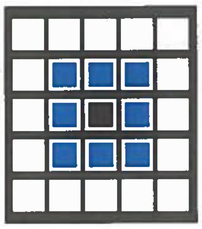


Figure 2.3.2.2.C dilated Pixel

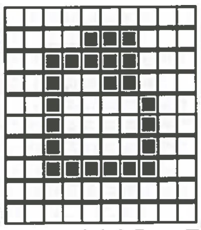


Figure 2.3.2.2.D Black pixels

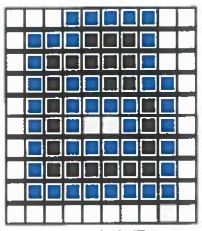


Figure 2.3.2.2.E dilated Pixels shown

A edge pixel, is a pixel that has neighbours that are not all black. A neighbourhood is the set of pixels that are reachable by one move in any direction. This includes diagonal moves. Figure 2.3.2.2A highlights all the edge pixels in red. Diluting takes edge pixels, and makes all the neighbours black pixels. Consider Figure 2.3.2.2.B above. Dilating, takes this single black pixel and ensures that all neighbouring pixels are also black as shown in Figure 2.3.2.2.C. The blue pixels show the newly added pixels. That is the simplest case. A more complicated example takes the image found in Figure 2.3.2.2.D and dilates once to get Figure 2.3.12.E where the newly added pixels are highlighted in blue. Figure 2.3.2.2.D is typical of the types of gaps encountered in skin classified images such as Figure 2.3.3.2.B. It is clear that if Figure 2.3.2.2.D were dilated once again, the gap would be filled.

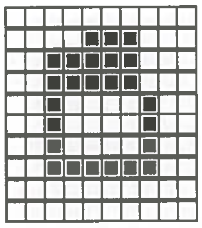


Figure 2.3.2.2.F An eroded image.

Eroding, like dilation, selects edge pixels. It then turns these edge pixels white. Consider Figure 2.3.2.2.A where the edge pixels have been highlighted. If these pixels are changed to white, then the new image becomes that found in Figure 2.3.2.2.F. As a second example, consider Figure 2.3.2.2.C above where the edge pixels are highlighted in blue. Eroding, takes these edge pixels and turns them white, leaving a single black pixel which is shown in Figure 2.3.2.2.B.

2.3.3.2.2 Locating largest region

A connected region refers to a group of pixels where any one pixel can be reached by one move from another pixel that is a member of the same group. There are two types of connectivity and they define what type of move can be made from one member to another. There is 4-way connectivity and there is 8-way connectivity. 4-way connectivity simple allows pixels to be added to a group if that pixel can be reached by at least 1 of the 4 possible directional moves from an already existing member of the group in just one step. The four moves are up, down, left and right. 8-way connectivity is similar but instead of just four moves, there are 8. The extra four correspond to the diagonal moves, top left, top right, bottom left, bottom right. If a pixel can belong to more than one group, depending on what type of connectivity is being enforced, then all those groups are merged to form a much larger group. Figure 2.3.3.2.2A and Figure 2.3.3.2.2B give examples of what happens to regions when they are grouped together based on 4-way connectivity and 8-way connectivity.

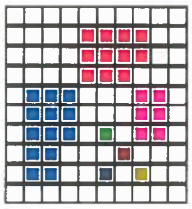


Figure 2.3.3.2.2A 6 regions being grouped together under 4 way connectivity. Each colour represents a different group. When a pixel can be reached from a neighbouring pixel by at least 1 of the four moves allowed by 4 way connectivity, it joins the neighbouring pixel's group' that is, it becomes the same colour as the neighbouring pixel. White pixels are ignored for this example.

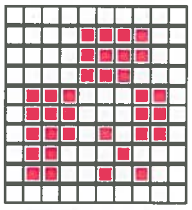


Figure 2.3.3.2.2B 1 regions being grouped together under 8 way connectivity. Each colour represents a different group. When a pixel can be reached from a neighbouring pixel by at least 1 of the eight moves allowed by 8 way connectivity, it joins the neighbouring pixel's group' that is, it becomes the same colour as the neighbouring pixel. White pixels are ignored for this example.

Once all the pixels have been put into groups, the number of pixels in each group is counted. That is, the mass of the group or region is calculated. These masses can then be sorted so that the larger masses are at the top of sorted elements and the smaller elements are at the bottom of the sorted elements. All regions with masses that are not number 1, number 2, or number 3 in the sorted list of elements are then deleted.

2.3.3.2.3 Understanding strategy

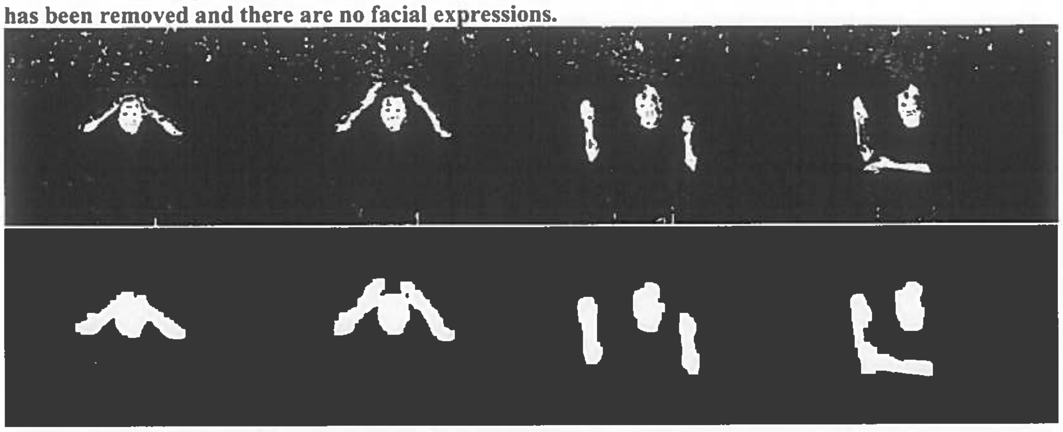
The erode operator is first called. The noisy regions in the binary image are small and sparse and so eroding once should cause many of the regions to disappear. In an attempt to repair any damage done to the human skin pixels by eroding, the dilute operator is called once. The dilute operator is then called a further 2 times to fill in some of the gaps in the skin. It was called twice because diluting once is often not enough to fill in the gaps in binary images as demonstrated by Figure 2.3.3.1.2.E. At this stage, the image should contain most of the correctly classified pixels with few or no gaps in the skin. However, there may have been some noise that remained even after eroding once. To remove most or all of the noise, the 3 largest connected regions are selected. Here, 4-way connectivity is used. The noise in images is sparse. The correctly classified human skin in images is dense. Therefore, 4 way connectivity should cause the noise to break up into small regions and should not significantly affect the number of human skin regions. Sometimes all the correctly classified pixels may be connected to form one large region or 2 large regions rather than 3 large regions. Always selecting the 3 largest pixels regardless of the actual size of the regions means that noisy data will often be selected as well as desired regions. Therefore, the largest region is always selected. However, the second largest region has to be at least 0.3 times the size of the largest region otherwise only the largest region is selected. Given that the second largest region has been selected, the third largest region has to be at least 0.3 times the size of the second largest region otherwise, only the first and the second largest regions are selected. Since noisy regions tend to be smaller than 0.3 times the smallest desired regions, this prevents noisy regions from being selected with the desired regions. The erode operator is called once more because there should typically be the same number of erode operators to the dilate operators.

