Chereme-Based Recognition of Isolated, Dynamic Gestures from South African Sign Language with Hidden Markov Models

by

Christopher Rajah

A thesis submitted in fulfillment
of the requirements for the degree of
Master of Science
Department of Computer Science
University of the Western Cape
Supervisor: Prof. Christian W. Omlin
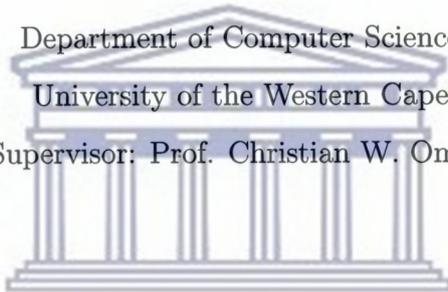
UNIVERSITY *of the*
WESTERN CAPE
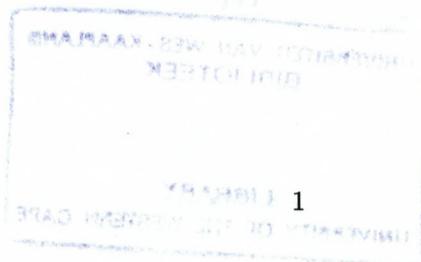
Declaration

I declare that this thesis is my own work, that it has not been submitted for any degree or examination in any other university, and that all the sources I have used or quoted have been indicated and acknowledged by complete references.

Christopher Rajah   January 2006

Signed:

2

Acknowledgements

I would like to thank my supervisor Professor Christian Omlin for his patience and dedication and also my wife for helping me to edit.

UNIVERSITY *of the*

WESTERN CAPE

Abstract

Much work has been done in building systems that can recognise gestures, e.g. as a component of sign language recognition systems. These systems typically use whole gestures as the smallest unit for recognition. Although high recognition rates have been reported, these systems do not scale well and are computationally intensive. The reason why these systems generally scale poorly is that they recognize gestures by building individual models for each separate gesture; as the number of gestures grows, so does the required number of models. Beyond a certain threshold number of gestures to be recognized, this approach becomes infeasible.

This work proposes that similarly good recognition rates can be achieved by building models for subcomponents of whole gestures, so-called *cheremes*. Instead of building models for entire gestures, we build models for cheremes and recognize gestures as sequences of such cheremes. The assumption is that many gestures share cheremes and that the number of cheremes necessary to describe gestures is much smaller than the number of gestures. This small number of cheremes then makes it possible to recognize a large number of gestures with a small number of chereme models. This approach is akin to phoneme-based speech recognition systems where utterances are recognized as phonemes which in turn are combined into words.

We attempt to recognise and classify cheremes found in South African Sign Language (SASL). We introduce a method for the automatic discovery of cheremes in dynamic signs. We design, train and use hidden Markov models (HMMs) for chereme recognition. Our results show that this approach is feasible in that it not only scales well, but it also generalizes well. We are able to recognize cheremes in signs that were not used for training HMMs; this generalization ability is a basic necessity for chemere-based gesture recognition. Our approach can thus lay the foundation for building a SASL dynamic gesture recognition system.

4

Contents

UNIVERSITY *of the*

WESTERN CAPE

6

List of Figures

7

1 Introduction

1.1 Motivation

Imagine you want to have a conversation with a deaf person. Already this may seem a daunting task, especially if you have no idea on how to communicate using sign language. Such is the problem faced by milions of deaf people who are unable to communicate and interact with hearing people. The Deaf are marginalized in society and are made to feel unimportant and unwanted. How then can we help to improve the quality of life of the deaf community?

The solution may lie in information technology. There have been great strides in the field of human-computer interaction [33]; today, the use of a keyboard or mouse is seen as an obtrusive and almost inefficient way for human-computer interaction. In our quest to seek a most natural form of interaction, we have promoted the development of recognition systems, e.g. speech and gesture recognition systems [36] [28]. The advancements in information technology thus holds the promise of offering solutions for the deaf to communicate with the hearing world. Furthermore, the cost of computer hardware continues to decrease in price whilst increasing in processing power, thus opening the the possibility of building real-time sign language recognition and translation systems [42]. Real-time sign language translation systems will be able to improve communication and allow the deaf community to enjoy full participation in day-to-day interaction and access to information and services.

Sign languages all over the world use both static and dynamic gestures, facial expressions and body postures for communication. In this work, we are limiting ourselves to the investigation of dynamic gestures, i.e. we are concerned with the reliable recognition of dynamic gestures. The recognition of other components of sign languages is beyond the scope of this thesis. Nevertheless, our investigation will be one of the foundations for the development of a full sign language recognition and translation system. We have chosen dynamic manual gestures from South African Sign Language and ask how we can build a system that recognizes dynamic signs while being unobtrusive and allowing for scalability. Gesture-based recognition systems may also be used in other applications

8

including robotics, control, and videoconferencing.

1.2 Premises of the Investigation

The hand gesture recognition that contributes to natural man-machine interface is still a challenging problem. Closely related to the field of gesture recognition is that of sign language recognition. Sign language , which also contains structured gestures, is one of the most natural means of exchanging information for most hearing impaired people [43] [6]. Sign language recognition systems are developed to provide an efficient and accurate mechanism to transcribe sign language into text or speech.

This process, however, poses challenging problems. If we build a system on the idea that whole signs are the smallest recognizable unit, then we face the challenge of scalability. Most systems developed to date need to be retrained when new signs are added. The problem is further compounded by the need to store large amounts of data for recognition and classification purposes. Furthermore, if we aim for real-time recognition, then computational time for processing also becomes a very important factor.If the system is to function remotely, bandwidth may be an issue. Sending whole sign information down the wire to the recognition engine may then not be the ideal solution. Most systems make use of expensive data capture equipment such as electronic gloves, which do not make for a natural scalable solution [37] [3]. These are some of the problems that hinder significant progress in the development of sign language recognition systems from becoming mainstream.

The premise for the work presented here is to reduce costs in both resources and the development of such systems. We make use of sign subcomponents, so-called cheremes, as the smallest recognition unit to reduce bandwidth, storage and processing costs. Since the number of possible cheremes is indeed limited, we are able to fix the processing time and cost to acceptable limits. This approach also opens the way for the development of scalable systems that are able to accommodate expansions of sign language vocabularies.

We focus on the manual gestures for this research and we track only hand movements [38]. We ignore finger spelling, although it does feature to a large extent in signs for names [31]. Isolated

sign production has clear and definite silence intervals separating each sign. This approach makes the segmentation process in the image analysis a simpler one for sign recognition. We focus on isolated, dynamic movement signs. Dynamic signs are made up of dynamic sub-components. We focus on the identification, recognition of these sub-components.

1.3 Problem Statement

Our goal is to abandon the notion of whole signs as the smallest recognition units of South African Sign Language and replace them with dynamic cheremes [37]. In order to demonstrate that this approach to building gesture recognition systems with large vocabularies is indeed feasible, we need to demonstrate that gestures cannot only be automatically subdivided into cheremes and models built for these individuals gesture components, but also that cheremes from gestures used for building models can be recognized in previously unseen gestures; in other words, we have to demonstrate a generalization capability. It is this capability to recognize cheremes in unknown gestures which will make the approach scale well as the number of gestures increases.

1.4 Research Hypothesis

In order to solve the above problem, we will investigate and answer the following research questions:

- Since we have changed our perspective of gesture recognition from whole gestures to their constituting cheremes, it is imperative to ask: is it possible to automatically identify begin and end points of cheremes? This is useful both for off-line preparation of a training data set which requires the 'segmentation' of whole signs into cheremes as well as for on-line recognition where convenient definition of begin and end points of cheremes will facilitate real-time recognition. In speech recognition, phoneme recognition is made more difficult by the presence of phoneme coarticulation which blurs the begin and end points of phonemes. If possible, we would like to define cheremes in such a manner that avoids similar gesture coarticulation and thus acilitates chereme recognition.

- Since cheremes are dynamic, we need to ask: what is an appropriate manner for describing cheremes that can be used for model development and recognition, i.e. how can we capture

10

the dynamic essence and distinguishing features that will allow us off-line chereme model construction and on-line recognition?

- The premise of our investigation is that cheremes allow for scalability to an unlimited number of gestures from a limited number of cheremes. We thus need to ask: are we able to generalize, i.e. recognize cheremes that were isolated from known gestures and used for building a model in new, previously unseen gestures where the same cheremes occur?

- Gesture recognition uses as input a sequence of video frames. All of the above questions rely on the extraction of appropriate, salient features from video sequences. What are these appropriate, salient features that will allow successful, real-time chereme recognition?

1.5 Technical Objectives

In order to find answers to the above research questions, we need to achieve the following technical objectives:

- Preparation of a training set: We have to find a set of signs that properly describes all the dynamic cheremes that we want to recognize and choose the number of cheremes such that the recognition process is sufficiently rich. We are not looking to characterize all dynamic gestures occurring in sign languages in terms of their cheremes. Thus, our investigation will necessarily focus on a very small subset of possible gestures occuring in sign languages.

- Chereme definition: In order to avoid the problem of chereme coarticulation, it is helpful to define cheremes such that their begin and end points are easily and automatically identifiable; we thus need to devise a method for easy identification. We are aware of the mutual dependence between richness of the defined set of cheremes and the need for easy identification of their begin and end points.

- Preprocessing: Image preprocessing and feature extraction are crucial to the success and achievable performance levels for gesture recognition. Image preprocessing will involve segmentation of continuous video; since various video segmentation methods exist, we will have to identify and implement a method that is appropriate for our purposes. Some of these

11

methods are based on changes in velocity and directions while others use analysis of joints and degrees of freedom. Following image preprocessing, we will need to identify suitable features for proper characterization and definition of hand movements.

- Model development: What distinguishes cheremes, and thus gestures, from one another is their dynamic behavior. We thus need to identify an appropriate model that can capture the essence of individual cheremes and that can then be used for discrimination between different cheremes. The literature offers a number of modelling techniques.

- Performance analysis: We need to empirically verify that our model can indeed identify cheremes in previously unseen gestures, i.e. these are gestures that were not used in the preparation of the training set, and measure the accuracy of chereme recognition and classification.

1.6 Research Methodology

In order to achieve the above technical objectives, we will use the following research methodology:

- We will implement image preprocessing algorithms and extract salient features that are appropriate for our objectives. Very briefly, within each video frame, we will identify skin regions using standard skin segmentation algorithms and appropriate colour space; these regions define regions of interest that are common for the entire video sequence. We calculate difference images for consecutive frames and use the regions of interest from previous steps to remove noise from the difference images.

- The difference of images can be interpreted as a distribution of the motion over the image space; different motions have different characteristic distributions. We extract features such as center of motion, relative change of the center of mass, and wideness and intensity of motion.

- We will use hidden Markov models (HMMs) for characterizing different cheremes. This involves choosing a particular type of model (number of states, transitions, etc), training the models and their use for discrimination. HMMs are one of the standard method used to

12

model processes whose output can be observed but whose underlying dynamical structure is unknown.

- We test the discrimination performance of our HMMs by letting them discover their corresponding cheremes in previously unseen gestures.

1.7 Research Contributions

Much research has been done in the area of sign language recognition. The work done ranges from isolated sign recognition including fingerspelling to full continuous sentence recognition. Most of the research done recognized only whole signs and made no attempt to decompose sign and build systems around its constituent parts. Although systems built in this way were very promising and achieved good recognition and translation results, they generally did not scale well to vocabulary extensions; retraining was necessary whenever new signs were added to the vocabulary.

The work done tries to address this issue of scalability. The salient point here is that the number of subcomponents found in existing signs and yet to be discovered sign is finite. Systems built with this premise are not likely to suffer the same limitations as systems which consider whole signs as the basic recognition unit. Our work contributes to gesture recognition research by showing that it is indeed possible to build gesture recognition systems using cheremes as the basic recognition unit. We break down signs into their components and analyse them.Using a statistical modeling technique, we build a small recognition system around this paradigm of cheremes. Our empirical results show that our approach achieves gesture recognition performances that are comparable to those achieved by systems built around whole signs. Some signs often have a gesture component that is repeated. Our system is able to detect such repetitions no duplication of models is thus necessary for each repetition.

We envision that our work done here will give new impetus to future research and development using the chereme paradigm for gesture recognition and drive forward the development of scalable systems without jeopardizing performance and thus bring us closer to the development of a full,

13

real-time, scalable sign language recognition and translation system.

