Gesture Recognition in a Meeting Environment

N. Hassink
M.G. Schopman

Master's thesis
Enschede, February 2006

Graduation committee
dr. Mannes Poel
ir. Ronald Poppe
prof. dr. ir. Anton Nijholt
dr. Dirk Heylen

Human Media Interaction group
Department of Electrical Engineering,
Mathematics and Computer Science

University of Twente
Enschede - The Netherlands

Abstract

This thesis describes a research project related to gesture recognition in a meeting environment. In this research project we want to determine where the challenges lie in gesture recognition and what the recognition performance is when we apply existing machine learning techniques in a real life setting such as meetings. The research is split up in four parts. The first part is feature selection. This part encompasses the process of analyzing meetings on useful gestures, annotating these gestures, parameterization with possible features and selecting the most useful features. The second part is segmentation, the process of automatically locating gestures in a meeting. Two segmentation approaches are examined; whole gesture segmentation and gesture part segmentation. Two methods, BIC and AM, are compared for each approach. The third part is feature clustering, the mapping of continuous data to discrete data. Two methods are compared for this purpose, K-Means and Expectation Maximization. The final part is classification, labeling data parts with the correct gesture label. Hidden Markov models are used for classification. The main goal is to compare the classification performance on annotated gestures with the classification performance on automatically segmented gestures. From these four phases follows the project's conclusions and recommendations.

Preface

Gesture recognition is a young research field. The lack of standard, fully developed approaches was a big motivation for us to do this research. The thesis before you is the summary of a year's hard work on gesture recognition. Although the project has taken a bit longer than the customary seven months it is over before you know it. We still recall reading the first articles and discussing the new book of Alpaydin during weekly book meetings. For us it is also the final part of our computer science study at the University of Twente. After five and a half pleasant years it is time to take the next step into the business or academic world.

We would first like to thank our graduation committee for their feedback and ideas on this thesis. Especially Ronald Poppe and Mannes Poel, who read often lengthy chapters filled with ideas, but sometimes lacking a clear line. Also we thank our roommates of the notorious Black Coffee Room, Laurens Satink, Michel Boedeltje, Hans Wim Tinholt, Hans Dollen and Hilco Kats, for the necessary news discussions and soup lunch breaks.

I (Niek) would personally like to thank my parents and sister for their support during the weekends last year. Also for the nice short trip to Denmark, this really took my mind of the project for a while. I would also like to thank my roommates Bart ten Brinke and Alexander Vos de Wael for helping to unwind after work hours. Last but not least I would like to thank my project partner Maarten Schopman. I found working on the same project very pleasant because you always have someone to directly discuss ideas with, someone who precisely knows what you are doing.

I (Maarten) would like to start with thanking my project partner Niek Hassink. The ability to discuss ideas, test results and reports, with someone who knows just as much of the subject was a huge advantage. Next I would like to thank my family and friends for their repeated interest in how the project was going on. Last but not least I would like my girlfriend Marieke van Gemert for listening to my progress reports and lengthy explanations on the gesture recognition subject.

Niek Hassink and Maarten Schopman
Enschede, February 2006

Table of contents

Chapter 1 - Introduction

To introduce the topic of gesture recognition we will first explain what we mean when we speak of gestures. What kind of movements are gestures and what are common characteristics that separate them from other types of movements? Following this we give our motivation and the objective of this research project. To place our research into context, we give an overview of the current state of the art in gesture recognition. We will present the background, some examples of previous work and commonly used methods and approaches. With the state of the art in mind, we look at the approach we want to take in this project.

1.1. What is a gesture

When people hear the term gesture recognition, their first reaction almost always is: "Oh so you are doing something with sign language." Although sign language is an important and well known type of gesture, it is certainly not the only type of gesture. In this thesis we look at a broader definition. Next to hand gestures we also look at head gestures such as nodding and whole body gestures such as standing up. These gestures differ from each other in a lot of aspects. But the thing that separates gestures from other types of movement is their relation with communication. This idea is nicely formulated by Nespoulous [43].

> *"The notion of gesture is to embrace all kinds of instances where an individual engages in movements whose communicative intent is paramount, manifest, and openly acknowledged."*

Before we look further into techniques used in gesture recognition it is important to consider what the characteristics of gestures are. Gestures are variable in space and time, a so called spatio-temporal event. Gestures are variable in time because they have a certain start time, a variable duration and an end time. Also two examples of the same gesture will never be exactly alike. This makes gestures also variable in space.

Some early studies have looked into the temporal characteristics of gestures. Kendon [34] states that a gesture consists of three phases: preparation, nucleus and retraction or reposition. The gesturing person first makes a preparatory movement, followed by the actual core of the movement, the nucleus, followed by a retraction to a rest position or a reposition for a new gesture phrase. McNeil [38] proposed a similar structure where he distinguished the following phases: preparation, (optional) pre-stroke hold, stroke, (optional) post-stroke hold and retraction. The structures of Kendon and McNeil focus only on the different temporal phases of a gesture and give no description of the gesture itself. Rossini [51] looked into spatial characteristics of gestures and proposed to enrich the phases of Kendon and McNeil with certain measurable parameters. Such as the angle of the moving joint, gesture timing, point of articulation (the main joint involved in the gesture) and locus (the main body part involved in the gesture).

In the next paragraph we first describe our motivation and objective for this research project, before we take a look at the state of the art in the research field of gesture recognition.

1.2. Project goal

The goal of this project is based on the motivation of the AMI project and our own motivation. From our motivation follows the main objective for this project. The obvious project goal is to fulfill this objective.

1.2.1. Motivation

New technologies open up new channels of communication for human-computer interaction. Traditional human-computer interaction devices (keyboard and mouse) are more and more replaced with these other channels of communicative input. These channels offer opportunities for communication in a more natural way using a variety of modalities, for example speech, text and nonverbal cues such as gestures. New opportunities also create new challenges. To face some of these challenges, sixteen partners from both the academic and the industrial world have combined their efforts in the AMI (Augmented Multiparty Interaction) project [5]. The general target of the AMI project is to support human interaction in the context of smart meeting rooms and remote meeting assistants. The main goals are to enhance the value of offline meeting recordings and to make real-time human interaction more effective. To achieve these goals new tools are developed for computer supported cooperative work and browsing and searching in multimodal meeting recordings.

As part of the multimodal input interface, the AMI project looks into gesture recognition as a form of visual input. Gesture recognition can play a role in the two main goals stated above, the online and offline enhancement of the meeting environment. A real-time form of gesture recognition can serve as part of the multimodal input interface in the online meeting environment. An offline form of gesture recognition can help enhance the value of recorded meetings, for example to search for video sequences where voting gestures occur.

Our own motivation for this research is based on some general observations we made during our literature study. We observed that most research in the gesture recognition field makes several assumptions on the recognition problem at hand. It is often assumed that:

- The features describing the gestures are very precise and insensitive to noise. The features are often obtained in a controlled "laboratory like" environment.
- The gesture set is limited and consists of easy separable gestures.
- The gestures can be easily segmented from other types of movement in the feature sequences or the gestures are already segmented beforehand.

In our opinion these assumptions cannot be made for real-life applications of gesture recognition. In more natural gesture recognition applications the feature extraction will most likely be less controlled. This is because using obtrusive devices, such as a data glove, will be out of the question. The features will most likely come from a computer vision based analysis of one or more camera recordings. We also think that it is unrealistic to assume that feature sequences are segmented beforehand. We expect that gesture segmentation will be a difficult research area.

A second aspect of our motivation is that gesture recognition is a relatively new area of research. The lack of a standard, fully developed approach makes this research challenging.