The three largest regions were chosen for a reason. Let the human skin regions be classified such, that the hands were always connected to the forearms. Also, let each forearm be connected as one piece each. Let the head be connected as one piece. The two forearms and the one head make three pieces. Therefore, the following classification system will only be interested in the three largest regions. It will be shown below that these assumptions are overly optimistic. All other regions, except the three largest regions are deleted. The dilate operator is then called again to undo the effect of the previous erode operator. There has now been three dilate operators and three erode operators. This should close many of the gaps in the skin and leave the human skin regions to a similar state to that before the dilate operator was called. This was the first strategy attempted.

2.3.3.2.3 Results

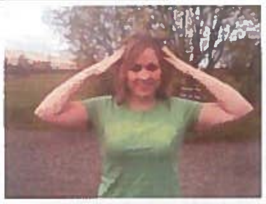
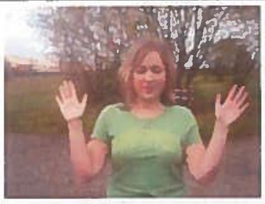
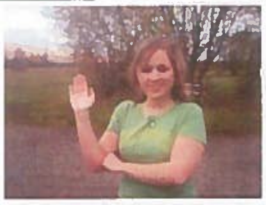
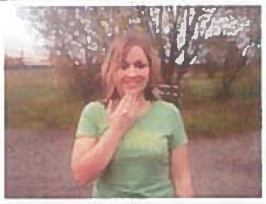
Figure 2.3.3.2.3.A shows what typically happens to skin classified images after the skin removal procedure has been applied. The procedure works very well and is also reasonably fast. Most gaps in the skin have been closed and the original shape of the classified human body parts is retained.

Figure 2.3.3.2.3.A. The top row shows the skin classified images. The bottom row shows the same images after the noise removal procedure. There is sparse noise in the top row and facial expressions are visible. The noise has been removed and there are no facial expressions.



3.1 Pose recognition

3.1.1 The poses

	**Figure 3.1.1.A - Pose 1 has both hands on the head with elbows away from the body.**
	**Figure 3.1.1.B - Pose 2 has both hands, one to either side, away from the body with the hands about neck height. The elbows are allowed to drop naturally**
	**Figure 3.1.1.C - Pose 3 has the hand held in the air at neck height and is away from the body. The right elbow is allowed to drop naturally. The left hand makes contact with the right elbow and the left elbow is allowed to drop naturally.**
	**Figure 3.1.1.D - Pose 4 has the right hand touching the chin. The elbows of the right hand are allowed to drop naturally. The left hand is allowed to drop against the left side of the body**

The four poses are shown and described in Figure 3.1.1.A & Figure 3.1.1.B & Figure 3.1.1.C & Figure 3.1.1.D. They were chosen as they had to satisfy two constraints:

1. Two poses, poses1 and 4, have at least one hand touching the face or head.
2. Two poses, poses 3 and 4, have one hand around the neck region with the other hand lower down the body.

These constraints were used to diversify the poses that would be used.

3.1.2 Training the model

The colour image has been skin classified and then cleaned to remove all noise. This cleaned image then represents a pose. The next stage in the system is to classify this pose. That is, the pose system will label the image. The label will be a number, 1, 2, 3, or 4, corresponding to which pose the image contains. The classification will be based on the properties of the image. Therefore, properties that are good at distinguishing one pose from another should be used in classification. The following subsection describes the properties that are used to describe each pose.

[Will talk about how this is different to previous system as there is no tracking involved just basic image recognition and so requires no knowledge of how poses change from one frame to another]

3.1.2.2 Features
3.1.2.2.1 Introduction

Various recognition systems already exists and are used in inspection of integrated circuits, recognition of protein molecules, scene analysis, and various other sectors. The input device used to initially capture the state of the scene of interest can vary; but, typically, the data captured from the input device is transformed to a multidimensional numerical vector typically known as a feature vector. Each dimension of the vector is a feature, so collectively the vector is known as a feature vector. Feature vectors tend to be better for representing a scene than the raw data initially captured from the input device. There are various reasons why this is true. One good reason is the reduction in dimensionality. Consider a scene that is captured with a single camera and is represented as a black and white picture with 640 rows and 480 columns. This is a 307200 dimension picture. A typical feature vector for that picture will have significantly less than 307200dimensions; it may have 10 dimensions for example. Another better reason for using features vectors over the raw data input is the reduction in data redundancy. The raw data input may have some dimensions that are just interpolations of other dimensions; some dimensions are just superfluous. A good feature vector should have much less data redundancy that the original picture.

Previous projects on gesture recognition have relied heavily on tracking the regions on the body that were deemed more dominant than others. Those using a single camera would find dominant regions were often occluded and either had to use more cameras, or had to encode in the feature vector that a dominant region was occluded. This project avoids any tracking and treats the problem as an image recognition task and significantly simplifies the task of labelling a pose. For example, the issue of occlusion is resolved by observing that if a particular body part is occluded with the images of a pose used as training data, then that very same body part should be occluded with future instances of that pose. Anything that is visible with the training data, under typical circumstances should be visible in all images of the same pose. All training images and unseen images contain the head region, for example. Typically, moments are used for describing an image in a feature vector.

3.1.2.2.2 Choice of moments
Features used should be a good representation of the scene but should have other properties that are relevant to the classification task. Consider the task of classifying handwritten digits. There are certain properties that a feature vector should posses in order to achieve sufficiently accurate results to classify handwritten digits. It should be scale invariance, for example. However, in some domains, scale invariance is an inhibiter for good performance for a classification system. A trivial system that has to classify an object's distance from a fixed point would find that the size of the object was a good

indicator of its distance. Another valuable attribute is that the features are not sensitive to noise. Furthermore, the less data redundancy in the features, the more useful the features. Given a set of various moments, including geometric, Legendre, Zernike, pseudo-Zernike, rotational and complex moments, Zernike and pseudo-Zernike outperform the others [16] in terms of information redundancy and overall performance. Zernike moments are also scale invariant and translation invariant which is a desirable property in a pose system. Therefore, they are used in this project.

3.1.2.2.3 Zernike Moments

Zernike moments are computed using a set of complex polynomials which collectively form an orthogonal basis set over the interior of a unit circle defined by the equation x^2+y^2+1.

Let m be a non negative integer: m $^{\geq 0}$. M defines the order of the moment.
Let n be a positive or negative integer from the set {x | |x| ≤ m} union {x | m - |x| = 2k, k Îó}.
Let r be the length of a vector from the origin to the pixel with Cartesian coordinate (x, y). This means that r = sqrt(x^2+y^2)

Let θ be the angle between the vector r and the x axis in counter clockwise direction.