1.8 Thesis Outline

In Section 2, we discuss related research done on gesture recognition. We discuss the family of gestures and place dynamic manual gestures found in sign languages in this context. We explain the difference between manual and non-manual gestures. We look at the beginnings of sign subcomponent research by discussing Stokoe's model and Pose-Movement model. Both models are based on subcomponents of signs but take different approaches. We discuss the phonological approach to sign language modeling. For completeness, we briefly discuss other approaches to sign language recognition. In Section 3, we give a detailed presentation of the image processing and feature extraction processes and algorithms that we implemented. This is followed by an introduction to the theory and application of hidden Markov models, in Section 4, in conjunction with Gaussian mixture models. We discuss the three Markov assumptions together with the problems of evaluation, decoding and recognition and give some examples of HMM use. In Section 5, we discuss our experiments and present our results we achieved. Concluding remarks and possible directions for future research are presented in Section 6.

14
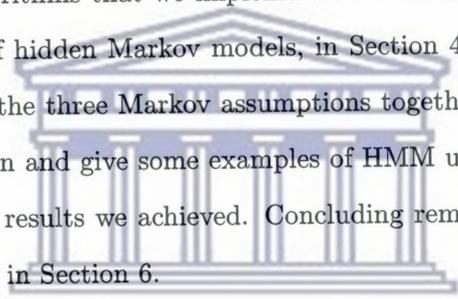
2 Related Work

2.1 A Taxonomy of Gestures

Gestures form the large family of body movements that serve to convey meaning themselves or reinforce the thoughts of a speaker [30] [44]. We look at the different classes of gestures to identify an appropriate context for sign languages. Figure 1 below shows the gesture class hierarchy:
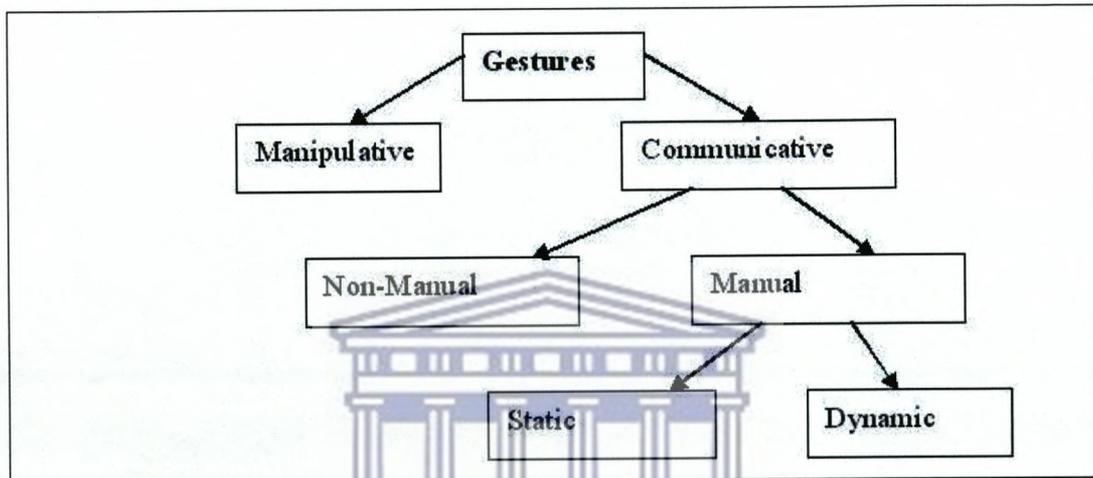
Figure 1: A taxonomy of the gesture classes is used to find the context of sign gestures [35].

Gestures can be categorized as follows:

- Manipulative gestures are gestures that are used to move objects and they convey no direct meaning in conversations [22].

- Communicative gestures are gestures that are used to convey meaning in conversation; for instance, a "shrugging" of the shoulders may mean that the speaker does not know something [29] [36] [44].

- Non-manual gestures are those gestures that do not involve the hands, for example nodding of the head.

- Manual gestures are gestures involving only the hands such as finger spelling in sign language.

We consider only two manual movement classes, namely static and dynamic. Static gestures are those gestures where no perceived change takes place for a time interval whereas dynamic move-

15

ments denote gestures where the hands are in motion. We can distinguish three types of dynamic manual movements in South African Sign Language (SASL) as follows:

- Global movement: SASL is made up of global movements where the start and end location of a signer's hands in sign space are different. Typical movements include linear movement from side to side, up and down, toward and away from the signer, and circular movement along these three axes [20] [9] [12].

- Local movement or secondary movement such as wiggling the fingers accompanies this global movement.

- Epenthesis movement is the movement that is not part of any sign; it is normally found at the beginning of a sign or between two signs [7].

2.2 Stokoe Model

We discuss the Stokoe model here mainly because it was the first model to suggest that building recognition systems around sign subcomponents was indeed possible [39][38]. During his investigation, Stokoe realised that signs are composed of smaller atomic units. This was contrary to popular belief at that time because signs were considered to be entities that could not be decomposed any further; hence, they had to be analysed as a whole unit. He used this finding to devise a transcription system based on the constituent components of a sign and termed these components units *cheremes* [1] and equated them with the phonemes of spoken languages. He defined three parameters that describe cheremes: (1) tab or tabula refers to the location where the sign is made - the forehead, the chin, or the trunk of the body, (2) dez or designator refers to the hand shape used to make the sign and (3) sig or signation refers to the motion involved in executing a sign. Battison added a fourth chereme parameter, ori or orientation , to refer to the orientation of hand(s) while making a sign [38].

The fundamental weakness in Stokoe's sign model is that it assumes that the tab, dez, and sig contrast only simultaneously. In other words, it completely ignores the sequential processes in

---

[1]Cheremes means 'handy' in the Greek language and used to represent non-semantic units of a gesture.

16

sign language; it considers variations in the sequence of these three parameters within a sign to be insignificant [11]. Stokoe's model cannot distinguish between gesture components with a single movement, and gesture components with repeated movement, with all other characteristics being equal. The work on sign language animation by Zhao et al suggests that much can be learnt from looking past Stokoe's model and considering sequentiality [24]. Thus, sequentiality in sign language needs to be handled at the modeling level instead of the recognition level. More sophisticated models have been built that have superseded Stokoe's system.

Liang and Ouhyoung demonstrate that fairly good recognition results can be achieved by using Stokoe's model to detect handshape, position and orientation aspects of the running signs. They used hidden Markov models (HMMs) for continuous recognition of Taiwanese Sign Language with a vocabulary of 250 signs, which they extracted from a Cyberglove in conjunction with a magnetic three dimensional tracker [32].

2.3   The Pose-Movement Model

We discuss the Movement-Hold model for modeling sign language since we follow the same basic ideas in this work . Vogler et al were able to achieve a recognition rate of 91.82% when recognizing cheremes and following the teachings of the Movement-Hold model [11]. It is an example of a segmental model in which each sign is broken down into a series of segments. The two major segments in this model are movements and holds or motion and pose elements. Movement is defined as those segments during which some aspect of the sign's configuration changes, such as a change in hand shape, a hand movement, or a change in hand orientation. Holds or poses are defined as those segments during which all aspects of the sign's configuration remain unchanged, i.e. the hands remain stationary for a brief period. Figure 2 illustrates this idea on the following page

17
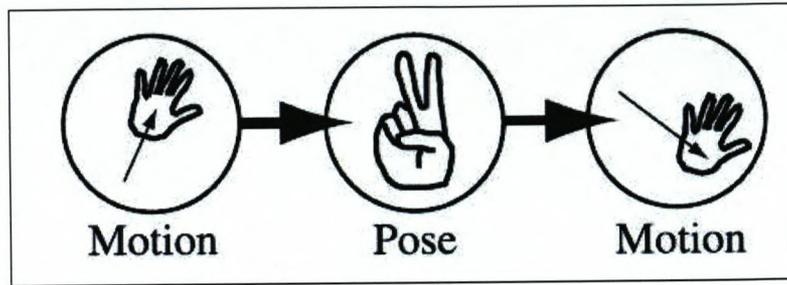
Figure 2: The structure of sign is considered to be a sequence of motion and pose units [11].

The model is defined as follows:

1. A pose is a static subunit of a gesture composed of three simultaneous and inseparable components represented by vector P = [hand shape, palm orientation, hand location]. The static subunit occurs at the beginning and at the end of a gesture.

2. Movement is a dynamic subunit of a gesture composed of velocity and direction of the trajectory described by hands as they travel between successive poses. M=[direction, velocity].

3. A manual gesture is a sequence P-M-P of poses(P) and movements(M).

4. Signing space refers to the physical location where signing takes place. This space is located in front of the signer and is limited by a cube bounding the head, back, shoulders and waist.

2.4   3D versus 2D Modeling

There are typically two approaches to gesture modeling for sign language recognition: The first approach consist of modeling human hand movement in 3D for recognition. Kim and Waldron use a PinchGlove to obtain a sequence of 3D positions of a hand trajectory [27]. They argue that the direct use of 3D data provides more naturalness in generating gestures and helps prevent performance degradation when the trajectory data is projected onto a 2D plane. Vogler and Metaxas show that 3D features outperformed 2D features in recognition performance although they employ vision methods to segment the data [7] [8]. Nam et al note that it is difficult to analyse 3D trajectory data because of the following complexities involved [45]:

18

- Temporal and spatial variances: The changes in speed with which signers perform gestures differ from signer to signer and even vary for each individual as do the manner in which signers perform the rotations and translations of their hands and arms.

- Start/end points: There are no explicit indications of starting and ending of gestures.

- Segmentation: The repetition and connectivity of gesture patterns add difficulties because the recognition process has to deal with segmentation.

- Multiple attributes: In addition to hand movement, the gesture recognition process also has to simultaneously consider other aspects such as hand postures, changes in orientation and the region in sign space where the gesture is performed.

The other approach is view based, where we work with 2D image data obtained from a camera. Grobel and Assam use HMMs to recognize isolated signs with 91.3% accuracy out of a 262-sign vocabulary [23]. They extract features from video recordings of signers wearing coloured gloves. Starner and Pentland use a view-based approach with a single camera to extract two-dimensional features for training and testing hidden Markov models [41]. They a train on a 40-word vocabulary and a strongly constrained sentence structure. They treat signs as whole units and make no attempts to break them down into smaller components. Their results indicate that comparable recognition rates can be achieved without the cost of three-dimensional approaches. Yanghee Nam et al demonstrate how three-dimensional hand movement can be reduced to two dimensions by using plane fitting [45]. Figure 3 shows some of the relevant 3D attributes. The reduction of 3D to 2D using plane fitting is shown in Figure 4; the hand movement path is a three-dimensional attribute that possesses a high degree of freedom in the rotational and translational aspects. They employ the chain encoding scheme for describing the hand movement path to eliminate the translational variance; to achieve the rotational invariance, they reduce the 3D data to two dimensiona. This approach reduces the complexity compared to 3D approaches [45].

19
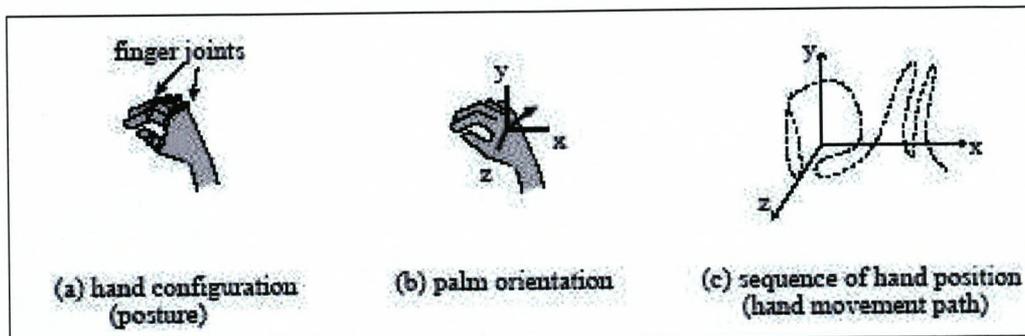
Figure 3: 3D hand gesture attributes used by Nam to capture sign details. The hand posture and palm orientation were tracked and changes recorded in 3D [45].
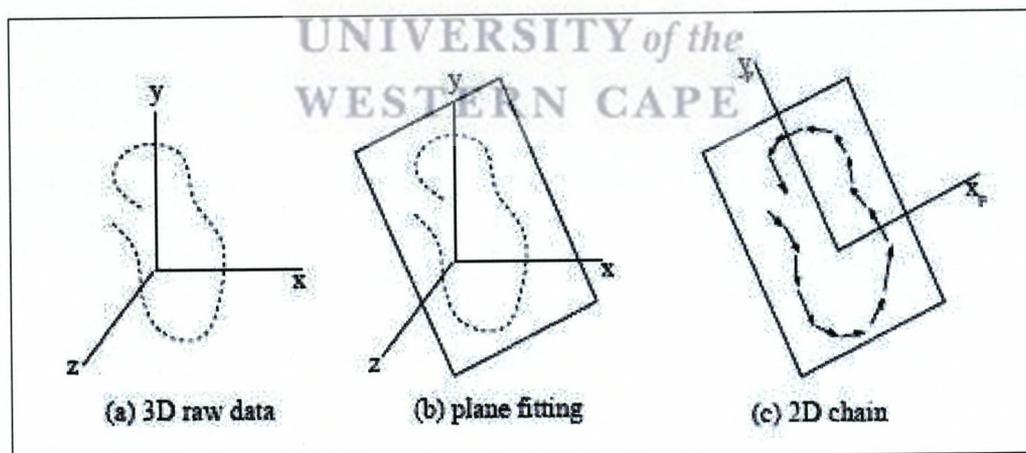

Figure 4: The 3D raw data is fitted with a plane that best describe the data. The plane characteristics are then used as data for further analysis [45].