1.2.2. Objective

As we have seen in the motivation paragraph most gesture recognition systems used in the literature are tested and applied in a laboratory like environment. In these previous studies the boundaries, on how to perform a certain gesture, are often strictly defined. The features describing the gestures are often precise and not very susceptible to noise. The chosen gesture set mostly consists of gestures that are easily distinguishable by gesture recognition systems. Moreover, the video data is specifically recorded for gesture recognition. It often contains one isolated gesture, making gesture segmentation unnecessary.

The meeting setting of the AMI project is a different environment, in which gestures are produced in a more natural way. The meetings were not specifically recorded for use in a gesture recognition system. Recognizing gestures in this more natural environment poses some new challenges. Features, obtained from a video recording with computer vision techniques, will be less precise than features obtained with a data glove for example. Gestures performed during meetings such as nodding will not be so easily distinguished from other types of movements. Also, the gestures are not recorded as separate isolated gestures but they are part of one entire meeting recording. This introduces the additional problem of having to segment the gestures from the rest of the meeting.

This discrepancy brings us to our research objective. In this research project we want to determine where the challenges lie in gesture recognition and what the recognition performance is when we apply existing machine learning techniques that are used in gesture recognition to recognize a set of predefined gestures in the more natural meeting setting.

We focus in this project on an offline form of gesture recognition. We want to locate gestures in pre-recorded meetings. This limits the problem area because additional real-time constraints are not included. To place our research into context and to determine which techniques and approaches are commonly used in gesture recognition, we take a look at the state of the art in the next paragraph.

1.3. State of the Art

In the past two decades different fields of computer science have taken more and more interest in the domain of "Looking at People". This domain covers a wide span of problem areas such as face recognition, gesture recognition, tracking humans and emotion research. Because of this wide problem span there is much attention from both the field of computer vision (finding and tracking objects) as well as the field of machine learning (pattern recognition). Recently the field of Human Computer Interaction has taken an interest, because of the promising applications this domain offers in creating a more natural way for humans to interact with computer technology in their environment. The ability to recognize human gestures opens up a wide range of possible applications. Just a few examples of application areas are:

- Automatic recognition of sign language to facilitate communication with the hearing impaired.
- Using gestures as part of a more natural interface with computer technology.
- Making video searchable, for example searching for voting gestures in a video recording of a debate.
- Using a recognition result as input for character animation to replay a certain event in a virtual world.
- Using gestures as input to explain the emotion of a gesturing person.
- Recognizing suspicious movements for surveillance purposes.

This survey first provides an overview of recent surveys in the gesture recognition domain. Secondly it illustrates two common approaches seen in gesture recognition. In the remaining part of the survey, one of these approaches is examined in detail. This will result in an illustration of problems areas and state of the art methods and solutions. This approach and these methods will serve as a source of inspiration for our own approach described in Paragraph 1.4.

1.3.1. Previous work

The interdisciplinary research field of gesture recognition originated from the fields of computer vision and machine learning. Because of this, gesture recognition surveys often make a categorization in methods originating from either a computer vision or a machine learning point of view. Computer vision approaches typically use some form of low-level modeling of the motion dynamics to recognize a gesture. For example Motion History Images used by Davis and Bobick [17] in combination with template matching. The machine learning point of view concentrates more on finding patterns and relationships in high-level features, for example trajectory parameterization and state-space approaches. Because work in the domain of gesture recognition has been carried out for more than a decade, there have been a number of previous surveys on this topic. To put this survey into context these will be discussed shortly in chronological order.

- Cédras and Shah [12] covered work prior to 1995 in their survey. They focus on the subjects of motion extraction, motion recognition and motion tracking. The survey does not focus on a specific type of gesture but uses the global idea of human motion.
- Wu and Huang [66] published a survey in 1999 about how to represent human gestures, what features to use and how to collect this data. It also surveys some techniques for temporal gesture modeling and sign language recognition.

- Aggarwal and Cai [1] published a survey in 1999 covering mainly human motion analysis and motion tracking. The last chapter of this survey focuses more on activity recognition.
- In 1999 Gavrila [24] published a survey about the visual analysis of human movement. The survey starts with the different 2D and 3D approaches for motion analysis and concludes with a description of different action recognition methodologies.
- In 1999 Wang and Singh [61] describe the methods used in tracking and motion analysis of the whole body and of the different body parts. This survey focuses mainly on the feature extraction process.
- Moeslund and Granum [39] made a survey in 2001 which covers the initialization of gesture recognition systems, the tracking of human motion, pose estimation and recognizing human motion.
- Turk wrote, in the handbook of virtual environments [58] in 2002, an overview chapter on gesture recognition. This overview looks into the nature and representation of gestures. It also surveys different pen-based, tracker-based and vision-based gesture recognition approaches.
- Wang et al. [62] presented in 2003 a survey about motion detection, tracking, behavior understanding and action recognition. This survey focuses mainly on the feature extraction process.
- Chellappa et al. [13] discussed in 2005 different techniques for human identification using face or gait analysis. Also human activity recognition is covered but the focus is not on gestures but on activities, such as walking, for use in surveillance systems.

As said, the surveys mentioned above frequently make the distinction between a computer vision and a machine learning point of view. In the next paragraph these two approaches are examined and compared.

1.3.2. Two common approaches

In this paragraph we examine the computer vision and machine learning point of view in more detail. The simple roadmap of Figure 1.1 is the same for both approaches. First a feature extraction module produces some form of features from the video. These features are used by the classification module to identify and label the gestures. A gesture interpretation module uses this classification result to give a meaning to the gesture, given the context in which the gesture was made. For example a recognized nodding gesture in context of a discussion indicates that the gesturing person agrees.

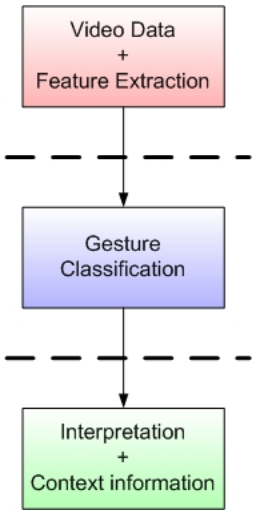


Figure 1.1 – Common roadmap in gesture recognition

In the feature extraction phase, most computer vision approaches transform a sequence of raw video images to a new image which captures the motion present in that sequence. These motion history images are compared in the gesture classification phase with a number of templates, one for each gesture. The gesture template which resembles the current motion image the most, classifies the current sequence. Most methods update the template afterwards with the new information of the current sequence to improve the template quality.

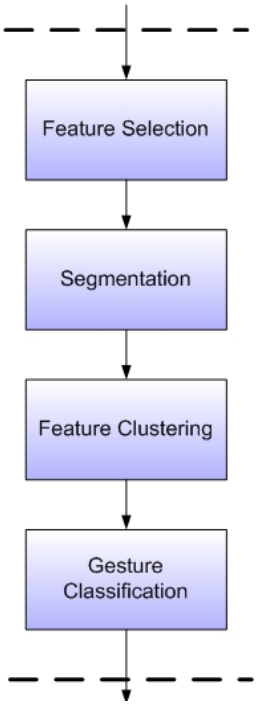


Figure 1.2 – Structure of machine learning based classification

Machine learning approaches take a different route in the feature extraction phase. Instead of computing one new image these methods abandon this image representation and transform a sequence of images into a sequence of higher level features. These features describe the motion in the images for example in terms of position of the hands or changes in velocity. The actual machine learning techniques come into play in the gesture classification phase. The general idea is to look for patterns and relationships in the feature data which are specific for a certain gesture. When such a specific pattern is present in the sequence of features, the corresponding gesture was performed in the original video.