This means that $\theta = \tan^{-1}\left(\frac{y}{x}\right)$

A two dimensional Zernike moment, A_{mn}, is defined as

$$A_{mn} = \frac{m+1}{\pi} \int_x \int_y f(x,y)[V_{mn}(x,y)]^* \, dx \, dy \qquad \text{where } x^2+y^2 \leq 1$$

The * denotes the complex conjugate. $f(x,y)$ is the function being describe. In the case of binary images where white pixels depict an object and black pixels depict the background, $f(x,y)$ corresponds to the value found at the pixel denoted by coordinate (x, y). This simply means that if there is a white pixel at (x, y) then $f(x,y)$ is 1 otherwise it is 0. $f(x,y)$ is denoted as P_{xy} for the reminder of the text.

Finite sized images are used with discrete height and width. There is no need to use integration when summation is equivalent. Therefore,

$$A_{mn} = \frac{m+1}{\pi} \sum_x \sum_y P_{xy}[V_{mn}(x,y)]^* \quad \text{where } x^2 + y^2 \leq 1$$

$V_{mn}(x,y)$ is the Zernike polynomial. Expressed as polar coordinates using r and θ calculated above, $V_{mn}(x,y)$ = $V_{mn}(r,\theta)$ and

$V_{mn}(r,\theta) = R_{mn}(r)\exp(jn\theta)$ where (r,θ) is defined over the interior of a unit disk. $j = \sqrt{-1}$ and so makes the moments complex.

Finally, $R_{mn}(r)$ is the orthogonal radial polynomial defined as

$$R_{mn}(r) = \sum_{s=0}^{\frac{m-|n|}{2}} (-1)^s \; F(m,n,s,r)$$

where

$$F(m,n,s,r) = \frac{(m-s)!}{s!(\frac{m+|n|}{2} - s)!\;(\frac{m-|n|}{2} - s)!} r^{m-2s}$$

and it is true that $R_{mn}(r) = R_{m,-n}(r)$ and $R_{mn}(r) = 0$ if the constraints between m and n are not satisfied.

3.1.2.3 Classification

Once 4 sets of training images, 1 set for each pose, have been processed to give feature vectors, these feature vectors must be modelled. When a new feature vector is presented, it must be assigned one of the four classes based on the model belonging to each class. This is known as classification. This project uses Gaussian distribution to model each of the 4 classes. When a new image is observed, the probability that it belongs to each class is calculated using Baye's rule and the image is assigned the label for that class.

A probability density function (pdf) gives the probability of the occurrence of each data point denoted as p(x). It has the probability that òp(x) =1 and p(x) ≥0 which are required for all probabilities. Modelling each class should produce a different probability density function if each class is different. Modelling each class from training data to obtain the probability density function is known as density estimation. The Gaussian distribution is one type of probability density function and will be good at modelling data that is clustered together in the space that it inhibits.

Our feature vectors have many elements so are multivariate feature vectors. The multivariate Gaussian distribution is given by

$$p(\mathrm{x}|\mu, \Sigma) = \frac{1}{\sqrt{\det(2\pi\Sigma)}} \exp\left(-\frac{1}{2}(\mathrm{x}-\mu)^T \Sigma^{-1} (\mathrm{x}-\mu)\right)$$

Where is the mean of the training data and is the covariance matrix for the training data. Therefore, the mean and the covariance are the only variables needed to calculate the Gaussian. They can be easily computed and this process is known as training the Gaussian:

$\mu = \frac{1}{P}\sum_{i=1}^{P} \mathrm{x}^i$ which is initiative and $\Sigma = \frac{1}{P}\sum_{i=1}^{P}(\mathrm{x}^i - \mu)(\mathrm{x}^i - \mu)^T$. A separate Gaussian is trained for each class to give a mean , and covariance matrix , for each class.

The probability p(class = c | x) is calculated for each of the four classes where x is the unseen feature vector and c is the name of the class. The label of the class that gives the greatest probability is assigned to x. However, the Gaussian distribution gives p(x | class = c) and not p(class = c | x). Baye's rule is used to calculate p(class = c | x).

Baye's rule:

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}$$

Once we substitute in our values, this becomes

$$\mathrm{p(class = c \mid x)} = \frac{\mathrm{p(x \mid class = c) \cdot p(class = c)}}{\mathrm{p(x)}}$$

p(x | class =c) is given to us by the Gaussian.

To calculate p(class = c), let there be N_c training examples in class c and let there be Nk be the sum of all training examples for every class. Therefore

$$\mathrm{p(class = c)} = \frac{N_c}{N_k}.$$

p(x) = Sp(x | class = c)p(class = c) over every class c.

3.1.3 Results

Each binary image was described by a 12 element feature vector where the 12 elements are real(A11), real(A22), real(A31), real(A33), real(A42), real(A44), img(A11), img(A22), img(A31), img(A33), img(A42),img(A44) where real(x) means the real component of x and img(x) means the imaginary component of x.

True pose	Label as pose (%)			
	1	2	3	4
1	100	0	0	0
2	0	99	0	1
3	0	2	98	0
4	0	0	6	94

Table 3.1.3.Aa Results of pose classification. The results show the percentage of correct classification of the verification data in blue and the error in red. Where 400 training data and 400 test data. 100 from each class.

True pose	Label as pose (%)				Overall Error (%)
	1	2	3	4	
1	94.9	5.1	0	0	5.1
2	1.0	93.9	3.0	2.1	6.1
3	0	2.1	95.9	2.0	4.1
4	0.08	0.9	4.52	94.5	6.5
					5.6

Table 3.1.3.B Results of pose classification. The results show the percentage of correct classification in blue and the error in red. The overall error is 5.6% which means a classification rate of 94.4%

100 images of poses from each class were selected for use as training data. A further 3800 were classified using the Gaussian distributions trained on the training data. 1309 images of pose 1.1492 images of pose 2. 506 images of pose 3. 493 images of pose 4. As can be from Table 3.1.3.A, the system has an overall 5.41% error rate, or 94.59% accuracy. Nearly Half the error is due to the system labelling pose 2 images. There is a simple explanation for this error.

The four poses used are distinctive. If the subject in the image maintained a pose while the image was being captured, one would expect an accuracy rate approaching 100%. However, the images classified here are taken from video sequences. The video sequence will capture the subject transitioning from one pose to another. The black and white cleaned image will not look exactly as any one pose based on our specifications of the four poses given in Figure 3.1.1.A- Figure 3.1.1.B, Figure 3.1.1.C and Figure 3.1.1.D. Examples of such transitional poses are given in Figure 3.1.1.E   Figure 3.1.1.F and Figure 3.1.1.G.