2.5   Phonological modeling and Hidden Markov Models

We notice that sign language closely follows the phonological model of spoken language. Spoken language is seen as consisting of basic units of phonemes. This means that changing or varying a phoneme in a single utterance alters the sound and may even change the meaning of that word. The phonemes themselves appear sequentially; thus, hidden Markov models (HMMs) are an appropriate tool for modeling spoken language; individuals models for each phoneme are trained and chained together for recognition at a higher level.

We argue that it is sensible to break down signs into smaller units which can be modeled separately [18] [11]. There are statistical variations in the ways two identical signs are performed; our recognition system must be able to handle these variations. A HMM is a state-based statistical model especially well-suited for modeling time-varying signals [5]. They have been used with great success for speech recognition and, more recently, for gesture and sign language recognition [21].

Sagawa et al present a format to describe sign language gestures and a method to recognise the meaning of the gestures based on the components of the gestures [18]. They call the gesture components *cheremes* and the components used to extract meaning and context *morphemes*. Their system recognises cheremes which in turn are then used as building blocks to recognize and isolate much more complex structure of morphemes. They classify cheremes according to hand shape, direction of the hand and the type of motion such as straight, arc etc. They use glove-based input devices to capture gestures and their descriptions consisting of some 56 attribute values. They collected 56 pieces of data such as the bending of fingers, the position and direction of the hands. The data was obtained 30 times per second as an input sign language pattern. Recognized cheremes are represented as combinations of a starting time, an ending time, an estimation value, a type of chereme and its attributes. The estimation value is calculated based on the difference between the parameters of the recognized chereme and the basis value determined in advance. They distinguish between static and dynamic cheremes and recognize 14 different types of cheremes. For the recognition of static cheremes, they use classification by discrimination function, classification

21

using vector quantization and a rule-based method. For dynamic pattern recognition, they use a pattern matching process and HMMs. Their chereme-based system achieves a recognition rate of 97.6% on a limited set consisting of 60 morphemes.

Nam and Wohn introduce the concept of movement primes, which make up sequences of more complex movements [45]. They identify three attributes, namely hand movement as the primary attribute, hand configuration and palm orientation. They then define the hand gesture as the across-time behaviour of the parallel observation of these three attributes. These three major attributes are processed in parallel and independently, followed by the inter-attribute communication for finding the proper interpretation of each hand gesture. They define the prime gesture as the unit of the hand gesture during which no significant changes of the posture or orientation are observed; the hand movement pattern is the movement prime. A hand gesture can then be described as a sequence of prime gestures. Similarly, the movement pattern of a gesture can be described in terms of a sequence of one or more movement primes and the junctions connecting them. They use hidden Markov models to recognize movement patterns since they are able to model spatial and time-scale variances. HMMs are trained for the movement primes and their junction patterns. Null transitions are used for the start node and final node of each movement prime so that there is no transition penalty. Each prime is fitted against its own plane with the result that a single gesture may contain many planes for its constituent prime gestures. The other attributes are used to decide when to fit the data. They use 200 examples for training and 100 for testing and achieve a recognition rate of 80%.

Vogler and Metaxas make use of cheremes (they use the word 'phoneme' instead of 'chereme') in sign language in pursuit of scalability in designing systems [11]. They do not make any distinction between whole signs and epenthesis movements. Instead, the latter movement is dealt with as just another sign. They follow the Movement-Hold model developed by Lindell. The only deviation they make is that their transcriptions contain descriptions of the movements found in the sign. The authors differentiate between local and global features. Local features are the position of the hands in the signing space and the velocity of the hands; the global features are the direction or

22

trajectory of the global movement. To measure the global movement, they fit the signal to a line or plane within a specific time interval. They compute this measure by estimating the covariance matrix over the points in the time interval and by taking its eigenvalues. If the largest eigenvalue is significantly larger then the other two eigenvalues, the signal fits a line well. If the eigenvalues have nearly equal values and are significantly larger than the smallest eigenvalue, then the signal fits the plane well. For their experiments, they collect 499 different sentences of differents length with 1610 signs. They make use of a magnetic tracking system to obtain three-dimensional positions and orientations of the hands at 60 frames per second. They split the 499 sentences into 400 training examples with 1292 signs and 99 test examples with 318 signs. At the word level, they achieve a recognition accuracy of 92%. At the chereme level, their system has a recognition performance of 88% and 92% using local and global input features, respectively. Clearly, it is possible for chereme-based HMMs to achieve sign language recognition performance comparable to sign-based HMMs.

2.6 Other Approaches to Sign Recognition

Bauer and Kraiss argue that following a phonetic acoustic model is not suitable for sign language recognition since some cheremes are performed simultaneously while HMM modeling requires a sequential order of models [5]. The existence of simultaneous processes in sign language fundamentally alters the nature of the recognition problem, and makes sign language recognition much harder than speech recognition. The reason is that, given a signal representing an utterance, simultaneous processes are much more difficult to model in terms of their individual contributions than sequential processes, without also triggering a combinatorial explosion of all these contributions. Also, no unified lexicon of transcription exists for sign language to date. Instead they make use of the fenomic model that is completely self-organised from the data itself. Since the fenomic model does not presuppose any phonetic concepts, its makes it an good approach for sign language modeling. They make use of a training bootstrap algorithm to partition signs, thereby requiring no transcriptions or defined vocabularies. They achieve a recognition accuracy of 92.5% for 100 different signs used in the bootstrap process. For the second experiment, they recognize 50 signs with an accuracy of 81%. In extensions of their work, they break down signs into smaller

23

units. These units are unlike phonemes, however, because they are determined computationally via clustering, instead of being determined linguistically. Using this computational non-linguistic appproach for the definition of sign components, they achieve an accuracy of 92.5% in isolated sign language recognition experiments.

Wang et al describe a large-scale HMM-based isolated recognition system for Chinese Sign Language with a very impressive vocabulary size of more than 5000 signs [13]. They use some tricks from speech recognition, such as clustering Gaussian probabilities, and fast matching, to achieve real-time recognition and recognition rates of 95%. These results show that some of the fast speech recognition algorithms are directly applicable to sign language recognition. They collect the data with magnetic tracking systems and cybergloves. Fang et al propose an approach to signer-independent continuous recognition based on an integration of simple recurrent networks (SRNs) and HMMs. They used the SRNs to segment the continuous sentences into individual signs. They achieve recognition rates of 92% over a test data set of 100 sentences with a 208-sign vocabulary. Erensthteyn et al use neural networks to recognize fingerspelling. Waldron and Kim also use neural networks to recognize a small set of isolated signs using Stokoe's transcription system to separate the handshape, orientation, and movement aspects of the signs.

2.7 Temporal Segmentation

The purpose of temporal segmentation is to split the continuous sequence into movement cheremes [13] [20]. The movement cheremes exhibit basic movements whose execution is consistent and easily characterised by a simple trajectory. The segmentation involves searching for natural inconsistent points within the whole sign performance. A change in the type of human movement usually causes dips in velocity or abrupt variations in moving directions. We exploit this by finding the local minima of velocity and the local maxima of changing direction. The minima and maxima below and above certain thresholds, respectively, are selected as segment points. Liang et al show that the explicit segmentation of the data stream based on dicontinuities in the movements yields results that are comparable to other work where HMMs are used to segment the input stream implicitly [32]. They integrate the handshape, position, orientation, and movement aspects through stochastic parsing

24

and a dynamic programming algorithm at a higher level than the HMMs.

UNIVERSITY *of the*
WESTERN CAPE

3 Video Image Processing and Feature Extraction

In this section, we describe the image preprocessing and feature extraction used to achieve our goals of chereme recognition. Much effort is spent here to ensure that reasonable feature vector values will be achieved for future use. We explain in detail how the feature vector is calculated that is to be used for training hidden Markov models.

3.1 Image Preprocessing

Figure 5 shows the image preprocessing process:
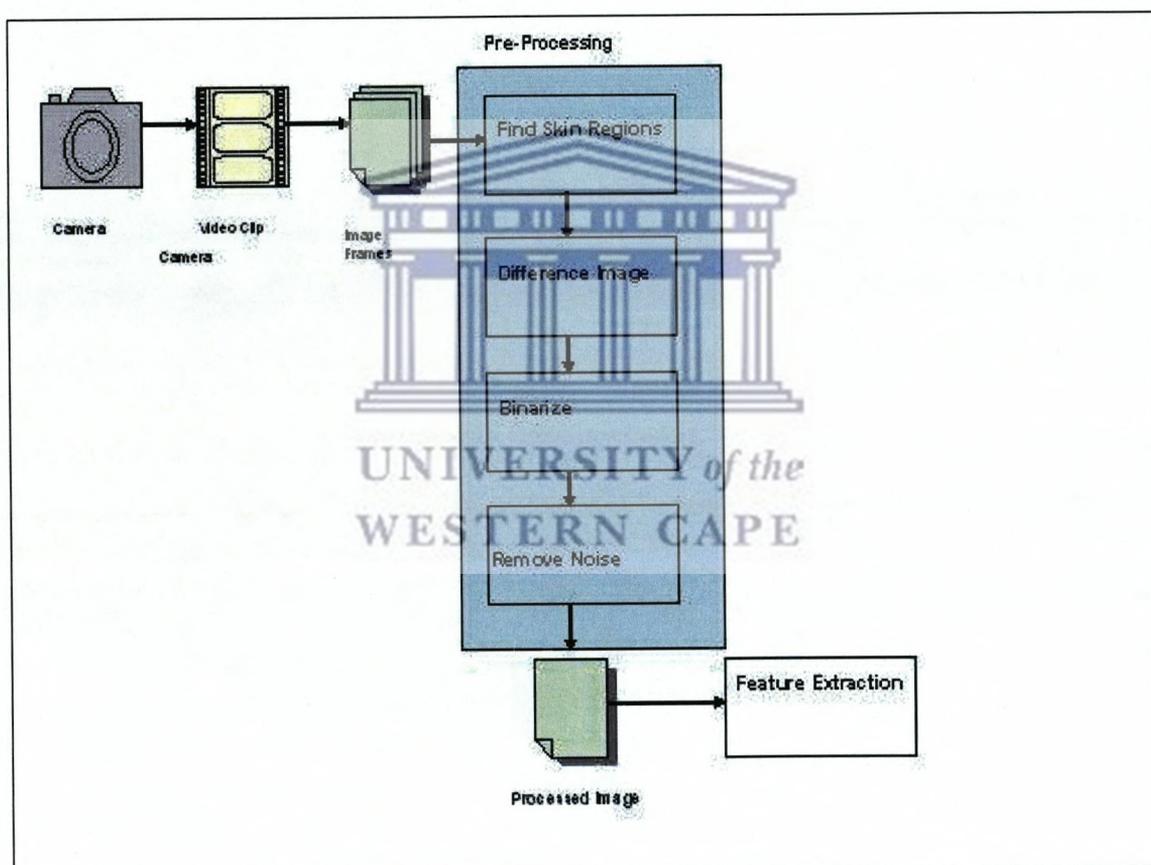
Figure 5: The image preprocessing process that takes place to produce feature vectors

The image preprocessing process is as follows:

1. The first step is to split the video file into image frames for processing.

26

2. For each image frame we identify the skin region through skin segmentation [19] [1] [2].

3. The identified skin regions are used as the regions of interest (ROI).

4. A common ROI is built for the sign sequence.

5. We calculate the difference image for consecutive frames.

6. We make use of the ROI built from the previous steps to remove noise from the difference image.

7. We calculate the feature vector for each difference image.

Since the region of interest (ROI) changes from frame to frame, we need to find the ROI for each image frame and then build the final ROI by joining all the individual ROI's. We first segment the image to find skin and non-skin pixels using Bayesian techniques. Thereafter, we are able to identify skin regions and hence the ROI of the image frame. We use skin segmentation to divide the colour image into two segments namely, skin and non-skin. The colour space YCBCr is well suited for this task [15] [4]. The segmentation process allows us to gather two pixel distributions for each channel. We make use of neighbouring pixel information to force our algorithm to create the regions. The Cb and Cr channels for skin and non-skin are found to be normally distributed. A mean μ and variance σ is calculated for each channel. The Y channel is uniformly distributed. The uniformly distributed values have a constant probability of $P = -ln(1/255)$, while the normal distribution follows the probability function:

$$P(y) = Ce^{-\frac{(y-\mu)^2}{2\sigma^2}} \tag{1}$$

where C is a constant. Using this knowledge, we proceed to do skin segmentation on the image frames. Once we have identified the skin regions, we have defined our region of interest for that image frame. Individual ROIs of each frame are used to build the common ROI of the image sequence which spans the entire image set. The common ROI is considered to be our signing space for that sign sequence.