This survey focuses only on the machine learning approach to the gesture classification phase of Figure 1.1. A more detailed approach of this phase is given in Figure 1.2. This phase consists of selecting the most useful features from the feature extraction phase, segment these feature streams in time and cluster them in space to get a suitable representation for classification. These four sub-phases and corresponding techniques will be discussed in more detail in the next paragraphs.

1.3.3. Feature Selection

The first step of this phase is to determine all high level features that can be obtained directly or that can be calculated from the feature extraction module of Figure 1.1. Features that can be obtained directly are for example angles between different joints. Calculated features can for example be angular velocity or angular acceleration.

The second step of the feature selection phase is to select a subset of features which are most suitable for the segmentation, clustering and classification phases that follow. An approach to this problem is to determine which features give the best description of the different gestures and at the same time discriminate the best between the different gesture classes. The so called most expressive features (MEF) and most discriminating features (MDF). This approach is described by Wu and Huang [66].

Another possibility is to represent the features in a smart way by deriving a new smaller set of features from the original feature space. When enough features and possibly different representations of the same features, are calculated it is possible to reduce the feature space to a more compact form. During this process it is important to retain as much of the original feature information as possible in the new, smaller feature set. A number of techniques are available for this problem. Principal component analysis (PCA) is for example used by Wu and Sutherland [65]. Fang et al. [21] use Self Organizing Feature Maps (SOFM) to reduce their feature vector.

1.3.4. Segmentation

The general idea of the segmentation phase is to segment the spatio-temporal feature data in time. The aim is to obtain time sequences of features containing a gesture or gesture part. Segmentation is necessary because the person on the video will not be making gestures all the time. The feature streams will therefore not contain useful gesture data at every given moment. By segmenting the gestures explicitly, one can leave out those non-interesting parts of the data and consider only the gestures or gesture phases. It is also possible that two different gestures follow each other directly. Segmentation is in this case necessary to be able to split these two gestures. Kendon [33] mentions that humans also segment gestures first before recognizing the gesture itself. This may also be an indication that segmentation is useful.

The most common approach to the segmentation process is explicitly extracting a complete gesture from the stream of feature vectors using some form of activation function or threshold. An alternative approach is not to segment explicitly before classification. This option is possible when a state space approach, such as a hidden Markov model (HMM), is used for classification. State transitions in an HMM are based on transition probabilities. These probabilities can be considered as an activation function. This way the classification method implicitly segments the data stream by remaining in a start state until it detects a gesture start. The end of a gesture is detected when the HMM enters an end state. A third option is a combination of the two methods mentioned above. This method explicitly segments the data stream on a lower level into gesture parts, instead of segmenting the whole gesture. These gesture parts are then combined in an HMM. These three options will be discussed below in more detail.

Almost every explicit segmentation method mentioned in the literature makes the assumption that a gesture has rest poses at the beginning and at the end of a gesture. It is assumed that in such a rest pose a certain measured activity drops below a threshold or is in a local minimum. When this happens a gesture boundary is detected. This notion of rest poses is described by McNeill [38]. He states that a gesture always starts and ends in a certain rest state. McNeill also describes optional rest poses or holds within a gesture phrase: preparation, (optional) pre-stroke hold, stroke, (optional) post-stroke hold and retraction. These holds within a gesture could be a potential problem because they can result in boundary detection within a gesture. On the other hand, holds and rest states will likely differ in duration. Holds will generally be shorter than rest states making them distinguishable.

The features used to calculate an activity measure differ between papers. Camurri et al. [10] measure the amount of detected motion by looking at variations in the silhouette and position of music and dance performers in the last few frames. This quantity of motion measure contains information about velocity and force. Fang et al. [21] segment continuous sign language data produced with a data glove. They first extract position, orientation and posture data, this data is transformed using a self organizing feature map (SOFM). The output of this SOFM is used to train a recurrent network which is used to segment the data. They obtain good results using this approach. However, the data glove and sign language setting, is less relevant to our research. Howell and Buxton [26] use the amount of changing pixels in pointing and waving gestures, between two frames, to detect how much motion is present. When the amount drops below a certain threshold a gesture boundary is detected.

Zhao [67] uses a combined zero-crossing and curvature method to detect boundaries in all sorts of motion of the human body. For their motion samples this method is more reliable than only using zero-crossing in the second derivative of the motion data, to detect significant changes. Zhao also mentions that the curvature is prominently high when a motion starts, ends or changes direction. Kahol et al. [32, 31] use a minimum force measure to determine the local minima in total body force of dance movements. This total body force is calculated from the force, kinetic energy and momentum of the different body segments. While 93% of the boundaries are correctly detected, it is mentioned that some insertions were made.

Implicit segmentation using a state space approach such as an HMM is applied by Rigoll et al. [49], and McCowan et al. [37]. The disadvantage of this segmentation method is that an HMM cannot cope well with overlapping gestures or gestures with intermediate poses that resemble begin states of other gestures, see [31]. Overlapping gestures could also affect the performance of explicit segmentation. However explicit segmentation has some possible solutions for this problem, such as different activity measures, while implicit segmentation has not.

The last possibility, to combine the two methods mentioned above, is taken by Wang et al. [63]. They use local minima in acceleration to find certain boundaries. The blocks between these boundaries are seen as characters of a gesture alphabet. The concatenation of different gesture characters forming a gesture, is modeled using an HMM. Parallels can be drawn between this approach and speech recognition. For the interested reader we refer to a book by Jurafski and Martin [30], chapter 7 in particular. The elements of a gesture alphabet can be seen as phoneme-like units. In speech recognition the audio data is segmented into phonemes and combined to form different words. The same principle can be applied to gesture recognition by segmenting the feature data into small phoneme-like elements and combine them to form a single gesture. A difference with speech recognition is that predefined phonemes do not yet exist for gestures. This approach is applied to sign language recognition by Murakami and Taguchi [42], they use recurrent neural networks to construct the different sign language characters. Birk et al. [8] use principal component analysis to construct these sign language characters.

1.3.5. Feature Clustering

Whereas in the segmentation phase the spatio-temporal feature data is segmented in time, feature clustering is used to cluster the feature data in space. A number of classification methods used in the next phase require, or work better on discrete data. The general idea of clustering is to label all feature vectors at frame level with a discrete label. This maps similar feature vectors to the same label, reducing the search space and making the classification problem less complex. To handle this task there are several vector quantization techniques at our disposal. There are unsupervised iterative techniques such as K-means clustering or its generalized counterpart Expectation Maximization (EM). There are also supervised learning based techniques, such as self organizing maps or feed forward neural networks.

Instead of labeling at frame level, it is also possible to go a step further by grouping several feature vectors together and give a single label to this group. We call this labeled group of feature vectors a gesture building block. In the remainder of this thesis we will abbreviate this term as GBB. A gesture can then be seen as a concatenation of a number of these GBBs. The GBBs can be intuitive, visually observable parts in a gesture or unintuitive, non-observable parts. For example, the

phoneme like units, mentioned in the previous segmentation paragraph, can be used as parts. An advantage of creating an intermediate abstraction layer of GBBs is that you can map different feature sets to the same GBBs. This way you can treat the classification of GBBs as a separate and maybe simpler classification problem. Bauer and Kraiss [6] use K-means clustering in sign language recognition. The feature vectors are clustered into different classes. Each class has its own representation called a Fenonic baseform. These Fenonic baseforms are the GBBs they use. A concatenation of baseforms forms a sign. Gaffney and Smyth [23] use a linear regression mixture model and a kernel regression mixture model to cluster trajectory based data. This method compensates for trajectories that belong to the same cluster but differ in duration.