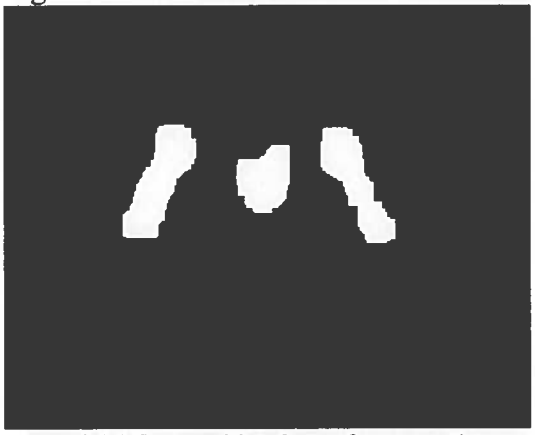


Figure 3.1.1.E - transitional pose from pose 1 to pose 2

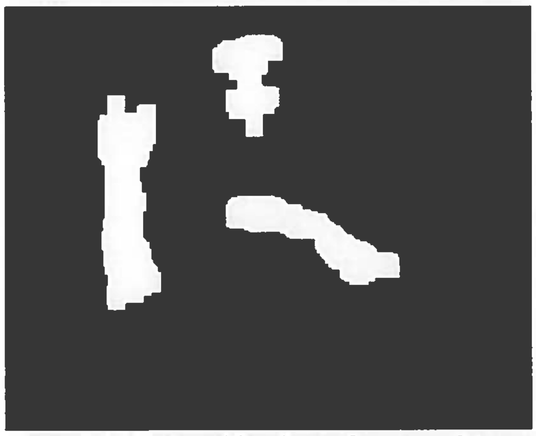


Figure 3.1.1.F - transitional pose from pose 2 to pose 3

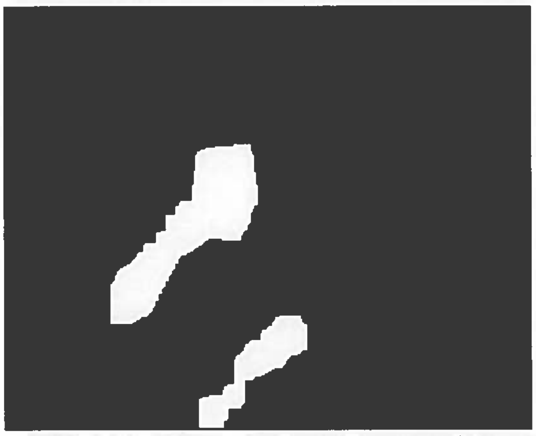


Figure 3.1.1.F - transitional pose from pose 3 to pose 4

Given such transitional poses, it is not difficult to understand why there will be misclassified images. Each of these transition poses had to be given a label based on the specification given of each pose. However, the distribution of the labels imposed on the transitional poses may be spatially more distant than the alternative pose in the 12D space given by the moment vector that describes the image. That is, the transitional pose may be more similar spatially, to one pose but was given another label based on the specification given for the poses above. For example, Figure 3.1.1.F is labelled as pose 4 because one hand is touching the face. Given just the human's left hand, either pose 3 or pose 4 would have been suitable labels. However, because the right hand is touching the face and this was a key feature for this pose, the transitional pose is labelled as pose 4. The classifier labelled Figure 3.1.1.F as pose 3.

4. Sequence Recognition and Results

4.1 Hidden Markov Model

4.1.1 Motivation

There already exist various powerful classifiers that can accurately classify objects from a scene given training data and a novel feature vector. However, sign language and gesture recognition in general require a sequence of feature vectors to correspond to the state of the scene over different time frames. Different people sign at different speeds so the same sign from different people will vary in duration. Therefore, the number of time steps is often unknown. The feature vectors for all time steps could simply be concatenated together to form a much longer feature vector if it were known. A system that uses such a feature vector would probably require a large amount of training data. This particular property of sign language and gesture recognition make traditional classifiers inept to the task of recognition in this domain. However, speech processing has the same issues with time. Speech processing has been successfully implemented using Hidden Markov Models (HMMs) which are able to resolve the issues.

HMMs were first introduced in a series of papers by Baum in the 1960s and were later used for speech processing from the early 1970s. It was not until the 1980s that the technology was fully embraced [11]. Currently, HMMs are being applied in the field of handwriting recognition, gesture recognition, sign language recognition and other time varying processes. It is because of a HMM's ability to deal with time varying processes that they are especially attractive in this project. Also, they have been successfully used in other similar projects and found to be good models. Therefore, they will be used here to model our sequences.

4.1.2 Theory

4.1.2.1 Deterministic Finite State Machine

Consider the simple Deterministic Finite State Machine (DFSM). There are a number of distinct states. Let S_i denote the various states with $1 \le i \le n$ where n is the number of states and S_1 is the start state as shown in figure 4.1.2.A.

S1
,
S2
,...
Si
,
Si+1
,...
Sn-1
,
Sn

Figure 4.1.2.A N number of distinct states.

There is a start state S_1, a number of end states {Se}, a fixed vocabulary {V}, and a number of edges from one state to other states, possibly none or to itself {Eij| exists edge(i, j)}. This is illustrated in Figure 4.1.2.B.

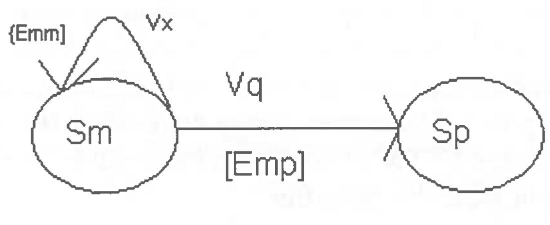


Figure 4.1.2.B Number of states with edges and transitions with example vocabulary.

State S_m has an edge from itself to itself and an edge from itself to another state, S_p. State S_p does not have any edges from itself to another state nor back to itself. The edge E_{mp} is denoted as it is because it moves from state S_m to state S_p. This is edge is used whenever the DFSM is in state S_m and the input V_q is encountered. Similarly, the edge E_{mm} is denoted as it moves from state S_m to state S_m. This is edge is used whenever the DFSM is in state S_m and the input V_x is encountered. It is clear to see that this method of defining the DFSM requires that a given vocabulary symbol yields a single edge to another or the same state. That is, a particular input vocabulary symbol must cause the DFSM to move to one and just one state given the state it is currently in. That means that each edge from a state must be associated with a different vocabulary symbol to all the other edges originating from that same very state. At any one time, the DFSM will only be in one state.

The DFSM described above is very powerful but any such classification system used will cause it to perform badly for gesture recognition. Consider that all the intermediate poses of a well gestured gesture are known in advance and all transitions from one pose to another are also known. A simple solution to gesture recognition would be to build a DFSM where the number of states correspond to the number of possible poses for each gesture to be classified. The end states would correspond to the final pose where each gesture could end. An edge would be present from one state S_j to another S_k if it is known that there is a possible direct transition from the pose represent by S_j to the pose represented by S_k. Assuming that the classifier that would analyse the scene would give 100% accurate classifications, this DFSM would accurately recognise gestures 100% of the time. Also, this system is time invariant. That is, given a user of the system whose gestures tend to have a greater duration than other users, then the DFSM would still recognise the signs accurately. The DFSM would just spend a greater amount of time in each state that it visits because from it may take a number of time steps before a new pose is encountered every time the DFSM enters a new state. However, there are real life considerations that prevent this from being the ideal system.

Firstly, the DFSM would be massive if the states to be used were chosen as described above. The second problem is what makes the system infeasible. Knowing all the poses in a well gestured gesture means very little in the real world filled with massive variability. A person doing the same gesture twice should lead to two very similarly gestures but never exactly the same gesture; this is even more significant from one user to another. There will be some variance from the perfect example of that gesture each time that gesture is realised.