27

Then, for each $r_i(x,y)$ where r_i is the defined region of interest for frame i, we have $R(x,y) = r_1(x,y) \cup r_2(x,y) \cup ... \cup r_n(x,y)$ for n image frames in the sequence, i.e. $R(x,y)$ is the union of individual frames $r_i(x,y)$. Figure 6 shows the result of performing the skin segmentation algorithm on one of our images.

(a) Before Skin Segmentation   (b) After Skin Segmentation
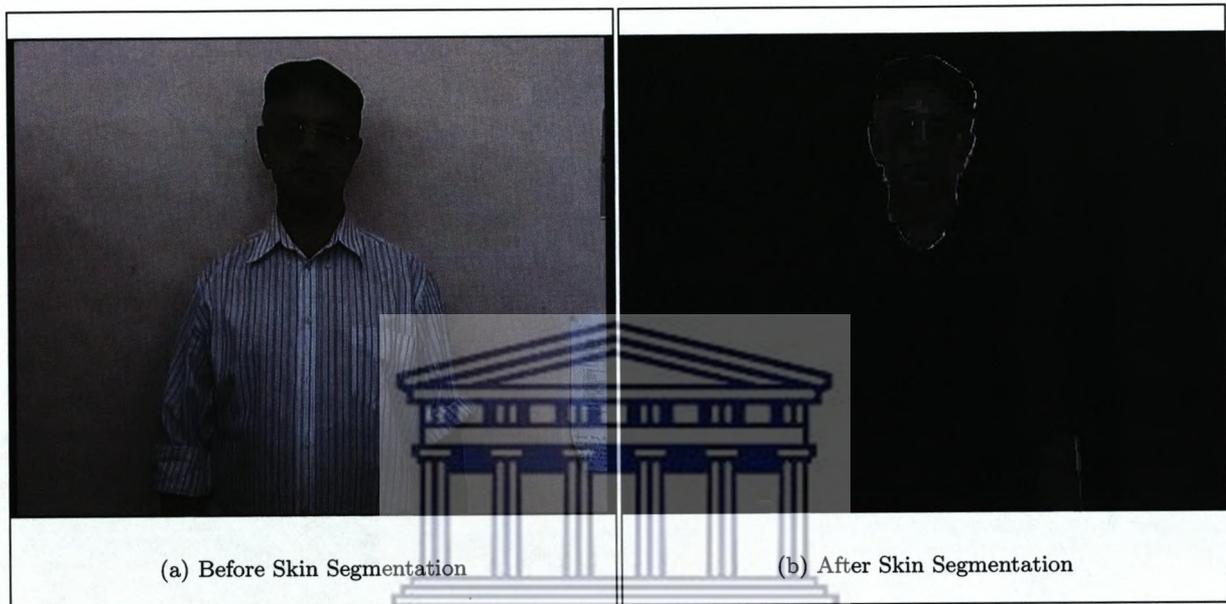
Figure 6: An image undergoes skin segmentation. The resulting image has less detail , with the background and signer's clothing information removed.

Image differencing allows us to spot differences between two consecutive frames [17]. The differences between the two frames gives us an idea of what has changed during that time interval. This change represents the motion that took place and calculations can be performed on these resultant pixel values. Building the difference image is an efficient and effective method for the extraction of motion or moving objects in slow changing environment. For our experiments, the image backgrounds are fairly static and non-complex; thus, the image differencing method works very well. The difference image sequence $I'_d(x,y)$ is built by subtracting the pixel values at equal positions (x,y) of every second frame from the previous frame in the original image sequence:

$$I'_d(x,y) = I_t(x,y) - I_{t-1}(x,y) \tag{2}$$

28

where $I_t(x, y)$ is the gray pixel value at position (x, y) for an image frame at time t.

Following this operation, we need to clean up the difference images by removing noise and movement outside our region of interest. $I'_d(x, y)$ may consist of positive and negative values. We apply three processing steps to reduce noise in the difference image:

1. Thresholding on pixel values: We use thresholding to remove small gray pixel values that represent motion noise in the difference image. Every pixel with an absolute value smaller than a threshold T is set to zero [16] [40].

$$I_d(x, y) = \begin{cases} 0 & |I'_d(x, y)| < T \\ I'_d(x, y) & |I'_d(x, y)| >= T \end{cases} \tag{3}$$

The thresholded absolute gray values $I_d(x, y)$ represent the gray value for each spatial position (x, y) of the difference image.

2. Focus on region of interest: We ignore all values from the difference image that are outside the common region of interest $R(x, y)$.

$$I_d(x, y) = \begin{cases} 0 & |I'_d(x, y)| \notin R(x, y) \\ I'_d(x, y) & |I'_d(x, y)| \in R(x, y) \end{cases} \tag{4}$$

where $R(x, y)$ is the common skin region.

3. Image smoothing neighbourhood operations: We take the average pixel values of the neighbourhood pixels in a 4x4 matrix and apply it to each pixel in the image. The result is a smooth image [34].

We are now ready to proceed with our feature vector calculations.

3.2   Feature Vector Calculation

A difference image can be interpreted as a distribution of the motion over the image space in x and y direction with the weights $I'_d(x, y)$. Each distribution is characteristic for a specific motion,

29

and so describes an action. Characterizing this distribution with certain features results in a good representation for the current motion in the difference image.

Calculating the center of mass $m'(t) = [m'_x(t), m'_y(t)]^T$ delivers the center of motion:

$$m'_x(t) = \frac{\sum_{(x,y)\in R_i} x|I_d(x,y,t)|}{\sum_{(x,y)\in R_i} |I_d(x,y,t)|} \tag{5}$$

$$m'_y(t) = \frac{\sum_{(x,y)\in R_i} y|I_d(x,y,t)|}{\sum_{(x,y)\in R_i} |I_d(x,y,t)|} \tag{6}$$

Since the features should be independent of the location of a signer, the center of motion relative to the signer's center of mass $m(t) = [m_x(t), m_y(t)] = [m'_x(t) - p_x(t), m'_y(t)]^T$ is used for further processing steps. A relativization of $m'(t)$ is not necessary because signers are located at the same height in the image. To consider the changes in the direction of a movement, the relative change of the center of mass $\Delta m_x(t) = m_x(t) - m_x(t-1)$ and $\Delta m_y(t) = m_y(t) - m_y(t-1)$ is added as a component to the feature vector.

Additionally, we use the mean absolute deviation of a pixel (x,y) relative to the center of motion $\sigma(t) = [\sigma_x(t), \sigma_y(t)]^T$ to describe motion:

$$\sigma_x(t) = \frac{\sum_{(x,y)\in R_i} x|I_d(x,y,t)|(x - m_x(t))}{\sum_{(x,y)\in R_i} |I_d(x,y,t)|} \tag{7}$$

$$\sigma_y(t) = \frac{\sum_{(x,y)\in R_i} x|I_d(x,y,t)|(x - m_x(t))}{\sum_{(x,y)\in R_i} |I_d(x,y,t)|} \tag{8}$$

This feature allows us to distinguish between actions where large parts of the body are in motion (e.g. get-up) from actions concentrated in a smaller area, where only small parts of the body move (e.g. nodding). This feature is also referred to as *wideness of motion*.

Another important feature describing motion is the *intensity of motion* $i(t)$; it simply the average absolute height of the motion distribution, which can be expressed as:

30

$$i(t) = \frac{\sum_{(x,y)\in R_i} |I_d(x,y,t)|}{\sum_{(x,y)\in R_i} 1} \tag{9}$$

A large value of $i(t)$ represents a very intensive motion of parts of the body, and a small value characterizes an almost stationary image. The complexity and dimension of the high dimensional action space is significantly reduced by extracting this 7-dimensional feature vector

$$\vec{x_t} = [m_x, m_y, \Delta m_x, \Delta m_y, \sigma_x, \sigma_y, i]^T \tag{10}$$

while preserving the characteristics of the currently observed motion. This motion vector is derived for every second frame so that a vector sequence $\vec{X_n} = [\vec{x_1}, ..., \vec{x_n}]^T$ arises, where each vector carries important information about the current motion, and thus the entire sequence contains the information about the performed actions.

4 Hidden Markov Models and Gaussian Mixtures

4.1 Markov Process

Often, we are interested in finding patterns in spatio-temporal signals, i.e. signals which appear over a space of time. Such patterns occur in many areas: the pattern of commands someone uses in instructing a computer, sequences of words in sentences, the sequence of phonemes in spoken words, the fluctuations of stock prices, recording of seismic events in mines and the purchase behavior of consumers are some examples; any system that produces outputs as part of underlying process can produce useful patterns - or timeseries - that we may want to model, distinguish or predict. Such patterns may generated deterministically or non-deterministically.

To solve non-deterministic pattern recognition problems - for example, weather prediction - it is often useful to assume that we can identify discrete system states in the underlying processes and that the the systems' current state depends only upon previous states. This is called the Markov assumption and greatly simplifies the task of system identification. Although this is a gross simplification and much important information may be lost because of it, in practice this assumption often works well, i.e. we can build good models of unknown underlying processes based on this assumption that lead to good performance.

A Markov process is a process which moves from state to state depending only on the previous n states. The process is called an order n model where n is the number of states affecting the choice of next state. The simplest Markov process is a first-order process, where the current state only depends on the previous state. This is not the same as a deterministic system, since we expect the choice to be made probabilistically, i.e. a state transition is performed with some probability. For a first-order process with M states, there are M^2 possible transitions between states since it is possible for any one state to follow another. Associated with each transition is a probability called the *state transition probability*; this is the probability of moving from one state to another. These M^2 probabilities may be collected together in an obvious way into a *state transition matrix*. These probabilities do not vary in time - this is an important (if often unrealistic) assumption.

32

We can now define a first-order Markov process as follows:

- states: The states of the model.

- π vector: Defining the probability of the system being in each of the states at the time of initialization.

- state transition matrix: The (fixed) probability of moving from one state to the next.

Any system that can be described in this manner is a Markov process.

4.2 Hidden Markov Model

A realistic, non-trivial problem is speech recognition. The sound that we hear is the product of the vocal chords, size of throat, position of tongue and several other factors. Each of these factors interact to produce the utterance of a word, and the utterances that a speech recognition system detects are the changing sounds generated from the internal physical changes in the person speaking.

In such systems, there are some processes where an observed output sequence can be probabalistically related to an underlying Markov process. In addition to hearing the sound of an utterance, we may also be able to observe a speaker's lips; however, since we generally do not have access to the state of vocal cords, tongue position, etc. - their states are hidden from the observer - a model for the recognition of utterance must also accommodate states that are non-observable, i.e. the the number of observable states may be different to the number of actual states. This gives rise to the the use of hidden Markov models (HMMs).

A formal definition of HMMs consists of the following elements:

1. The number N of states of the model.

2. The number M of observation symbols in the alphabet. If the observations are continuous then M is infinite.

33

3. A set of state transition probabilities $A = \{a_{ij}\}$ where $a_{ij} = P(q_{t+1} = j | q_t = i), 1 \leq i, j \leq N$ and q_t denotes the current state.