1.3.6. Classification

In the classification phase the feature data from the previous two phases has to be given a gesture label. Classification only gives a name to a sequence of feature data. The parameterization and interpretation are left to subsequent phases. In the literature there are two main methodologies commonly used for classification. One focuses on a model based approach, where certain dynamics of human motion are considered. The other is a state space approach, where intermediate states of a gesture are used for recognition. Both methodologies will be discussed.

When using the model based approach there are different ways in which one can model the underlying dynamics of a certain gesture. Examples are the calculation of hand trajectories [41, 35, 15], calculating angles between different body segments [53] or using changes in the velocity or acceleration of the motion [13]. The model based approach classifies different gestures by matching the data to be classified with certain predefined templates. These templates are different from the templates used in computer vision approaches, mentioned earlier in this survey. The computer vision approaches map the gesture to a static image template. The templates discussed here model the dynamics of certain gesture features and therefore do not take place in the image domain. The templates represent the average of a certain gesture class. How this average is computed is beyond the scope of this survey. The gesture template that resembles the data sample the most, will classify the gesture. This resemblance between template and data sample is usually determined using a similarity measure. Several measures can be defined such as the maximum deviation error between template and sample, the sum of the deviations for each data point or the sum of squared deviations for each data point. See for example [15]. A disadvantage of using a template is that the duration of the average gesture in the template is fixed at a certain value. A new data sample will likely have a different duration, which means you have to compensate for this difference. Dynamic Time Warping is frequently used for this purpose.

The state space approach assumes that a gesture is essentially a sequence of states. Static postures are most commonly used as states. These states in combination with certain state probabilistic transitions form a framework for gesture recognition. Classification is done by evaluating how probable it is that an observed sequence of states is a certain gesture. The method used in a state space approach is almost always a form of hidden Markov model (HMM). HMMs are specifically suitable for time varying data because the time component is implicitly modeled in the probability matrix of the HMM. Gestures can stay longer or shorter in intermediate states. However this duration compensation has its limits. If gestures differ too much in duration the probability matrix becomes distorted.

A disadvantage of HMMs is that they make the assumption that a gesture is made up of a sequence of discrete states. This implies that the data is piecewise stationary, which may not be the case. It is possible to overcome this problem by using continuous states in an HMM [3]. Also HMMs will generally have a poor performance if the Markov condition doesn't apply to the data. An *N*-order Markov condition assumes that the current state depends only on the *N* previous states. The problem with this assumption is that a large *N* is needed to model long term dependencies in gestures, which makes the problem computationally too complex and training virtually impossible. Nevertheless HMMs are the most commonly used method in different gesture recognition applications [21, 49, 6, 60, 25, 36, 59, 52].

Occasionally a few special cases of the HMM approach are used, such as an HMM/neural network hybrid [14]. In HMMs the output probability distribution of each state is assumed to have a certain trainable parametric distribution. Neural networks on the other hand do not make an assumption on the statistical distribution of patterns in the input space. A hybrid of these methods uses neural networks to estimate these state distribution functions of the HMM, resulting in the best of both worlds. Semi-continuous HMMs are used by Zobl [69]. This HMM bridges the gap between discrete and continuous HMMs, by using the codebook of a vector quantization in the output distribution of each HMM state. This gives the advantage of a lower quantity of estimated parameters compared to a continuous HMM, which makes the problem easier to learn. Bengio proposes asynchronous HMMs [7] to cope with asynchrony that can occur when multiple data streams are used as input for classification. For example pointing to a map and then saying "I want to go there" leads to asynchrony between the gesture and the speech fragment.

There are some other methods which can be used for classification such as radial basis functions (RBF) or neural networks. The use of these methods appears sometimes in the literature [26, 27], but they are far less commonly used than the two approaches discussed above. This can be explained by looking at the nature of these methods. Neural networks were originally devised to handle the classification or regression of static data. The gesture recognition problem has data that varies over time. This temporal component makes the problem less suitable for neural networks. Some extensions have been made, such as time delay neural networks, time delay RBF or recurrent neural networks, to cope with this temporal component. These extensions require an assumption on the time window to use. Short time dependencies can be modeled with these neural networks. Longer dependencies require a larger time window which increases the network complexity too much. These methods remain therefore in principle less suitable for time varying problems.

1.4. Approach

The approach we want to take is the same approach as the roadmap given in Figure 1.2 of the state of the art overview. This means that the project will be divided into four phases. A more detailed version of this part of the roadmap is given in Figure 1.3. Below, each phase is discussed separately in terms of the input, what is done with this input and the output. Furthermore we discuss the way in which we divided the task in this project.

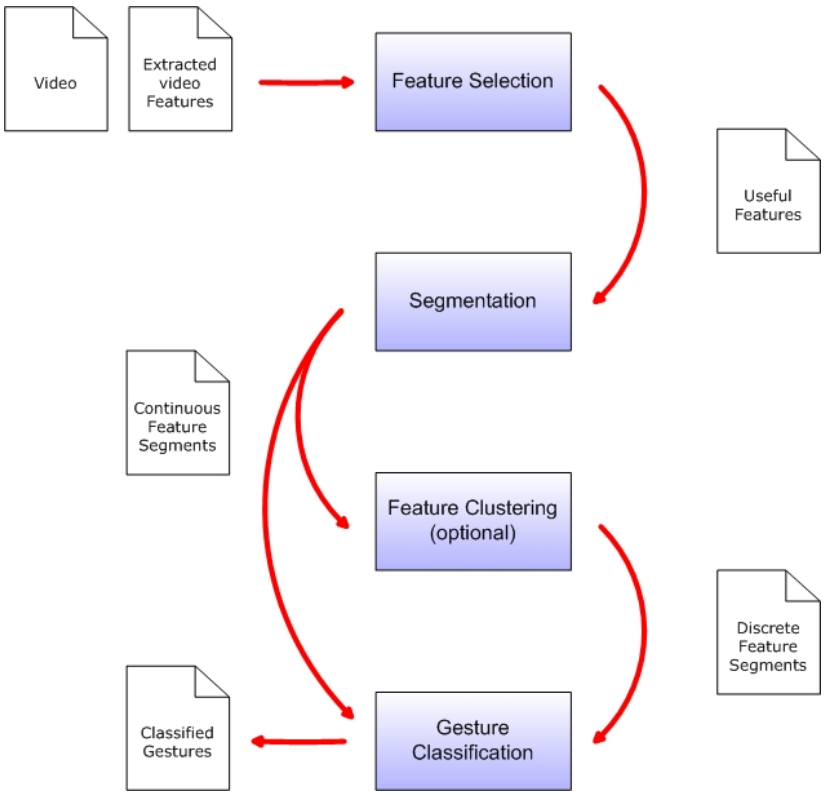


Figure 1.3 - A detailed figure of the input and output of each phase. The video files are annotated and useful features for segmentation and classification are determined from the extracted video features. The useful segmentation features are used in segmentation resulting in feature segments. The data in these feature segments can be made discrete with feature clustering. Based on the useful classification features, the data in the continuous or discrete segments is labeled in classification, resulting in classified gestures.

1.4.1. Feature selection

Video's, extracted video features → Useful features

The purpose of the feature selection phase is to find useful features for the segmentation and classification of gestures. Useful features for classification are those features which consistently describe the gestures and make a clear distinction between the gesture classes. These features can be obtained either directly from the extracted video features or by calculating new features from these extracted features. However before a selection of useful features can be made, some other steps have to be taken first.