Lastly, a classification system that works 100% of the time is highly unlikely unless in a trivial domain. There are classifiers that work with high levels of accuracy but it would be remarkable for a system to analyse a scene as used in this project with a diverse range of users and backgrounds with a number of poses. The system above demands a 100% classifier as a misclassification would lead to the DFSM choosing an incorrect transition state based on the current state and the output from the classifier.

A solution to the considerations above would probably result in a system much like the HMM.

4.1.2.2 Abstraction to HMMs

The definition given in **4.1.2.1** for a DFSM can be extend to accurately describe HMMs. Much like a DFMS, an HMM has a number of states with transitions or edges from one state to another. That is, each state S_i has an edge to all other states including itself, However, the edges have a probability assigned to each of them to indicate what the next state will be in the following time step. The sum of all probabilities assigned to edges leaving any state has to sum to 1. This means that the HMM has to use at least one edge for the following time step. If it is impossible to get from one state, S_m, to another, S_p, after just one time step, then the edge from S_m to S_p carries a probability of 0. The types of HMMs used in this project are first order. This means that the probability on each edge is based on the current state of the HMM. This means that the probability of being in state S_p, which is reachable by an edge from S_m, does not depend on the path that the HMM took to get to S_m but only on the probability on the edge connecting S_m to S_p. Mathematically,

$$P(q_{t+1}= S_p|q_t= S_m) = P(q_{t+1} = S_p| q_t = S_m, q_{t-1} = S_n, q_{t-2} = S_o, \ldots)$$

This means that the state of the HMM in time t+1 is independent of the states of the HMM in time t-1, t-2, …, 1.

4.1.3 Using HMMs

It is equivalent to say that the HMM outputs an element from the vocabulary whenever it uses an edge or at every time step because the HMM uses an edge every time step. Here, the time step option will be used rather than the edge option for clarity. Therefore, at each time step, the HMM outputs an element from our vocabulary which means that there will be a sequence of t outputs after t time steps. If the HMM was trained with sufficient data and its other attributes were correctly set, then the outputted sequence should be typical of the instance that was used as training data. Such sequences, or variants, will be used to classify the different gestures.

Using HMM to classify gestures is a two step process. Let there be k gestures that will be modelled by the HMMs. Therefore, k HMMs will be trained. One HMM for each gesture by providing the sequence of observations for every instance of the gesture's training data. This trains the HMM and is the first step in using HMMs. There are two ways a new unknown gesture may be trained by the system. The first is simply to input a sequence consisting of the elements of the vocabulary which is much like the sequence the HMMs produce themselves. The second option is to pass a sequence which is a column of probabilities at each time step. The column of probabilities corresponds to the observation of each element from our vocabulary at each time step. That is, each entry in the column of probabilities corresponds to the probability that the event observed at that time step was that element in the vocabulary. The HMM then uses the inputted sequence and outputs a log likelihood. This is simply the log of the probability that the inputted gesture was created by that particular HMM. This is done for each HMM and the HMM with the greatest log likelihood is the HMM that would most likely produce the sequence.

That is, the HMM with the greatest log likelihood probably corresponds to the class of gestures that was used to train the HMM. Let O_j correspond to the observation at time j that will be a member of the input sequence at time j.

There are three main problems associated with using HMMs:

1. What is the probability of being in a particular state, S_i, given the input sequence which corresponds to
$$P(q_t = S_i | O_1O_2O_3O_4.. O_t).$$
TThe naïve solution would be a brute force calculation would result in an exponential number of calculations. However, dynamic programming enables a much smaller number of calculations and so solves this problem.
2. Given the sequence of inputs, what is the most probably path that the HMM took and what is the probability of the HMM generating such a path? The VITERBI algorithm, which is another dynamic programming algorithm, is used to solve this problem.
3. Given the input sequence $O_1O_2O_3O_4.. O_t$, what is the maximum likely HMM that has been trained that could have produced this sequence? The Expectation-Modification (EM) algorithm solves this particular problem.

There has been much work on HMMs that all three problems have been solved sufficiently well and more detailed presentation of the solutions can be found in [11].

4.1.3 Specifics to this project

The class of HMMs used here are left to right HMMs. At each time step, the HMM will use an edge that will cause it to remain in the same state or to move to a connected state that is to the right of the current state.

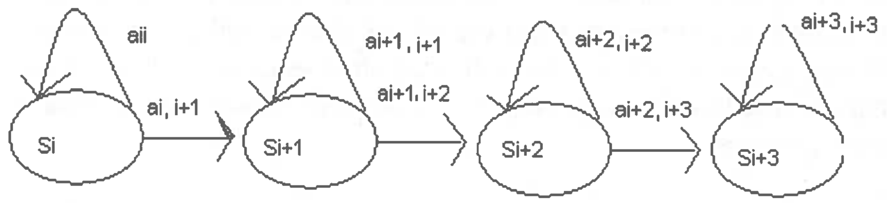


Figure 4.1.3.A Left to Right HMM where there is no path from a state back to itself which includes other states. Also, edges connecting a state to all the states to its right are not exhaustively drawn here but are also a feature of a left to right HMM.

This is displayed in Figure 4.1.3.A. There are no edges that allow the HMM to move from state k to j, where k > j. Let the matrix A be the transitional matrix for a given HMM where the entry on the kth row and jth column corresponds to the probability that the HMM will move to state j when in state k. Each HMM has its own distinct matrix A.

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}.$$

In order to force the HMM to be a left to right HMM, the transitional matrix is edit so that the probability of moving from state j from k when k>j is always 0 to get

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}.$$

$$A = \begin{bmatrix} 0.4676 & 0.1597 & 0.3072 & 0.0655 \\ 0 & 0.0868 & 0.1201 & 0.0874 \\ 0 & 0 & 0.6349 & 0.2380 \\ 0 & 0 & 0 & 0.3597 \end{bmatrix}$$

An example transitional matrix:

4.2 Experimentation

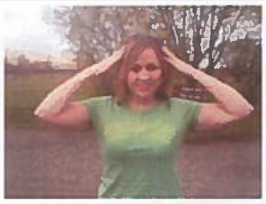
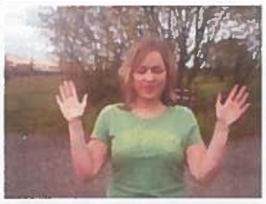
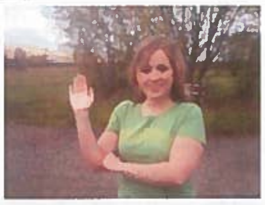
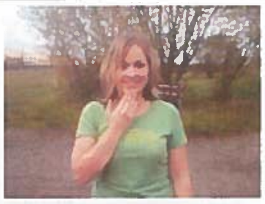
			

Figure 4.2.A Pose 1	**Figure 4.2.B Pose 2**	**Figure 4.2.C Pose 3**	**Figure 4.2.D Pose 4**

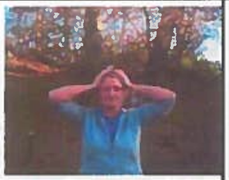
				

Table 4.2.A The sequences that make up each gesture

100 videos were collected of 10 people each doing 10 different gestures. Each gesture consists of a sequence of the 4 poses. The 4 poses are shown in Figure 4.2.A-Figure 4.2.D. The 10 gestures are given in Table 4.2.A. Each sequence consisted of 39 different frames and each frame for each video was classified using the pose classification procedure described in chapter 3. This created a 39 element vector for each video where the ith element in the vector corresponds to the label for the ith frame in that video.