   Transition probabilities must satisfy the normal stochastic constraints $a_{ij} \geq 0, 1 \leq i, j \leq N$ and $\Sigma_{j=1}^{N} a_{ij} = 1, 1 \leq i \leq N$

4. An observation $B = \{b_j(k)\}$ probability distribution in each of the states with $b_j(k) = P(o_t = v_k | q_t = j), 1 \leq j \leq N, 1 \leq k \leq M$

   where v_k denotes the kth observation symbol in the alphabet, and o_t the current vector. The following stochastic constraints must be satisfied: $b_j(k) \geq 0, 1 \leq j \leq N, 1 \leq k \leq M$ and $\Sigma_{k=1}^{M} b_j(k) = 1, 1 \leq j \leq N$

   If the observations are continuous, then we have to use a continuous probability density function instead of a set of discrete probabilities. In this case, we specify the parameters of the probability density function. Usually, the probability density is approximated by a weighted sum of Gaussian distributions for each state of an HMM:

   $b_j(o_t) = \Sigma_{m=1}^{M} c_{jm} N(\mu_{jm}, \Sigma_{jm}, o_t)$ where c_{jm} denote weighting coefficients, μ_{jm} are the mean vectors, and Σ_{jm} are the covariance matrices. The weighting coefficients c_{jm} must satisfy the stochastic constraints $c_{jm} \geq 0, 1 \leq j \leq N, 1 \leq m \leq M$ and $\Sigma_{m=1}^{M} c_{jm} = 1, 1 \leq j \leq N$.

5. The initial state distribution $\pi = \{\pi_i\}$ where $\pi_i = p\{q_1 = i\}, 1 \leq i \leq N$.

Therefore, we can use the compact notation $\lambda = (A, B, \pi)$ to denote HMMs with discrete probabilities while $\lambda = (A, c_{jm}, \mu_{jm}, \Sigma_{jm}, \pi)$ is used to denote HMMs with continuous observation density functions.

4.3 Simplifying Assumptions in the Theory of HMMs

For the sake of mathematical and computational tractability, the following assumptions are often made in the theory of HMMs:

- First-order assumption: As given in the definition of HMMs, transition probabilities are defined as $a_{ij} = P(q_{t+1} = j | q_t = i)$. In other words, it is assumed that the next state is dependent only upon the current state. This is called the first-order assumption and the

34

resulting model thus becomes a first-order HMM. However, the next state may depend on past k states and it is possible to obtain such a model, called an kth-order HMM by defining the transition probabilities as follows:

$$a_{i_1 i_2 \ldots i_k j} = P(q_{t+1} = j | q_t = i_1, q_{t-1} = i_2, \ldots, q_{t-k+1} = i_k), 1 \leq i_1, i_2, \ldots, i_k, j \leq N.$$

Clearly, a higher order HMM will have a higher complexity. Even though first-order HMMs are most commonly used, some attempts have been made to use the higher-order HMMs as well.

- Stationarity assumption: Here it is assumed that state transition probabilities are independent of the actual time at which the transitions takes place. Mathematically, $P(q_{t_1+1} = j | q_{t_1} = i) = P(q_{t_2+1} = j | q_{t_2} = i)$ for any t_1 and t_2.

- Output independence assumption: This is the assumption that current output or observation is statistically independent from all previous outputs. We can formulate this assumption mathematically, by considering a sequence of observations $O = o_1, o_2, \ldots, o_T$. Then, according to the first-order assumption for an HMM, $P(O | q_1, q_2, \ldots, q_T, \lambda) = \Pi_{t=1}^{T} p(o_t | q_t, \lambda)$. Unlike the previous two assumptions,, this assumption has a very limited validity. In some cases, this assumption may not be valid at all and can therefore becomes a severe weakness of the HMMs.

4.4 Basic Problems of HMMs

Once we have an HMM, there are three problems of interest:

1. The Evaluation Problem: Given an HMM and a sequence of observations $O = o_1, o_2, \ldots, o_T$, what is the probability that the observations are generated by the model, $P(O | \lambda)$?

2. The Decoding Problem: Given a model and a sequence of observations $O = o_1, o_2, \ldots, o_T$, what is the most likely state sequence in the model that produced the observations?

3. The Learning Problem Given a model and a sequence of observations $O = o_1, o_2, \ldots, o_T$, how can we adjust the model parameters (A, B, π) in order to maximize $P(O | \lambda)$?

4.4.1 Evaluation Problem and Forward Algorithm

Given a model $\lambda = (A, B, \pi)$ and a sequence of observations $O = o_1, o_2, ..., o_T$, we must calculate $P(O|\lambda)$. This can be done using simple probabilistic arguments. However, the number of operations needed is of the order of N^T. Thus, even for moderate values of T, this computation seems quite inefficient and a more efficient algorithm is desirable. Fortunately, there exists such an algorithm with considerably lower time complexity; it makes use of an auxiliary variable $\alpha_t(i)$, called forward variable. It is defined as the probability of the partial observation sequence $o_1, o_2, ..., o_t$, when it terminates at the state i:

$$\alpha_t(i) = p\{o_1, o_2, ..., o_t, q_t = i|\lambda\} \tag{11}$$

It is easy to see that the following recursive relationship holds:

$$\alpha_{t+1}(j) = b_j(o_{t+1})\Sigma_{i=1}^{N}\alpha_t(i)a_{ij}, 1 \le j \le N, 1 \le t \le T - 1 \tag{12}$$

where $\alpha_1(j) = \pi_j b_j(o_1), 1 \le j \le N$. Using this recursion, we calculate $\alpha_T(i), 1 \le i \le N$ and then the required probability is given by

$$P(O|\lambda) = \Sigma_{i=1}^{N}\alpha_T(i) \tag{13}$$

The complexity of this method, known as the forward algorithm, is proportional to $N^2 T$, which is linear with respect to T; we note that the direct calculation mentioned earlier has exponential time complexity. In a similar way, we define the *backward variable* $\beta_t(i)$ as the probability of the partial observation sequence $o_{t+1}, o_{t+2}, ..., o_T$ given the current state is i:

$$\beta_t(i) = p\{o_{t+1}, o_{t+2}, ..., o_T, q_t = i|\lambda\} \tag{14}$$

As in the case of $\alpha_t(i)$, there is a recursive relationship which allows for an efficient calculation of $\beta_t(i)$:

$$\beta_t(i) = \Sigma_{j=1}^{N}\beta_{t+1}(j)a_{ij}b_j(o_{t+1}), 1 \le i \le N, 1 \le t \le T - 1 \tag{15}$$

36

where $\beta_T(i) = 1, 1 \leq i \leq N$.

Furthermore, we observe that

$$\alpha_t(i)\beta_t(i) = P(O, q_t = i|\lambda), 1 \leq i \leq N, 1 \leq t \leq T \tag{16}$$

Thus, we obtain

$$P(O|\lambda) = \Sigma_{i=1}^{N} P(O, q_t = i|\lambda) = \Sigma_{i=1}^{N} \alpha_t(i)\beta_t(i) \tag{17}$$

4.4.2 The Decoding Problem and the Viterbi Algorithm

Here, we want to find the most likely state sequence for a given sequence of observations $O = o_1, o_2, ..., o_T$ and a model $\lambda = (A, B, \pi)$. The solution to this problem depends on the definition of "most likely state sequence". One approach is to find the most likely state q_t at $t = t$ and to concatenate all such q_t's. Sometimes this method does not result in a physically meaningful state sequence. Another method known as the $Viterbi$ $algorithm$ does not suffer from this problem; it finds the entire state sequence with the maximum likelihood. In order to facilitate the computation we define an auxiliary variable

$$\delta_t(i) = max_{q_1 q_2 ... q_{t-1}} P(q_1, q_2, ..., q_{t-1}, q_t = i, o_1, o_2, ..., o_{t-1}|\lambda) \tag{18}$$

which gives the highest probability of the partial observation sequence and state sequence up to $t = t$ for current state i. It is easy to observe that the following recursive relationship holds:

$$\delta_{t+1}(j) = b_j(o_{t+1})[max_{1 \leq i \leq N} \delta_t(i)a_{ij}], 1 \leq i \leq N, 1 \leq t \leq T-1 \tag{19}$$

where $\delta_1(j) = \pi_j b_j(o_1), 1 \leq j \leq N$.

So, the algorithm for finding the most likely state sequence starts from calculation of $\delta_T(j), 1 \leq j \leq N$ using recursion in Equation 18, while always keeping a pointer to the "winning state" in the maximum finding operation. Finally the state j^* is found for which $j^* = arg$

37

$max_{1 \leq j \leq N} \delta_T(j)$. Starting from this state, the sequence of states is backtracked as the pointer in each state indicates resulting in the required set of states. This algorithm can be interpreted as a search in a graph whose nodes are formed by the states of the HMM in each of the time instants $t, 1 \leq t \leq T$.

4.4.3 The Learning Problem and Baum-Welch Algorithm

The Baum-Welch algorithm iteratively adjusts the parameters of the model $\lambda = (A, B, \pi)$ such that $P(O|\lambda)$ is locally maximized; thus, it solves the HMM parameter estimation problem. First, let us define $\xi_t(i, j)$ as the probability of being in state s_i at time t and state s_j at time $t + 1$, given the model and the observation sequence

$$\xi_t(i, j) = P(q_t = s_i, q(t + 1) = s_j | O, \lambda)$$

and define the variable $\gamma_t(i)$ as the probability of being in state s_i at time t, given the model λ and the observation sequence O, as follows:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i, j)$$

Then, the re-estimation formula for π, A, B becomes:

$$\overline{\pi_i} = expected\ frequency\ in\ state\ s_i\ at\ time(t = 1) = \gamma_1(i),$$

$$\overline{a_{ij}} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)},$$

and

$$\overline{b_j(k)} = \frac{\sum_{t=1,,\ with\ O_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}.$$

The re-estimation formula for π_i, is simply the probability of being in state i at time t. The formula for $\overline{a_{ij}}$ is the ratio of expected number of times making a transition from state i to state j to the expected number of times of making a transition out of state i. The formula for $\overline{b_j(k)}$ is the ratio of the expected number of times of being in state j and observing symbol o_k to the expected number of times of being in state j. If we denote the initial model λ and the re-estimation model by λ'

38

consisting of the parameters estimated above, then it can be shown that either:

1. The initial model λ is a critical point of the likelihood function in which case $\lambda' = \lambda$ or

2. $P(O|\lambda') > P(O|\lambda)$, i.e, we find a better model with higher likelihood of producing the observation sequence $O = o_1, o_2, ..., o_T$.

3. Hence we iteratively compute $P(O|\lambda)$, until $P(O|\lambda)$ is maximized, i.e. until $P(O|\lambda)$ for the current iteration is not higher than the value of the previous iteration.

4.5   Gaussian HMMs

In a continuous HMM, the observation space Y is considered to be infinite, multidimensional and continuous. Therefore, for each state j, it is necessary to be able to compute the probability $b_j(y)$ for each vector y in a continuous space. The simplest state output probability density function b_j is a one-dimensional Gaussian with mean μ_j and covariance C_j.

$$b_j(y) = \Pi_{k=1}^{K} \frac{1}{\sqrt{2\pi C_{jk}}} exp(-\frac{1}{2}\{\frac{y_k - \mu_{jk}}{C_{jk}}\}^2) \tag{20}$$

The probability density function in Equation 19 cannot sufficiently model the variations in empirical data; thus, Gaussian mixtures are typically used to model broad sources of variability [5]. The most popular form of Gaussian mixture is the multiple-component Gaussian mixture. A M-component Gaussian mixture is just a linear combination of M Gaussian probability density functions. The state output probability density function b_j then has the form:

$$b_j(y) = \Sigma_{m=1}^{M} W_m b_j^m(y) \tag{21}$$

where W_m is the weight which describes the relative likelihood of the mixture components being generated from each of the components.

4.6   Applications of Hidden Markov Models

One of the basic premise followed by speech recognition research is that speech itself is composed of smaller contrastive units called phonemes [5]. A phoneme is the smallest unit of sound that

is unique, i.e. distinguishes one word from another for a given language. For example, the words "seat","meat","beat" are different words since the initial sound is a separate phoneme in English. Most sounds, including speech phonemes, are complex waves with a dominant or primary frequency. This is the rate at which vocal cords flap against each other during the production of a voiced phoneme [26]. In Figure 7, we show two wave diagrams for sound produced by a human's vocal cords.