Step 1 – Video analysis

Before we can select features, we first have to select the gestures we want to recognize. We want to know which gestures are commonly made during a meeting and which are useful to recognize. For this purpose the fist objective of this phase is to make an analysis of the meeting videos resulting in a selection of useful gestures.

Step 2 – Annotation

Features are calculated for the entire duration of a meeting. To be able to say something about the usefulness of a feature for a certain gesture, we have to label the correct part of the corresponding feature file. For this purpose we have to know exactly when a certain gesture takes place. This asks for the annotation of a certain amount of meeting data. The annotation process is the second objective of this phase.

Step 3 – Gesture parameterization

The third objective of this phase is to analyze ways in which gestures can be parameterized with features. For this purpose we are going to look at how people recognize gestures, what properties a good feature should have and which features we can calculate from the extracted video features. We also take the human perception into account because this might provide clues for the parameterization of gestures.

Step 4 – Feature Selection

Once we have determined a collection of features, the question which features should be used, can be answered. Our goal is to first filter the obtained feature set on outliers and noise. Second, using dimensionality reduction techniques, we try to remove those features that do not contain enough useful information. This leaves a set of features, from which we can select an optimal set in both the segmentation and classification phase.

1.4.2. Segmentation

Useful features → Feature segments

The purpose of segmentation is to automatically find the gesture boundaries in the feature stream of an entire meeting. In the annotation step this segmentation is already done by hand for a number of meetings. However, if we want our system to recognize gestures by itself in a meeting video, this segmentation has to be done automatically. The approach to the segmentation problem is also split up in a number of steps.

Step 1 – Segmentation technique selection

The first step is to select and elaborate on two segmentation techniques using the available literature on this topic. The two techniques we are going to compare are a technique based on an activity measure (AM) and a probabilistic technique that uses the Bayesian information criterion (BIC).

Step 2 – Segmentation feature selection

The next step is to determine which subset of the useful features is most suitable for segmentation. Hand gestures will be segmented on a different feature set than head gestures for example.

Step 3 – Testing and evaluation

The last step is to test the performance of the two selected segmentation techniques. To test this performance we need to set up an evaluation method that compares the automatically placed boundaries with the annotated boundaries. We will use two evaluation criteria. One is based on segmenting the gesture as a whole from the rest of the meeting. The second is a more tolerant criterion which is based on segmenting gesture parts. These parts may be used later to construct the earlier discussed GBBs. With these evaluation criteria we can evaluate the performance of the two segmentation techniques and select which technique is best.

1.4.3. Feature clustering

Continuous feature segments → Discrete feature segments

Feature clustering is used to map the continuous values of the feature data to a number of predefined discrete values. The clustering method serves as a way to remove certain variations within a gesture class and map an occurrence of a gesture to a range of discrete values. This clustering step also allows the use of some classification techniques, such as discrete HMMs, which require discrete input values.

The approach in this phase is to analyze two vector quantization techniques. The first is the commonly used method of K-means clustering. The second is a probabilistic variant on K-means called Expectation Maximization. We will determine in this phase which technique best suits our purposes. This technique can then be used in the classification phase to cluster the input data at frame level or to define GBBs as mentioned in the state of the art paragraph.

1.4.4. Classification

Feature segments → Classified gesture

The goal of classification is to label the incoming feature segments with their correct gesture label. The approach in this phase can also be split up in a few steps.

Step 1 - Selecting classification technique

The first step is to select one classification technique out of the options that were mentioned in the state of the art survey.

Step 2 – Classification analysis

The second step is the analysis of different classification problems. The first problem involves the input representation. Questions like how to apply clustering or how to construct GBBs. The second problem is how to classify the generated feature segments of an entire meeting and find the correct gesture locations.

Step 3 – Determining the test space

The third step of the classification phase is to determine the options we want to test in the test step. These options are divided in the selecting feature subsets that seem most suitable for classification and determining the different classifier parameters.

Step 4 - Testing

The fourth step is to test all the options selected in the test space and determine the performance of the best setup. The approach is divided in three tests. The first test will be used to reduce the options in the test space to a manageable size. The second test will determine the best classification setup and the performance on the manually annotated gestures. The third test will determine the performance on the feature segments generated by automatic segmentation.

Step 5 - Evaluation

The last step of the classification phase is to make an evaluation and give an explanation of the test results and the observations made during testing.

1.4.5. Dividing the tasks

Dividing a large project such as this into two parts is not an easy task. An apparent solution would be to just work separately on different phases. This is however not an option because the phases are sequential, the next phase relies on the previous phases. Therefore we chose to divide the work of each of the phases in two, as the project progresses. This also has advantages, since working on the same subject simplifies the discussion about that subject and makes it easier to give a second opinion. This is, of course, beneficial to the research. This advantage is at the same time a bit of a disadvantage because more time is spent in discussion about a subject. But we believe discussion leads to better ideas and results in the end.

Chapter 2 - Feature selection

This chapter covers the process of selecting gestures from the meeting videos and determining a set of features which describe these gestures. Paragraph 2.1 covers the selection of a set of gestures for future recognition. In Paragraph 2.2 the annotation process of these gestures is described. Following this we examine possible ways to parameterize these gestures with features in Paragraph 2.3. Outlier filtering and smoothing of these features is covered in Paragraph 2.4. This chapter concludes with a selection of useful features using dimensionality reduction in Paragraph 2.5.

2.1. Video analysis

This paragraph covers the process of choosing a set of gestures. First we describe how the meeting data was collected. Then we make an analysis of all occurring movements during a meeting and make a subdivision in certain categories. The last step is selecting the gestures that are meaningful and plentiful enough to recognize.

2.1.1. Meeting data collection

The meeting data used in this project is recorded at the IDIAP institute in Switzerland, for the purpose of serving as test data in a number of meeting related projects such as the AMI project. The IDIAP smart meeting room is equipped with fully synchronized multichannel audio and video recording facilities, for technical specifications see [40].

The layout of the room is shown in Figure 2.1. Two cameras each record a frontal view of two meeting participants, while a third camera films the projector screen and whiteboard, see Figure 2.2. The results are the M4 public scripted recordings [28]. For both the train and test set 30 meetings were recorded resulting in a total of 60 recorded meetings with a total duration of 15 hours.

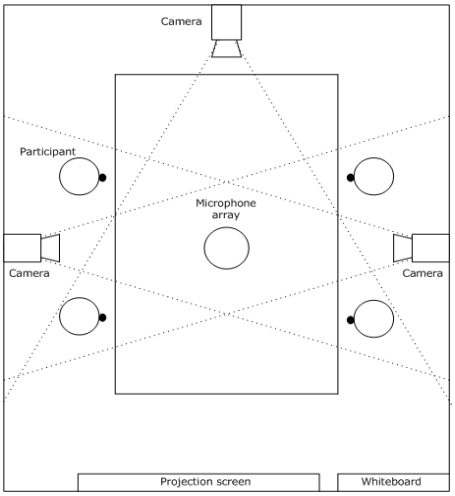


Figure 2.1 - Layout of the IDIAP smart meeting room

Figure 2.2 – Meeting video example

2.1.2. Dividing all occurring movement

The first step in selecting a set of gestures is to identify all different types of movement present in the recorded meeting data. To make this subdivision we looked at a previous annotation proposal by Reiter [48] for gestures and actions in meetings. We also looked at a few meeting recordings and came up with the categories listed below. These categories are structured with respect to the location of the gesture or action. This overview covers almost all the occurrences of human movement present in the meeting recordings.