Each HMM that was used to model each gesture is a 4 state, left to right HMM.

To train a HMM using Kevin Murphy's Hidden Markov Model Toolbox for Matlab, one simply has to pass an initial transitional matrix A, and each element of the matrix is updated based on the previous state of the matrix and the set of training data passed as input. The EM algorithm is used to estimate these values. The initial transitional matrix is created randomly and some of its elements are set to 0 to make the HMM a left to right matrix. Training an HMM has randomness embedded into it. Therefore, the results obtained from training each HMM with the training data once and then classifying the remaining gestures will not necessarily give accurate results because of the randomness. Therefore, the experiment was performed 100 times to give more accurate results.

There were two set of experiments. The first simply used the 39 element feature vectors to train and then test the system. The second set of experiments modified the 39 element feature vector in an attempt to correct some of the mislabelling from the pose recognition state.

Consider the first gesture made up of the sequence 1-2-3. There should be no frame labelled 4 for a particular 39 frame video for this gesture. Also, no frame should be labelled 3 before the label 2 is encountered. The gesture has a syntax. A finite state machine was created for each gesture which accepted a sequence of labels that described the gesture and rejected sequence that did not describe the gesture. A finite state machine was created for each gesture. Both the training data and test data were used as input to the finite state machines. If the finite state machine accepted the sequence, then the sequence remained as it was. However, if the finite state machine did not accept the sequence, then the particular label in the sequence that was deemed erroneous was changed to that which it should have been given the grammar. That is, the label was correct so that the sequence would be accepted by the finite state machine.

The results for the 100 experiments with the unmodified data is given in Table 4.2.B and the results for the 100 experiments with the modified data is given in Table 4.2.C.

Gesture				Classifed as							Total Error
	1	2	3	4	5	6	7	8	9	10	
1	0.97	0	0.03	0	0	0	0	0	0	0	0.03
2	0	0.8	0	0	0	0.07	0	0	0.13	0	0.2
3	0.23	0	0.76	0	0	0	0.01	0	0	0	0.24
4	0	0	0	0.8	0	0	0.2	0	0	0	0.2
5	0.13	0	0	0.07	0.8	0	0	0	0	0	0.2
6	0	0.1	0	0	0	0.86	0	0	0.04	0	0.14
7	0	0	0	0.07	0.08	0	0.85	0	0	0	0.15
8	0	0.01	0	0	0	0	0	0.8	0	0.19	0.2
9	0	0.21	0	0	0	0	0	0	0.79	0	0.21
10	0	0	0	0	0	0	0	0	0	1	0
											0.16

Table 4.2.B. The results for 100 experiments with unmodified data. There is an 84% correct classification rate.

Gesture	Classifed as										Total Error
	1	2	3	4	5	6	7	8	9	10	
1	0.98	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02
2	0.00	0.90	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.10
3	0.13	0.00	0.87	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.13
4	0.00	0.00	0.00	0.90	0.00	0.00	0.10	0.00	0.00	0.00	0.10
5	0.07	0.00	0.00	0.03	0.90	0.00	0.00	0.00	0.00	0.00	0.10
6	0.00	0.03	0.00	0.00	0.00	0.97	0.00	0.00	0.01	0.00	0.03
7	0.00	0.00	0.00	0.02	0.03	0.00	0.96	0.00	0.00	0.00	0.04
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.90	0.00	0.09	0.10
9	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.89	0.00	0.11
10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
											0.07

Table 4.2.C. The results for 100 experiments with modified data. There is a 93% correct classification rate.

The results show that the classification process is successful with 84% correct classification rate and works particular well with modified data with 93% correct classification rate. Good results were obtained with a small training set. However, there are only 10 different gestures. There experiments do not show if the gesture recognition stage is scalable to a much greater number of sequences.

5. Conclusion

5.1 Summary

The system has a number of recognition stages; each with varying success. The first stage is the skin recognition and background removal stage. Here, people's head, hands, and forearms are automatically extracted from a colour image if a number of small constraints are satisfied. This stage worked sufficiently well. Much of the noise is removed in the black and white image produced and most of the holes in the skin are filled. However, this stage contributed the most to the total processing time of the entire system. The success of that stage will influence that of the gesture recognition stage.

The gesture recognition stage uses Zernike moments as features. This stage works particularly well but is influenced by the noise that is propagated from the previous stage. Unlike previous projects, there is no tracking involved in this stage and so the recognition problem becomes an image recognition problem, and issues such as object occlusion do not greatly inhibit performance on current gestures or potential future gesture recognition. Fortunately, the moments used are especially resistant to noise and are powerful descriptors. Those properties allow this particular recognition stage to have a high accuracy rate with greater than 94% for each class of signs despite a small training set. Clearly, this accuracy rate will have an impact on the performance of the final stage which is the sequence recognition stage.

The powerful HMMs are used to recognise a sequence of gestures which have been labelled by the previous stage. Again, good classification results have been obtained with over 84% gesture classification which increases to over 93% when the data used is automatically corrected for incorrect labelling. The performance was hindered by a small training example and any error present from the previous stage would not have been helpful in classification. There were only 10 sequences classified in this stage so there is no evidence of scalability but the results are encoring for such a small training set.

On a Pentium 4 3.0 GZ desktop using the Windows XP operating system it takes 10 seconds to process each frame using the Matlab Image Toolbox and less than a second to classify a gesture. This is a very slow rate considering the success of previous sign language recognition system that works in real time. However, the system is implemented for correctness and not fully optimised.

5.2 Future Work

The first recognition state had a large number of training points and test data points. Therefore, accurate results on the performance of the recognition state were obtained. The results are encouraging but the performance could be greatly improved if a better classification rule were used. Although there are rules which use the edges obtained from edge detection algorithms to improve the recognition of skin, the edges obtained from edge detection algorithms are not necessarily fully connected. Therefore, the uses of such rules are not guaranteed to outperform the simple classification and noise removal procedure used here. Considering that this state is the bottleneck in the performance of the system, any new background removal process should reduce the time needed to complete this process. Furthermore, this state does not exploit properties from one frame to another. For example, given the centre of mass and the mass of the image, it is reasonable to expect the centre of mass for the following frame to be in a similar location. This information can be used to construct a window of a particular size and location, representing where the human skin coloured pixels will reside; rather, than considering every pixel in the image. Such an addition to the simple rule, should aid in reducing the processing time for each frame any potential noise outside the window will be automatically eliminated so should produce cleaner images.

Cleaner images currently mean greater freedom from noise and less gaps in the skin. However, gaps in the skin are useful for capturing facial expression. There exist head locating systems but there has not been much work on automatically classifying facial expressions in the task of gesture recognition. The background removal stage currently removes any facial expression. A better solution would be to isolate the facial region and clean the image whilst preserving the facial expression for use in the next stage of gesture recognition.