(a) Phoneme (b) Noise
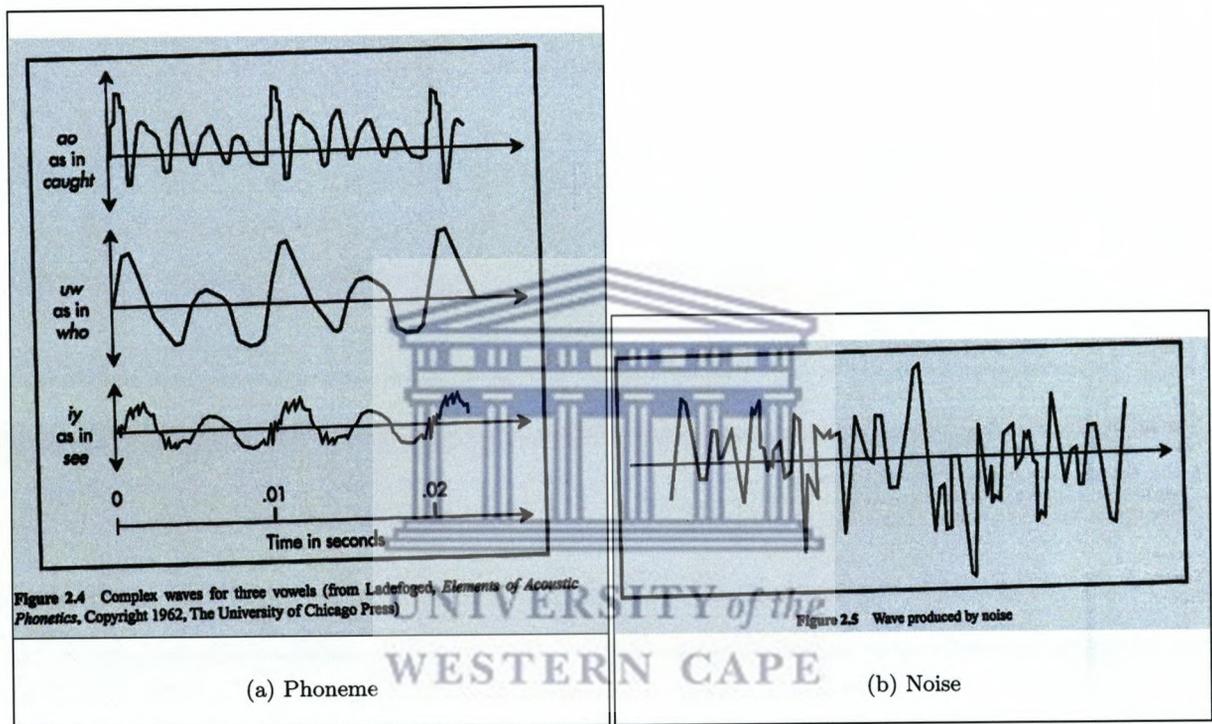
Figure 7: Examples of Complex Waves for Phonemes and Noise. The wave pattern produced by the noise exhibits inconsistent patterns in each cycle [25].

Figure 7(a) shows the waveforms of phonemes found in English words "caught" , "who" and "see", respectively. Figure 7(b) shows the waveform for "noise" for comparative purposes. We observe that the "noise" waveform is erratic and has no clear peaks or troughs. The approach used by speech recognition systems is to build models for and recognize individual phonemes that make up a word. Since the phonemes themselves appear sequentially, hidden Markov models are appropriate tools for modeling speech. One common approach is to build an HMM for each triphone [2]. Figure

---

[2] A triphone consists of a phoneme together with the predecessor and successor phonemes.

40

8 shows a triphone HMM with three states for each of the phonemes [25].
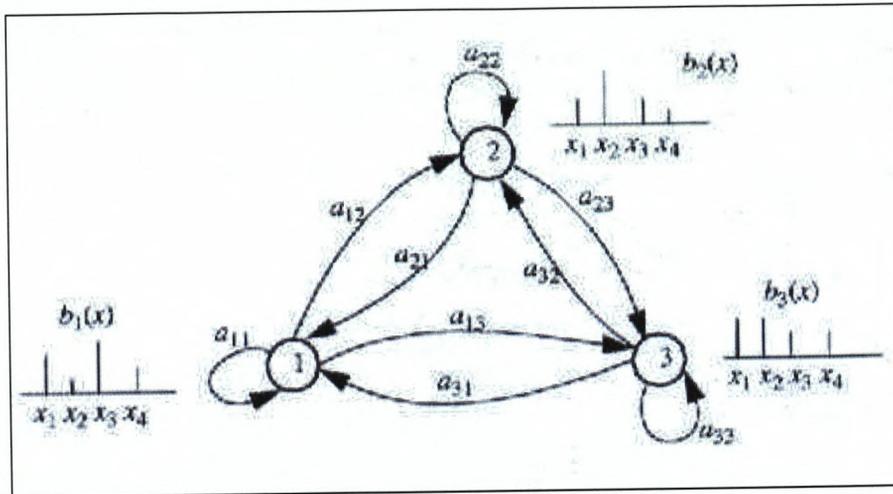
Figure 8: A 3-state circular HMM is used training. Each state represents each phone in the triphone but in no order [25].

Figure 8 shows that a triphone can be modeled by a 3-state circular HMM where each state emits an output probability function assumed to be Gaussian. A recognition system will then consist of a network of HMMs for each triphone. Figure 9 is a possible recognition flow chart adapted from [27].
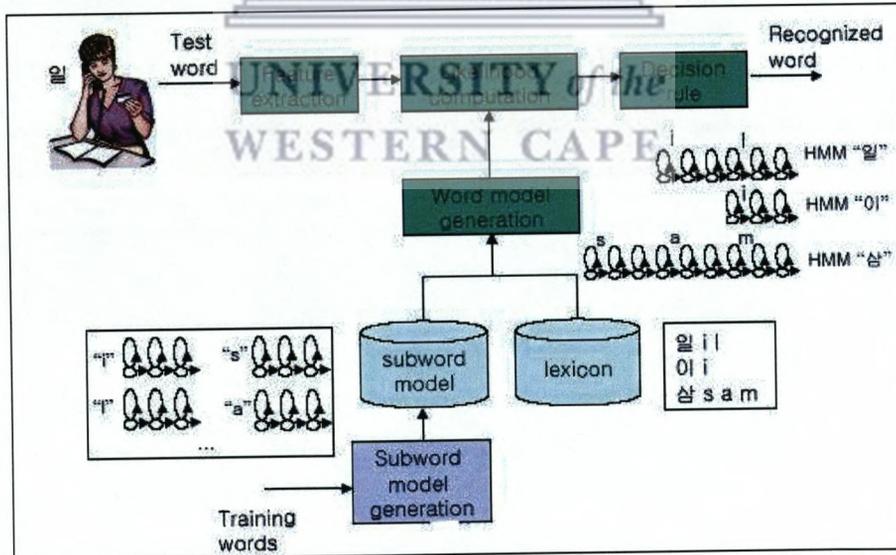

Figure 9: Recognition process flowchart for speech sub-words [25].

We notice in Figure 9 that distinct salient features are extracted to train HMMs for each phoneme waveform. A lexicon is pinned with the trained HMMs to generate word models for recognition

purposes.

4.7 Application to Chereme Recognition

Hidden Markov models were very successfully applied to speech recognition. The steps involved in preparing the HMMs prior to training include the following:

1. A dictionary is defined listing all the legal words that the system recognizes. Sentence construction takes place by selecting one or more words from the dictionary.

2. A grammar is define at the outset specifying the structure of the sentences and the rules to be followed in building sentences.

3. We can choose to manually label input data files or automatically. In either case, a word network is built for training purposes.

In order to make use of HMM for our research, we avoid fundamentally changing the preparatory steps prior to training. Rather, we use the same terminologies and sentence construction methods as proposed above. We define a dictionary of cheremes that we want to recognize and we build a grammar system to specify how the cheremes are pulled together to produce a gesture. It should be pointed out that this process does not in anyway make the recognition results questionable. It is done to facilitate model training without have to rewrite the software. After all, by using native HMM without parallel processes, we assumes a sequential phonological model structure for sign training and recognition. To address the simultaneous aspects of the signing process, models need to be built for each simultaneous process and chained together. We do not deal with that issue here. Our goal is to show that HMM are able to recognise and classify cheremes.

4.8 Gaussian Distributions and Mixtures

The normal (or Gaussian) distribution is one which appears in an incredible variety of statistical applications. A good reason for this is the central limit theorem. This theorem tells us that sums of random variables will, under the appropriate conditions, tend to be approximately normally distributed; this is sometimes true even when these conditions are not met. However, the distributions found for many experimentally generated sets of data still tend to have a bell shaped curve

43

that often looks quite like that of a normal. Even if a distribution is not truly normal, it is still convenient to assume that a normal distribution is a good approximation. In this case, we describe the entire distribution by simply a mean and a variance, two easily computed parameters that all statisticians and experimenters understand and use for comparison. The normal distribution has become a convenient standard for all to use.

Mixture distributions arise in practical problems when the measurements of a random distribution are taken under different conditions. For example, the distribution of heights in a population of adults reflects the mixture of males and females in the population; we find similar distribution in speech because of grouping of speech data from different speakers or accent groups. Mixture models are used in problems of this type, where the population units consist of a number of subpopulations within each of which a relatively simple model applies. Mixtures of Gaussian distributions help solve problems of this type in which a class conditional probability density function is formed from a weighted sum of individual Gaussians [8]. For a Gaussian mixture we have

$$P(x) = \Sigma_{m=1}^{M} c_m N(x, \mu_m, \sigma_m) \tag{22}$$

where M is the number of mixture components, x is the observation vector, μ_m and σ_m are the mean and variance of the sample, respectively and c_m is the mixture weight of each Gaussian component with $\Sigma_{m=1}^{M} c_m = 1$.

Parameter estimation of Gaussian mixtures are based on a general iterative Expectation-Maximisation (EM) algorithm:

1. Find the posterior probability of the mixture component for the current observation.

2. Update the parameters of the Gaussian mixture component.

Using the subscript *old* for the parameters from the previous iteration and *new* for the updated parameters, the details of this procedure are as follows:

1. Initialise the parameters of the mixture model, e.g. different μ_m vectors, $c_1 = c_2 = c_3 = \ldots =$

44

c_M, and $\mu_1 = \mu_2 = \mu_3 = ... = \mu_M$

2. Compute

$$P^{old}(m|x_k) = \frac{c_m P(x_k|m)}{P(x_k)} \tag{23}$$

where $P(m|x_k)$ is the posterior probability of mixture component m being associated with the vector x_k and

$$P(x_k) = \Sigma_{m=1}^{M} c_m P(x_k|m) \tag{24}$$

is the probability density of the vector of the entire mixture distribution.

3. Update the parameters:

$$\mu_m^{new} = \frac{\Sigma_k P^{old}(m|x_k) x_k}{\Sigma_k P^{old}(m|x_k)} \tag{25}$$

$$(\mu_m^{new})^2 = \frac{1}{d} \frac{\Sigma_k P^{old}(m|x_k)||x_k - \mu_m^{new}||}{\Sigma_k P^{old}(m|x_k)} \tag{26}$$

$$c_m^{new} = \frac{1}{n} \Sigma_k P^{old}(m|x_k) \tag{27}$$

4. Calculate the new log likelihood. If the new log likelihood is less than a threshold or $P^{new}(m|x_k) \leq P^{old}(m|x_k)$, then stop, else repeat from step 2.

45

Figure 10 illustrates an example of the training process with 5 mixture components.
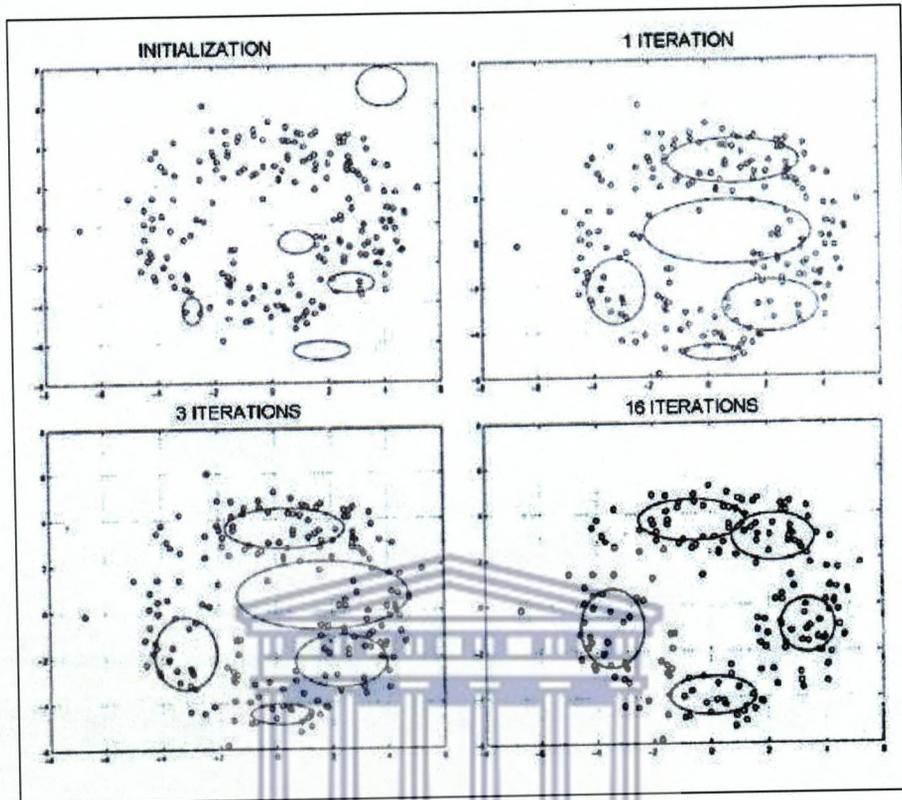
Figure 10: Training with 5 mixture models and recording at different iterations. Notice that the models gradually start to fit the data much better [14].