Hand movement
- Pointing
- Writing
- Voting
- Scratching / touching
- Handling objects (for example a pen)
- Beats (short gestures to emphasize speech fragments)
- Iconic/metaphoric gestures (short gestures to illustrate speech fragments for example the fish was *this* big)
- Writing on the whiteboard
- Wiping the whiteboard clean

Head movement
- Nodding
- Shaking

Body movement
- Leaning forward (on the table)
- Leaning backward
- Leaning with head on hand
- Reposition
- Standing up from the meeting table
- Sitting down at the meeting table

An analysis has been carried out to determine how often these movements occur during meetings. This analysis is based on twelve randomly selected meeting recordings of five minutes long, resulting in a total of one hour of meeting video. The results are summarized in Table 2.1. In the next paragraph we will assess which gestures are meaningful enough to select for classification.

Movement	Occurrence
Pointing	18
Writing	58
Voting	0
Scratching / touching	49
Handling objects	47
Beats	100+
Iconics / Metaphorics	29
Writing on the whiteboard	17
Wiping the whiteboard	7
Nodding	100+
Shaking	23
Leaning forward	16
Leaning backward	27
Leaning on table	25
Leaning with head on hand	41
Reposition	6
Standing up	12
Sitting down	12

Table 2.1 - Different types of movement and their occurrences in one hour of meeting video.

2.1.3. Gesture selection

This paragraph discusses the selection of the gestures that are useful to recognize from the different types of movement observed in the previous paragraph. To determine the useful gestures we use the gesture definition from the introduction, Paragraph 1.1. There we stated that a gesture is a form of movement which has a certain relation with communication. Some of the movements however, that would be useful to be recognized according to this definition, can not be chosen because of a lack of data samples. We will start with discussing the gestures that are chosen and continue with the movements that are left out. The criteria we use to select the gestures are:

- Is there a minimum amount of ten gesture samples present in the meeting recordings?
- Does the gesture give useful information about the meeting?
- Is this information not already sufficiently covered by another input modality of the smart meeting room?

The pointing gesture is very useful to recognize because it indicates that someone is pointing towards something or someone, which could be the unspoken subject of a conversation. An example is the sentence: "this person here is joking". Pointing gestures are very useful to help determine the focus of attention. This gesture is not performed very often, only 18 times in the analyzed hour. It is already annotated in the AMI M4 corpus, which makes it easy to find more samples for classification. However, the vast majority of the annotated pointing examples take place during a presentation. At this moment the feature extraction system we use is not yet able to process the video files of a presentation. Therefore, we have to leave the pointing class out of the segmentation, clustering and classification phases. The available feature data for this gesture is not significant enough, certainly not for training a classification method. We still annotate this gesture for later use when it is possible to process presentation recordings.

Writing is more an action than a gesture. It does not correspond well with the definition of a gesture because by itself it has no communicative intent. However, if someone is writing, this could indicate that the subject matter discussed at that time is important. This makes it is useful to recognize when someone is writing. Writing is performed fairly often and it will be no problem to obtain enough samples.

Beats are useful to recognize because they emphasize the importance of something a speaker is saying. Examples of this are the introduction of someone new into a story or summarizing an argumentation. The beat emphasizes the importance of the thing being said. Iconic and metaphoric gestures illustrate a speech fragment. McNeill [38] states that these gestures reveal a part of the memory image of a speaker and the viewpoint he has taken towards it. All speakers regularly used beats in the analyzed meetings. It was difficult to clearly distinguish the iconic and metaphoric gestures from the beats, without the use of textual information. When you only look at the video information these classes are too much alike. Therefore the beats and iconic and metaphoric gestures are grouped, forming a class of gestures illustrating or emphasizing a part of that what is being said. We will call this class speech supporting gestures, abbreviated as SSG.

The nodding and shaking gestures are also useful gestures to recognize. The nodding gesture indicates for example if someone is listening and agrees with the things being said. The same goes for shaking, but this time someone indicates disagreement. Both nodding and shaking can be used in the construction of an argumentation structure. An application could be to determine the proponents and opponents of a statement in a discussion. Both gestures are performed regularly especially the nodding gesture, so enough samples should be available.

The standing up and sitting down gestures are also useful to recognize, because it indicates if a person is present or not or if someone stands up to write something on a whiteboard for example. Both are performed in every meeting that contains a presentation. Presentations occur regularly in the meetings to give enough samples to classify these gestures.

This brings the total to seven gesture classes. The characteristics of these gestures are described in detail in Appendix A:

1. Pointing
2. Writing
3. Speech supporting gestures
4. Nodding
5. Shaking
6. Standing up
7. Sitting down

The reasons why the other observed movements are not chosen for classification will be discussed briefly. When someone writes on the whiteboard this indicates that something is being told, or written down that is important for all members of the meeting. However in the smart meeting room, writing on the whiteboard is already captured. Recognizing this action will therefore have no additional use. The voting gesture has not been seen once in twelve different meetings of five minutes each. It is impossible to perform machine learning on a gesture of which no substantial set of data samples exist. The same reason can be applied to the reposition and wiping the whiteboard gesture. Scratching, touching and handling objects are quite person-dependent gestures and are not interesting because they do not tell us anything useful about the meeting, or the subject and contents of the meeting. The leaning gestures are also very person-dependent gestures. In the meeting recordings you see people who lean in all possible different directions and people who almost don’t lean at all. We want to look at gestures which are more person-independent, for this reason the leaning gestures are left out.

2.2. Annotation

This chapter describes the different aspects of the annotation process. The seven chosen gesture classes are annotated with the Anvil annotation tool. We have compared three different tools and Anvil matched our requirements. The details of this tool comparison can be found in Appendix B. The rest of this paragraph gives an overview of the annotation guidelines and the inter-annotator agreement on these guidelines.

2.2.1. Annotation Guidelines

The annotation guidelines covered in this paragraph are meant to ensure that all annotators annotate the gestures in the same way. Table 2.2 gives a description of the start and end of each gesture. A special case occurs when the same gesture is performed repeatedly in one flow of movement, without a rest pose. This is annotated as one single gesture. For example, when someone is nodding and performs multiple nods, this is annotated as one nodding gesture.

In addition we add different attributes to the annotated gesture. These attributes can either be general for each gesture or specific for certain gestures. A general attribute is the indication whether a certain gesture is performed clearly or not. The first specific attribute is whether a gesture is performed repeatedly. The gestures which have the repeated attribute are: nodding, shaking and pointing. The second specific attribute is the direction attribute for the pointing gesture. It is not possible to annotate the precise angle of the pointing gesture, but it's reasonably easy to make a division using the eight wind directions. These eight directions are a 2D representation in the frontal plane of the 3D pointing direction from the perspective of the viewer.

Gesture	Begin movement	End movement	Attributes
Pointing	Moving the hands away from a rest position.	Ending with the hands in a rest position.	Clearness Repeated Direction
Writing	Moving hand, head and body from a rest position towards the object on which will be written.	Moving hand, head and body backwards toward the rest position.	Clearness
SSG	Moving the hands away from a rest position.	Ending with the hands in a rest position.	Clearness
Nodding	Beginning of the up or downward head movement.	End of the last up or downward head movement.	Clearness Repeated
Shaking	Beginning of the sideward head movement.	End of the last sideward head movement.	Clearness Repeated
Standing up	Moving arms backward and body forward.	Ending in a (straight) standing rest position.	Clearness
Sitting down	Begin movement downwards.	Ending with the body in a seated rest position.	Clearness

Table 2.2 – Annotation guidelines with begin movement, end movement and attributes for each gesture.

2.2.2. Inter annotator agreement

The gestures have been annotated by the two authors and both annotated a different set of meetings. Two meetings have been annotated by both annotators, to evaluate the inter annotator agreement. In this paragraph the agreement between the annotators is evaluated to ensure that both sets of annotations are the same. In theory this agreement has to follow from the given annotation guidelines, but in practice these may be explained differently.