Zernike moments are clearly very well suited to learning the different gestures. However, they are expensive to compute. There are pseudo Zernike moments that should give similar levels of expressiveness but can be computed in a fraction of the time. Further experiments with a larger number of gestures would be valuable to show the scalability of the gesture recognition stage using the moments. Furthermore, the gestures chosen here are very different from each other. A set of gestures that require less movement to get from one gesture to another would significantly increase the number of potential gestures and would really show how powerful the moments were for this recognition task. This new set of gestures will mean that one gesture would be different from another, but not to the extent that they are different in this system. The moments would have to be able to differentiate between gestures that are centimetres apart rather than the current greater distances between gestures. Currently, all the gestures involve movement of the arms. Fingers can be moved much quicker than arms. The newer richer set could incorporate the use of fingers to further distinguish one gesture from another. Clearly, the previous background removal stage would have to produce much cleaner images, with better focus on the fingers. While finger spelling recognition systems exist, they tend to focus on tracking and not simply image recognition. An extension of the system presented here could focus on using image recognition to incorporate the fingers and finger spelling.

It would be useful having a richer set of gestures as this allow the number of short gesture sequences to increase greatly. Short sequences will be necessary for a practical system as larger sequences are more prone to mislabelling and error. Unfortunately, a new richer set of gestures would mean significantly more training data. It may be possible to simple increase the number of moments used to better describe each gesture. More expressive feature vectors should reduce the overall amount of training data needed in the newer richer set.

Advances in the gesture recognition stage would have an impact on the sequence recognition stage. The frame rate of the camera would also have an impact on this stage, but the exact effects are unknown. A greater frame rate would mean less subtle changes would be captured which would be important when the fingers are used. However, this would also increase the length of the sequences. If the frame rate were too great, then it would take many frames to move from one gesture to another. Such a side effect is detrimental to the correct working of the HMM The HMM always attempts to move to further states when a new input is encountered. If the frame rate is too great the there will simply be long sub sequences of the same label within the larger sequence, then the HMM will be forced to continuously move to new states prematurely. A new model that changes this behaviour of the HMM that does not force premature transition would be very useful. Such a model would mean that the system could better adapt to individual needs. For example, different individuals could use different cameras with different frame rates. Suffice that the frame rates are sufficient to capture all the gestures, and then the model should be able to classify them with the same level of accuracy. This is not the way it is with HMMs. If the frame rate change and so produce sequences of different lengths, then the HMM would be more successful at classifying the shorter sequences than the longer sequences.

Bibliography

[1] B.D. Zarit, B.J. Super, and F.K.H. Quek, "Comparison of five color models in skin pixel classification" in ICCV' 99 Int'l Workshop on recognition, analysis and tracking of faces and gestures in Real-Time systems, 1999.

[2] H. Hermann ; K. Karl-Friedrich, "HMM-based continuous sign language recognition using stochastic grammars" GW99 - The 3rd Gesture Workshop: Towards a Gesture-Based Communication in Human-Computer Interaction, Gif-sur-Yvette, France, March. 1999.

[3] J. Ma, W. Gao, "A parallel Multi-stream model for sign language recognition", Institute of Computing Technology, China.

[4] J. Yang and A. Waibel, "A real-time face tracker", in Proceedings of WACV'96, pp. 142-147, 1996.

[5] K. Murakami , H. Taguchi, "Gesture recognition using recurrent neural networks", in Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology, p.237-242, 1991.

[6] K. Sandeep; A.N. Rajagopalan, "Human Face Detection in Cluttered Colour Images Using Skin Color and Edge Information"

[7] M. Choueiri; N. El-Sayegh, W. Said, "Real-Time Face Detection and Recognition", 2004

[8] M.J. Jones and J.M. Rehg, "Statistical color models with application to skin detection, " Computer vision and Pattern Recognition, pp 274-280.

[9] M Ohki, H. Sagawa, T. Sakiyama, E. Oohira, H. Ikeda, H. Fujisawa, "Pattern Recognition and Synthesis for Sign Language Translation System", in Proceedings of the first annual ACM conference on Assistive technologies, ACM Press, pp.1-8, 1998.

[10] R.K. Singh and A.N. Rajagopalan, "Background learning for robust face recognition", In Int'll Conference on Pattern Recognition, 2002.

[11] R.L. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," Proceedings of IEEE, 77(2), pp. 257-285, 1989.

[12] S. Akyol; U. Canzler, "An information terminal using vision based sign language recognition", ITEA workshop on Virtual Home Environments, C-LAB Publication, Band 12, pages 61-68, Germany 2002 .

[13] S.S. Fels and G.E. Hinton, "Glove-Talk: A neural network interface between a data-glove and a speech synthesizer," IEEE Trans. Neural Networks, vol. 4. pp. 2-8 Jan. 1993

[14] T. Sherwood, B. Calder, "Automated Design of Finite State Machine Predicators", UCSD

[15] T. Starner, J. Weaver, and A Pentland, "Real-Time American Sign Language recognition Using Desktop and Wearable Computer Based video, " Technical Report 466, Perceptual Computing, MIT Media Laboratory, July 1998.

[16] HIPR2 Image Processing Learning Resources, http://homepages.inf.ed.ac.uk/rbf/HIPR2/ 2004

A. Appendix

A.1 Image capture

Capturing a video of good quality is important for the subsystems that rely on the video. There are certain constraints, that once satisfied, will greatly improve the overall classification rate of the system:

1. The video should be captured in an environment where there is ample natural light or the camera is white balanced. Image A.1.A is a frame that satisfies this constraint because there is ample natural light. Image A.1.B is a frame that does not satisfies this constraint because there is too much ambient light.

Image A.1.A. A frame from a video with ample natural sunlight.

Image A.1.B. A frame from a video with ambient sunlight.

2. There should only be one human visible in each video.
3. The human should be completely contained, from the hip up, in the video. However, there is no loss of performance if the user is entirely contained in the video from head to toe.
4. The human must have and only have the following body parts completely visible and free from any type of jewellery, accessories, and cosmetic make-up that significantly alters the natural look of the skin:
   - The hand and forearm region from the tip of the fingers to the elbows.
   - The facial region from the top of the hairline to the bottom of the chin. However, a visible neck region will not lead to a loss of performance. Image A.1.A is a frame that satisfies this constraint; Image 1.1.C is a frame that does not satisfies this constraint In Image A.1.C, the human's naval region is slightly exposed. This is only a slight expose and the following section will correct for it. However, it is best to satisfy this constraint.