For finite mixture models, there often exists uncertainty concerning the number of mixture components M to include in the model. The computational cost of models with large values of M is quite high; it is thus desirable to begin with a small mixture and assess the adequacy of the fit.

5 Experiments and Results

5.1 Data Collection and Preprocessing

We have video clips of signers using a single, stationary colour camera against a static background. Digitization of the video clips reduces noise and occlusions. We implement feature extraction algorithms as discussed in Section 3; the feature vectors describe the hand shapes and locations.

The movement is tracked across multiple frames in a single segment. For each sign in the training data, we do the following steps:

1. Take the same sign from five different signers.

2. Perform temporal segmentation to discover cheremes. We are now left with movement sequence segments which describe one or more subunits of a movement.

3. The feature vectors of a single chereme are assigned to a single cluster.

We repeat steps 1 to 3 for different signs. As a result, we have one or more clusters that represent classes of cheremes that we want to recognise.
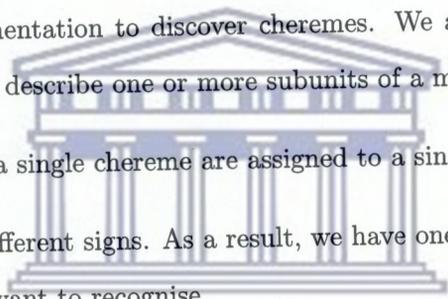
For illustrative purposes, we show how the sign for "open" is broken down into subunits using an example image sequence.

(a)

(b) Location A

(c)

(d)

(e) Location B

(f)

(g) Location C

(h)

(i)

(j) Location D
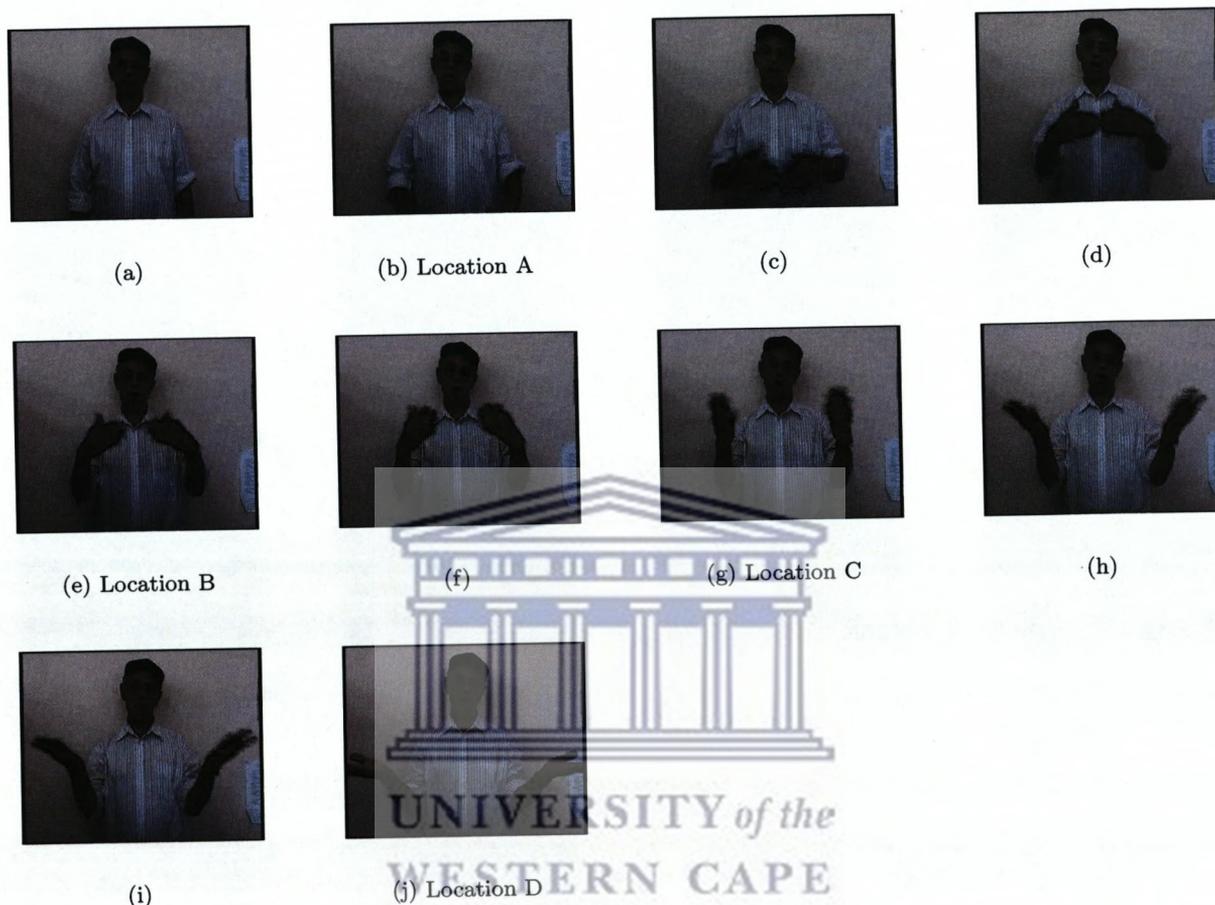
Figure 11: Motion sequence for 'open' sign capturing different stages in execution of the sign

Recall that one of the features calculated from the image frame was intensity of the motion $i(t)$. We make use of this feature to split signs into their chermemes. Figure 12 shows the graph of $i(t)$ against the image frame number generated through the temporal segmentation process.

www.etd.ac.za
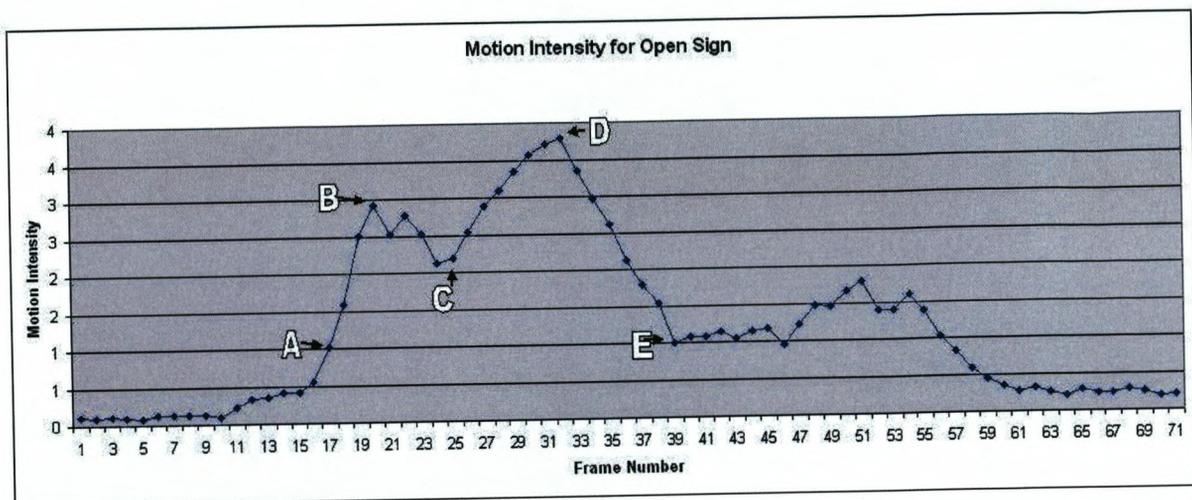
Figure 12: Intensity of the Motion is plotted for each frame to find inconsistencies in the data

We make the following observations from figure 12.

1. We find the first evidence of motion by simply testing whether the intensity of motion exceeds an predefined threshold value. This allows us to align the sign to the movement-pose model. Using this simple method, we find a subset of consecutive frames that have evidence of movement. In figure 12, we see that image frames between points A and E represent the dynamic part of the sign using the threshold value of 1.

2. The image frames in figure 12 before point A and after point E are ignored and treated as stationery, where no motion takes place.

3. The images frames in figure 12 between points A and B represent the movement of hands toward the face as shown in figure 11 on page 48.

4. The images frames in figure 12 between points B and C represent the straightening of the hands as shown in figure 11 on page 48.

5. The images frames in figure 12 between points C and D represent the outward movement of hands from the signer as shown in figure 11 on page 48.

6. The images frames in figure 12 between points D and E represent the return of the hands to the rest position which is not shown in figure 11 on page 48.

49

Table 1 below gives us the results of the number of subunits found for twelve signs.

Sign	Number of Clusters/Cheremes
beard	three
call	five
calm down	five
car	five
doctor	five
electric	six
old	four
open	four
page	six
queen	four
wife	four
you	four

Table 1: Cheremes found in SASL gestures using temporal segmentation

The number of cheremes is then the number of clusters. We now proceed to reduce the number of clusters by noting the following:

1. Signs such as "electric" have repeated motion segments in them; thus, we use just one cluster to represent each repetition of the motion.

2. All the signs start from a resting position and end in the same position; thus, the clusters found at the beginning of the sign and end of the sign are really not part of the sign and can be grouped together in just two clusters, namely upward motion and downward motion.

We are now in a position to train HMMs for each chereme. The signs for testing were the same signs used for training but by different signers. Through trial and error, we are able to find a suitable HMM topology and the number of mixture components to use. We choose the simple left-to-right HMM structure as shown in figure 13.

50
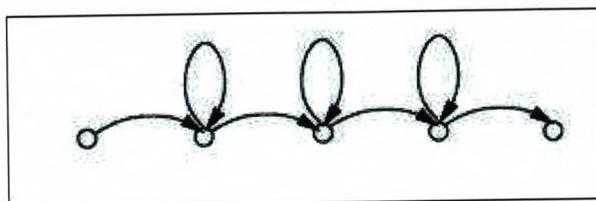
Figure 13: Simple left-to-right HMM

Each of the HMMs has a single initial state which represents the entry state and is restricted to the leftmost state in the HMM. Thus, the initial state probabilities have the property

$$
\pi = \begin{cases} 0 & i \neq 1 \\ 1 & i = 1 \end{cases} \tag{28}
$$

Similarly, the rightmost state is made to be the final state which represents the exit state. Both the first and last states are non-emitting and produce no output probability density distributions. The probability from leaving one state and moving the next state is found in the corresponding entry of the transition probability matrix. We experimented with different number of HMM states and Gaussian mixture components.

For each experiment, we first calculate the global Gaussian means and variances from the training data. Then, we set the component means and variances in each state to the global mean and global variances. For each individual HMM, we apply the Baum-Welch algorithm using the training data. Then, we perform embedded training on the network of HMMs using an embedded version of the Baum-Welch algorithm. We use the Viterbi algorithm to match the input test data against the network of HMMs and output a transcription of each.

Figure 14 shows the average accuracy achieved against a number of states. It shows that the data is best represented by a model with 10 states on average. Using more then 10 states leads to over-pruning errors and an over-fitting of the training data. We find that the system tends to stabilize at 16 mixture components per state. Adding more mixture components does not improve the recognition rate, but considerably increases the computational cost.

51
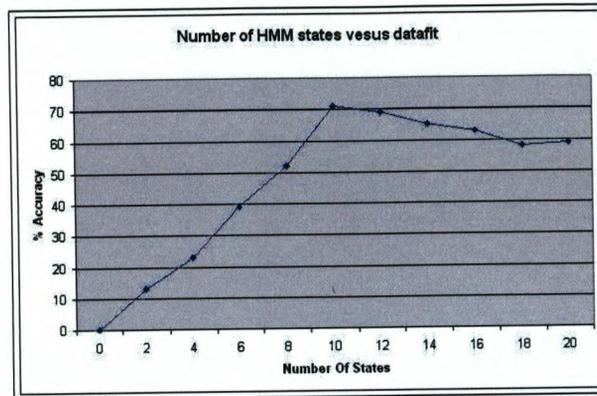
Figure 14: Average accuracy achieved with different number of states. No noticeable improvement in accuracy takes place with states greater then ten

5.2 Results

In our experiments, we used the hold back method whereby we used signs from the first five people for training HMMs and the signs of the sixth and seventh persons for testing of the trained HMMs. Each person performed seventeen signs for a total of 102 training signs. The clustering process resulted in the identification of 23 cheremes. The used the data to train HMMs. Each HMM had ten states. Each state had sixteen mixtures. We achieved an individual chereme recognition rate of 71%.