The agreement is tested on two levels. The first and most important level is label agreement. When there is no agreement, the training process of the classifier will be severely compromised. The classifier may get contradictory information, resulting in poor learning results and a poor classification performance. The second level is boundary agreement. When there is label agreement we can determine if both annotators agree on the start time and end time of the gesture. This level of agreement is important because the annotation data serves as a reference in evaluating different automatic segmentations.

Label agreement

Label agreement is evaluated by testing if, for a given gesture, there is also a gesture with the same label annotated by the other annotator. Possible outcomes of this test are listed below and illustrated in Figure 2.3:

- Agreement: both annotators agree there is a gesture and agree on the label.
- Insertion: one annotator says there is a certain gesture, whilst the other says there is no gesture.
- Deletion: one annotator says there is no gesture whilst the other says there is.
- Substitution: both annotators agree there is a gesture, but they disagree on the label.

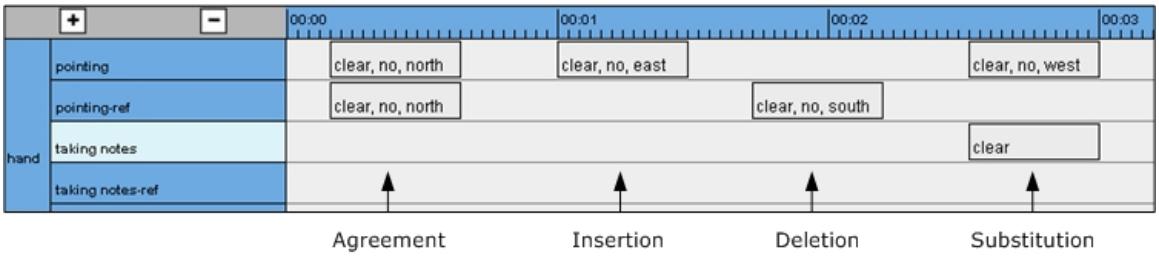


Figure 2.3 – Example of different label agreement test outcomes

The result of this test is the confusion matrix of Table 2.3 containing all seven gestures and an empty category for the absence of a gesture. The diagonal of this Table shows the agreement on the different gestures. The last column and last row show respectively the insertions and deletions. The remaining cells show the substitutions. The Kappa statistic, a chance corrected measure to determine agreement, is calculated on this confusion matrix to evaluate the annotator agreement on labeling. An interpretation of the Kappa values is given by Altman [4]:

- Poor agreement = Less than 0.20
- Fair agreement = 0.20 to 0.40
- Moderate agreement = 0.40 to 0.60
- Good agreement = 0.60 to 0.80
- Very good agreement = 0.80 to 1.00

By just looking at the results you can see that most gestures have good label agreement, the largest numbers are on the diagonal of the confusion matrix. The Kappa value of this matrix confirms this observation. The overall unweighted Kappa value is 0.74 which indicates a good inter annotator agreement on label, according to the interpretation of Altman.

	Pointing	Writing	SSG	Nodding	Shaking	Standing up	Sitting down	Empty
Pointing	15		2					1
Writing		4						
SSG			120					9
Nodding				81				13
Shaking					4			
Standing up						2		
Sitting down							2	
Empty			7	14	1			

Table 2.3 – Confusion matrix result of the label agreement test

Boundary agreement

A gesture annotated by the first annotator should also be annotated by the second annotator at the same location. An obvious approach to test boundary agreement would be to determine if both the start and end location of two annotated gestures match. This approach is not used because we think it cannot be applied to our annotation for the following reasons:

- It is difficult to define a maximum number of frames that boundaries may differ from each other. There is no general rule of thumb for this and the maxima may differ between gestures. Some gestures have very clear boundaries which imply a small maximum number. Other gesture boundaries are vaguer which justifies a higher maximum.
- Judging each gesture separately can give problems with repeated gestures. Take for example the situation in Figure 2.4 where annotator 1 annotates two single points and annotator 2 annotates one repeated pointing gesture. There is perfect agreement on where the nodding starts and where it ends. The annotators just disagree on whether it is one flow of movement. Separate judging would indicate that there is no agreement at all on the boundaries, because the long gesture differs too much from either short one.

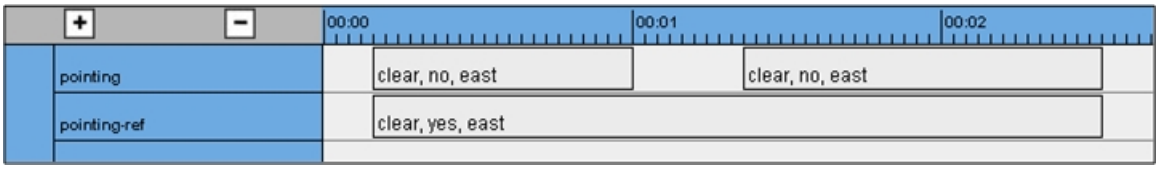


Figure 2.4 – Boundary agreement example

The approach we take is based on the average amount of overlap between gestures. For the cases where there is label agreement we determine the amount of overlap and the amount of disagreement between two gestures. This is done, by counting the number of frames that both gestures have in common and the number of frames where both gestures differ from each other respectively. An average percentage of these numbers is calculated for each gesture class. The higher the overlap percentage the more agreement there is on the gesture boundaries.

Table 2.4 shows the results of the boundary agreement test. For each gesture class the total number of overlapping and disagreeing frames and the corresponding overlap percentages is given. When we use the overlap percentage as a measure for boundary agreement we see some gestures with low agreement. This is especially the case for nodding, standing up and sitting down. Observation of the specific nodding annotations showed that the low agreement on nodding is due to the vague end boundary of this gesture. When someone nods repeatedly, the head movement typically goes on for a while, diminishing in amplitude till it eventually dies out. The point where an annotator decides to end this gesture is therefore vague and differs between annotators. Observations also show that the low agreement on standing up is due to a different interpretation of the guidelines by the two annotators. One annotator annotated only the process of lifting the body from the chair to standing position whist the other annotator included more of the preparatory movements. This is also the case for sitting down. It is wise to take the differences explained here into consideration when using the annotation data for verification in the segmentation phase.

	Overlap	Disagree		Overlap%
Pointing	464	144		76,32%
Writing	2027	157		92,81%
SSG	2896	1020		73,95%
Nodding	2699	1506		64,19%
Shaking	101	27		78,91%
Standing up	83	78		51,55%
Sitting down	133	73		64,56%
Total	8403	3005		73,66%

Table 2.4 – Boundary agreement test results

2.3. Parameterization of gestures

Now we have annotated the different gestures, we want to describe them using certain measurable parameters or features. There are also other ways in which gestures can be described instead of describing them with features. For the interested reader we refer to Noot and Ruttkay [44]. To find suggestions for an appropriate description we will first take a look at the human way of perceiving gestures. Following this we look at the properties and ways of calculating certain features. This paragraph concludes with the features, which can be calculated from the already extracted video features.

2.3.1. The human way

The features humans use to perceive gestures may give some hints for the recognition of gestures by means of machine learning methods. They do not provide strict guidelines and laws that need to be followed. In this paragraph some aspects of the human perception and processing of gestures and gesture typologies are presented. A quote from Dell [19] indicates that humans observe much more features than a simple change in position of a body or body segment.