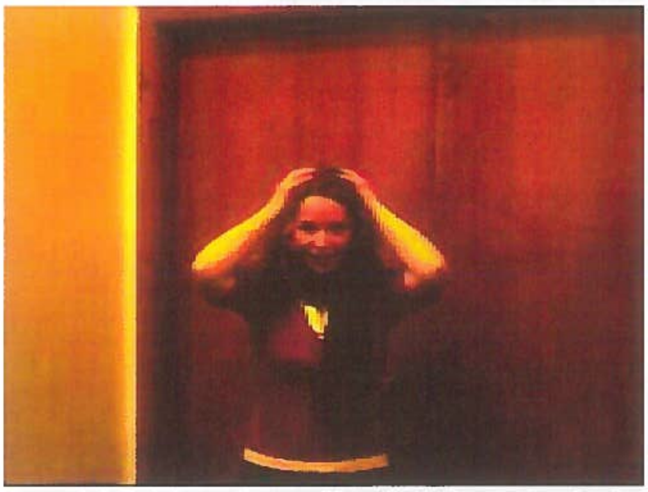
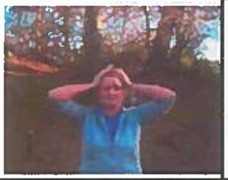
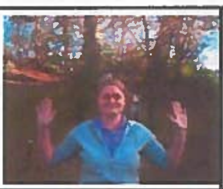
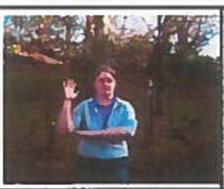
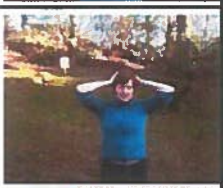
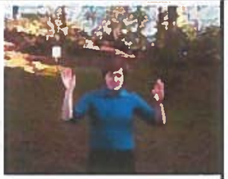
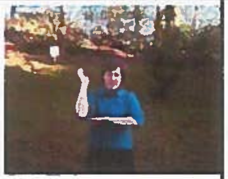
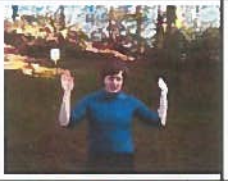
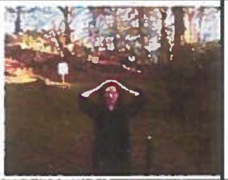
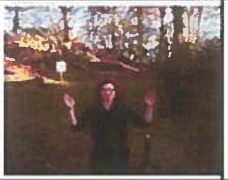
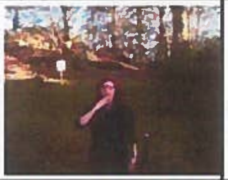
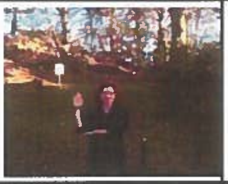
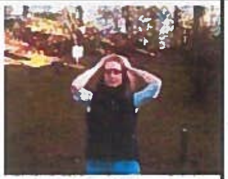
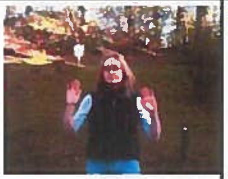
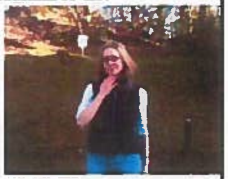
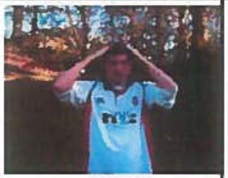
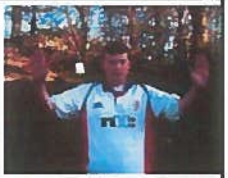
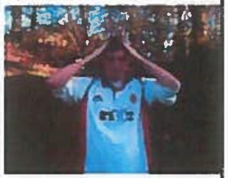
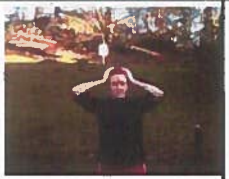
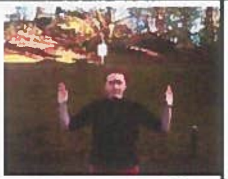
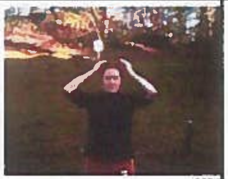
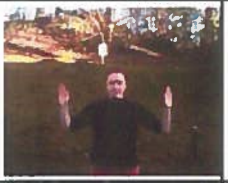
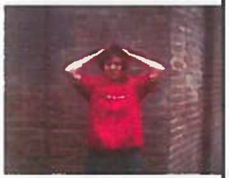
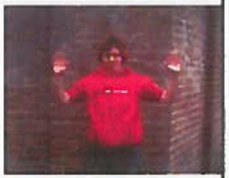
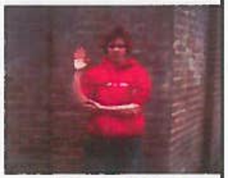
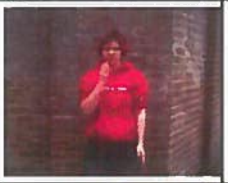
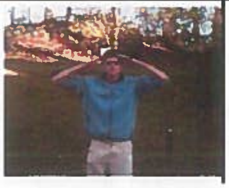
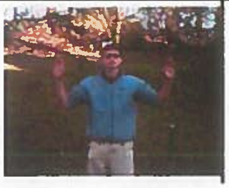
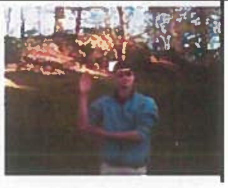
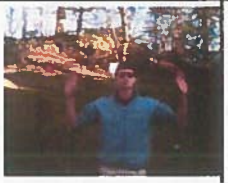
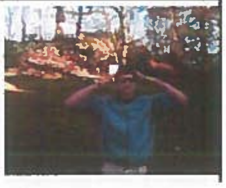


Image A.1.C. A frame from a video where the naval region is slightly exposed. This is not an ideal frame.

5. The frame rate used to record training data must be the same frame rate used in any testing. The frame rate was 4 frames per second.

A sign can either be captured one frame at a time, collectively making a sequence of frames, or it can be captured as a video which can then be processed to produce each frame from the video. At this stage in the development of the system, it is not possible to capture and process each frame in real time. Therefore, each gesture was recorded as a video and the program Konverter was used to extract each frame from the video and saved as a jpg.

A.2 The gestures

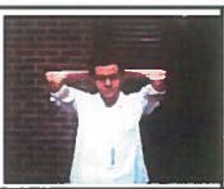
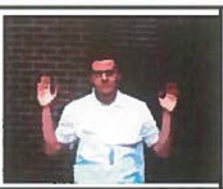
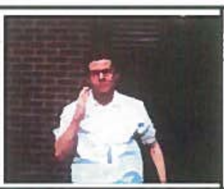
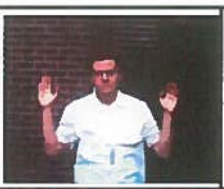
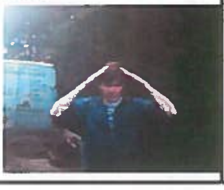
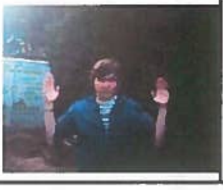
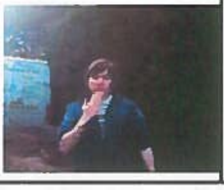
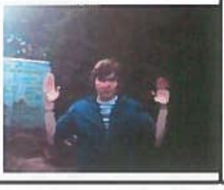
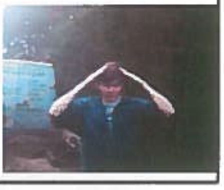
				
				

Table A.2.A The gestures

Table A.2.A shows a table where each row represent the grammar of each of the gestures. The frames are all taken from a video used to train or test the system. The able also shows the different backgrounds that were used in background extraction.