To test the robustness of our chereme models, we attempted to recognize four of them in complete signs. Table 2 shows representative results from these experiments. The definitions of the four cheremes are shown in Table 3:

1. Chereme 1 is recognized in the signs for "open","old" and "page", chereme 2 in the signs for "you" and "call", chereme 3 in the signs for "my", "me" and "our" and chereme 4 in the signs for "tall" and "doctor".

2. Chereme 1 produced a high recognition rate for the sign "open" because the training chereme originates from this sign. The same can be said for the high recognition rates for chereme 2 in "you" , chereme 3 in "'my"' and chereme 4 in "tall".

3. Chereme 3 produced a high recognition rates in all three signs "my" , "we" and "our" because these signs share very similar movements.

52

Sign	Chereme id	Recognition Rate
open	chereme 1	74 %
old	chereme 1	57 %
page	chereme 1	62 %
you	chereme 2	76 %
call	chereme 2	58 %
my	chereme 3	72 %
me	chereme 3	72 %
our	chereme 3	73 %
tall	chereme 4	73 %
doctor	chereme 4	61 %

Table 2: Recognition Results for Cheremes in Complete Signs
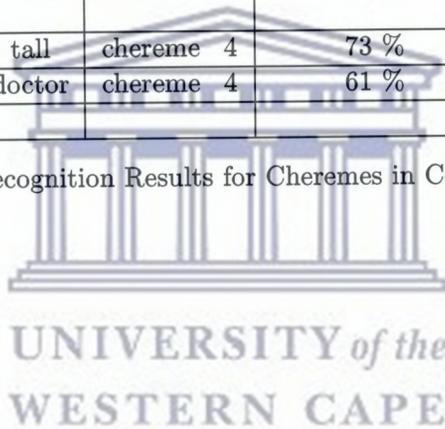
Chereme id	Description
chereme1	2 hands used, open hands ,wrist and elbow rotation in horizontal plane
chereme2	1 hand used, closed hand with index finger extended
chereme3	1 hand used, open hand , palm facing signer
chereme4	1 hand used,index finger extended with open hand, palm facing sideways

Table 3: Description of Cheremes Recognized from Complete Signs

53

6 Conclusions and Directions for Future Research

We have proven our hypothesis that dynamic cheremes can be used to recognise complete dynamic signs by virtue of the fact that our HMMs can pick out cheremes from signs that were not part of the training set used. Since the number of cheremes is considerably smaller than the number of complete signs, it is becomes feasible to build large vocabulary sign language recognition systems that work in real-time. Although we solely used signs from South African Sign Language to build a small chereme recognition system, the methodology is obviously applicable to other sign languages and gesture recognition in general.

While our research makes use of clustering to segment movement cheremes, other methods using transcription and video annotation tools have been applied with similar success. The results may be improved if signers wore coloured gloves; this eliminates the need to find skin regions which may yield better feature vectors. Features dealing with moments could be used to capture handshape descriptions. Other methods such as template matching exist which can be used to track motion and image frames. In our investigation, we chose the simple left to right topology of a Hidden Markov Model. We made the assumption that cheremes occur sequentially in sign language similar to phonemes do in spoken language; this is an unrealistic assumption. Cheremes occur both sequentially and simultaneously in sign. Therefore, different HMM topologies such as ergotic and one-skip-state structures may produce different results. The simultaneous aspect of sign means that the HMMs need to be adapted to properly model gestures found in sign languages. Some HMM variations which may prove useful include parallel HMM and factorial HMM [10].

Sagawa et al point out that a number of conditions must be met for the successful implementation of a full translation system based on chereme recognition that incorporates morpheme recognition [18]:

- Cheremes used in the morpheme recognition process must be recognized.

- The morphemes we have registered must have the same attributes as those morphemes that that we want to recognize.

54

- Make a proper selection of cheremes in the recognition of the sign language morphemes.

Although we have shown that cheremes recognition can be used to recognise and even lead to translation of entire signs, more work on modeling and recognition needs to be done. A real-time system needs to have a robust and efficient classification engine. This would enable proper chereme indexing and retrieval for recognition purposes. Proper classification is also needed to build new signs. One possible approach is to have a repository of cheremes from which new signs can be built, thereby reducing the need to retrain the recognition corpus.

UNIVERSITY *of the*
WESTERN CAPE

References

[1] *Imlib image processing library*, http://www.rasterman.com/imlib.

[2] *Tools for image processing (tip)*, http://sourceforge.net/projects/tip/.

[3] John Carlin Andrew Gelman, *Bayesian data analysis 2nd edition*, (1999).

[4] Will Archer Arentz, *Segementation of skin*, Computer Vision and Image Understanding 94 (2003).

[5] Karl-Friedrich Kraiss Britta Bauer, *Video-based sign recognition using self-organizing subunits.*, 16th International Conference on Pattern Recognition (ICPR 2002), 2002, pp. 434–437.

[6] Alison M. Okamura C. Sean Hundtofte, Gregory D. Hager, *Building a task language for segmentation and recognition of user input to cooperative manipulation systems*, IEEE Virtual Reality Conference (2002), 225–230.

[7] Dimitris Metaxas Christian Vogler, *Adapting hidden markov models for asl recognition by using three-dimensional computer vision methods*, IEEE International Conference on Systems (Orlando, FL), vol. Man and Cybernetics, 1997, pp. 156–161.

[8] ———, *Asl recognition based on a coupling between hmms and 3d motion analysis*, IEEE International Conference on computer Vision (Mumbai, India), 1998, pp. 363–369.

[9] ———, *On managing complexity in sign language recognition*, 2nd High Desert Linguistic Society Conference (Albuqerque, NM), 1999.

[10] ———, *Parallel hidden markov models for american sign language*, International Conference on Computer Vision (Kerkyra, Greece), September 1999.

[11] ———, *Toward scalability in ASL recognition: Breaking down signs into phonemes*, Lecture Notes in Computer Science **1739** (1999), 211–224.

[12] ———, *Handshapes and movements:multiple-channel asl recognition*, Artificial Intelligence 2915 **Springer Lecture Notes** (2004), 247–258.

[13] Jiyong Ma Chunli Wang, Wen Gao, *A real-time large vocabulary recognition system for chinese sign language*, Gesture Workshop, 2001, pp. 86–95.

[14] Jr Donald O. Tanguay, *Hidden markov models for gesture recognition*, Master's thesis, MIT University, 1995.

[15] Tiago Candeias Filipe Tomaz, *Improved automatic skin detection in color images*, http://w3.ualg.pt/ ftomaz/fr/fr.php.

[16] Andreas Kosmala Gerhard Rigoli, *New Improved Feature Extraction Methods for Real-Time High Performance Image Sequence Recognition*, IEEE Int. Conference on Acoustics, Speech, and Signal Processing (ICASSP) (Munich), 1997, pp. 2901–2904.

[17] Stefan Eickeler Gerhard Rigoli, Andreas Kosmala, *High performance real-time gesture recognition using hidden markov models*, Lecture Notes in Computer Science **1371** (1998), 69–80.

[18] Masaru Ohki Hirohiko Sagawa, Masaru Takeuchi, *Description and recognition methods for sign language based on gesture components*, International conference on intelligent user interfaces. New York, NY : ACM (Association for Computing Machinery) **Moore, Johanna / Edmonds, Ernest / Puerta, Angel (eds)** (1997), 97–104.

[19] Jin-Hyung Kim Hyeon-Kyu Lee, *Gesture spotting from continuous hand motion*, Pattern Recognition Letters,19(5-6) (1998), 513–520.

[20] George Kollios Jonathan Alon, Stan Sclaroff, *Discovering clusters in motion time-series data*, IEEE Computer Vision and Pattern Recognition Conference (CVPR) (2003).

[21] Mohammed Waleed Kadous, *Machine recognition of auslan signs using powergloves: Towards large-lexicon recognition of sign language*, In Lynn Messing, editor, Proceedings of WIGLS. The Workshop on the Integration of Gesture in Language and Speech **The Workshop on the Integration of Gesture in Language and Speech** (1996), 165–174.

[22] Ahmed H.Tewfik Keesook J.Han, *Eigen-image based video segmentation and indexing*, IEEE International Conference on Image Processing (1997).

[23] Marcell Assam Kirsti Grobel, *Isolated sign language recognition using hidden markov model*, In Proceedings of the IEEE International Conference on systems, Man and Cybernetics (1997), 162–167.

[24] C. Vogler W. Schuler N. Badler L. Zhao, M. Costa, *Modifying movement manner using adverbs*, Communicative Agents in Intelligent Virtual Environments (2000).

[25] Ladefoged, *Elements of acoustic phonetics*, The University of Chicago Press, 1962.

[26] Judith A. Markowitz, *Using speech recognition*, vol. Multimedia Database Management Systems, ch. 5, Prentice Hall, 1996.

[27] Soowon Kim M.B. Waldron, *Isolated asl sign recognition system for deaf persons*, IEEE Transactions on Rehabilitation Engineering, 3(3) (1995), 261–271.

[28] Djemel Ziou Nizar Bouguila, *Unsupervised learning of a finite dirichlet mixture model*, Pattern Recognition Letters 26(12): 1916-1925 (2005).

[29] Thomas S. Huang Pengyu Hong, Matthew Turk, *Gesture modeling and recognition using finite state machines*, IEEE Conference on Face and Gesture Recognition (March 2000).

[30] Manfred Lang Peter Morguet, *Feature extraction methods for consistent spatio-temporal image sequence classification using hidden markov models*, IEEE International Conference on Image Processing.

[31] P. Laskov R.Erenshteyn, *A multi-stage approach to fingerspelling and gesture recognition*, To appear in the proceedings of the Workshop on the Integration of Gesture in Language and Speech (Wilmington, DE, USA), 1996.

[32] M. Ouhyoung R.H. Liang, *A real-time continuous gesture recognition system for sign language*, In Proceedings of the Third International Conference on Automatic Face and gesture Recognition (1998), 558–565.

[33] Matthew Turk Ross Cutler, *View-based interpretation of real-time optical flow for gesture recognition*, Proc. Third IEEE Conference on Face and Gesture Recognition (1998).

58

[34] D.S.Chauhan Sanjay Kr. Singh, *A robust skin color based face detection algorithm*, Tamkang Journal of Science and Engineering **6** (2003), 227–234.

[35] Thad Starner, Joshua Weaver, and Alex Pentland, *A wearable computer based American Sign Language recognizer*, Lecture Notes in Computer Science **1458** (1998), 84–90.

[36] Gerhard Rigoll Stefan Eickeler, Andreas Kosmala, *Hidden Markov Model Based Continuous Online Gesture Recognition*, Int. Conference on Pattern Recognition (ICPR) (Brisbane), 1998, pp. 1206–1208.

[37] Don X.Sun Steven E.Golowich, *A support vector/hidden markov model approach to phoneme recognition*, 1998, http://citeseer.ist.psu.edu/golowich98support.html.

[38] William C Stokoe, *Sign language structure: The first linguistic analysis of american sign language*, Newly Revised. Silver Spring, MD: Linstok Press, Incorporated (1978).

[39] William C. Stokoe, *Sign language structure: An outline of the visual communication system of the american deaf*, Studies in Linguistics: Occasional Papers 8. Linstok Press, Silver Spring, MD, 1960 (Revised 1978).

[40] Words Nobuhiko Tanibata, *Extraction of hand features for recognition of sign language*, International Conference on Vision Interface, 2002.

[41] Alex Pentland Thad Starner, *Real-time american sign language recognition from video using hidden markov models*, Technical Report 375, MIT Media Laboratory (1996).

[42] Ying-Qing Xu Nan-Ning Zheng Tian-Shu Wang, Heung-Yeung Shum, *Unsupervised analysis of human gestures*, Lecture Notes in Computer Science **2195** (2001), 174–200.

[43] S. Gilbet T.Lebourque, *A complete system for the specification and generation of sign language gestures*, Springer Verlag, 1999.

[44] A. Waibel X. Zhu, J. Yang, *Segmenting hands of arbitrary color*, International Conference on Automatic Face and Gesture Recognition (Grenoble), 2000.

[45] Kwang Yoen Wohn Yanghee Nam, *Recognition of space-time hand-gestures using hidden markov model*, ACM Symposium on Virtual Reality Software and Technology (1996).

UNIVERSITY *of the*
WESTERN CAPE