> *When someone moves, you perceive it as more than a change of place or change in the mover's body shape. Movement does not flow along in a monotone – you see swellings and subsidings, quick flashes, impacts, changes in focus, suspension, pressures, flutterings, vigorous swings, explosions of power, quiet undulations. All this variety is determined by the way in which the mover concentrates his exertion of effort.*

Pollick gives a good overview of the different studies on human perception of movement styles [45]. For example, Johansson [29] created a so called point-light display by filming actors in the dark carrying lights on their joints. This subtracts from all other characteristics of the actor and reduces the movement to a small set of points in motion. How the lights organize themselves into human movement is something that has yet to be solved. Although several different explanations have been offered for the human capability to recognize point-light walkers, none of these explanations gives a thorough and convincing theoretical basis to explain the perception of biological motion. Pollick also mentions that studies investigating brain areas [2, 50] have revealed that a specific brain area in the human superior temporal sulcus (STS) appears to be active when human movement is observed. These studies have also shown that certain brain areas traditionally thought of as solely motoric also serve a visual function.

Once gestures are perceived by humans they are given a certain tag or classification, to give a meaning to the gesture. Different studies have tried to order this classification. Kendon's continuum [33] is widely used to order gestures into different categories: Gesticulation → Language-like Gestures → Pantomimes → Emblems → Sign Languages. As we move from left to right idiosyncratic (personal) gestures are replaced by more socially regulated signs. The spontaneous gestures (Gesticulation) form about 90% of all human gestures [58]. McNeill [38] divides this category in to four subclasses and gives definitions for classification.

Iconic:	representational gestures, depicting some feature of the object, actions or event being described.
Metaphoric:	gestures that represent a common metaphor rather than the object or event directly.
Beat:	a small and formless gesture often associated with word emphasis.
Deictic:	pointing gestures that refer to people, objects or events in space or time.

Besides classification some studies have looked into general properties of gestures. Stephens [54] found that iconic and metaphoric gestures are performed mostly with the dominant hand, as opposed to beats which can be performed by either the left or right hand or both. As mentioned in the introduction of this thesis, McNeill [38] identified different phases within a gesture. He defined the gesture phrase or so called G-phrase. This phrase consists of the following elements: Preparation, Pre-stroke hold, stroke, post-stroke hold and retraction. All phases except for the stroke phase are optional, but the preparation phase is rarely omitted. Functionally the stroke is the content-bearing part of the gesture. The effort used in the preparation and retraction phases is concentrated on reaching a certain rest-point of that phase. The stroke effort is concentrated on the form of the movement itself, for example on the trajectory, shape and posture.

McNeill also presents some instructions for describing hand gestures. Some of those are repeated here for their possible relevance for automatic gesture recognition.

- Describe the motion shape, the place in space where the motion is articulated and the direction of motion.
- Describe if the motion is toward, away or parallel in front of to the side of the body.
- Give the type of direction.
    - Unidirectional, the effort is exerted in one direction.
    - Bidirectional, the effort is exerted in two directions either both hands move in the same way (mirror images) or each hand moves in its own way.
- Bimanual gestures start at the same time but need not to start from the same place and need not to end at the same time.

As we have seen, studies into the human representation of gestures mostly focus on classification labels, properties of gestures and gesture phases. Describing a gesture using these high level features allows for a potentially easy classification. This kind of information is however very difficult to extract out of video data. From a gesture recognition point of view it is more realistic to look at ways to compactly describe a gesture using lower level features. The human way can give hints for these features. The next paragraph discusses a number of features and the properties an ideal feature should have.

2.3.2. Feature studies

There are many different features that can describe human gestures. Some of them are more descriptive than others. A problem arises in selecting the right set of features. This is a difficult task because different occurrences of the same gesture class vary in both space and time. Multiple occurrences of the same gesture may be translated, rotated or scaled. An ideal feature set has to describe all occurrences of a certain gesture class in approximately the same way. On the other hand, gestures from different classes still have to be separated from each other and from noise and non-gestures. So a feature set ideally has to be invariant to the within class variations, whilst ensuring separability between classes. The usefulness of a feature set can be expressed in terms of a few criteria:

- Translation invariance
- Rotation invariance
- Scale invariance
- Contain as much of the available "context" as possible (see below)
- Not susceptible to noise

These criteria can't be fulfilled all at once and therefore a trade-off has to be made. Campbell [9] gives an example: assume you move your hand in a perfect circle. A description of this gesture in terms of curvature and speed, for example, will be rotation and translation invariant. But these measures are constant during the gesture and thus do not contain any context information about where the top or bottom of the circle is. A description in terms of (x, y, z) coordinates does consider this context information but is not rotation and translation invariant. Other features which lie in between these two extremes, such as velocity (δx, δy, δz), trade off some invariance for context. But then again, the derivates used in velocity are more susceptible to noise. In the rest of this paragraph we will take a look at different feature sets proposed in earlier studies on feature selection for gesture recognition.

Position based features

Campbell [9] makes a comparison of different feature sets for 3-D gesture recognition. The different feature sets are tested in combination with a continuous HMM to recognize 18 T'ai Chi gestures. Also their performance is measured under translation and rotation variances. The feature sets tested by Campbell are:

- The Cartesian position of the hands (x, y, z)
- The polar position of the hands with Cartesian z value (r, θ, z)
- The Cartesian velocity (δx, δy, δz)
- The polar velocity with angular velocity $\delta\theta$ term (δr, $\delta\theta$, δz)
- The polar velocity with tangential velocity $r\delta\theta$ term (δr, $r\delta\theta$, δz)
- Two sets with instantaneous speed δs and local curvature ρ, see Equation 2.2 (δs, $\log(\rho)$, δz) (δs, $\log(\rho\delta s)$, δz)

The Cartesian features are relative to a world centered coordinate system. The polar sets are relative to a body centered coordinate system, with the head position as the origin. Speed and curvature are local properties of the paths traced out by the hands. A summary of these features and their invariance properties is given in Table 2.5.

Feature set	Translation invariant	Rotation invariant	Scale invariant	Remarks
Cartesian Coordinates (x, y, z)	No	No	No	Contains most of the context of the original model but is sensitive to translation rotation and scale.
Polar Coordinates (r, θ, z)	Yes	No	No	When the head is the center for polar coordinates this feature set is translation invariant.
Cartesian velocity (δx, δy, δz)	Yes	No	Yes*	*Only scale invariant if the larger movement is made at the same velocity. Sensitive to rotation.
Polar velocity (δr, δθ, δz) (δr, rδθ, δz)	Yes	Yes Horizontal	Yes*	*Only scale invariant if the larger movement is made at the same velocity.
Speed and curvature (δs, log(ρ), δz) (δs, log(ρδs), δz)	Yes	Yes Horizontal	Yes	Noisy due to second derivative ρ. Least amount of context.

Table 2.5 - Summary of different feature sets tested researched by Campbell

The main conclusions of Campbell's research are that feature sets designed to be translation and rotation invariant, indeed cope better with variations in translation and rotation. Cartesian velocity performs better in the presence of translational variations and polar velocity performs better with rotational variations. Higher derivatives such as the curvature suffer from derivative noise, which hinders recognition. Overall the polar velocity set is the best feature set to recognize the 18 T'ai Chi gestures with. Campbell reports 95% accuracy on the test set.

Trajectory based features

When you have a certain feature, such as the position of a hand, you can track that value over time creating a trajectory of that specific feature. Cédras [12] surveyed different ways to parameterize such motion trajectories. The first method uses simple trajectory velocities $v_x(t)$ and $v_y(t)$. These parameters are translation invariant but not rotation invariant and not always scale invariant. Another method is to calculate the speed s_i and direction d_i of a point at time i. These features are calculated as follows:

$$s_i = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \quad d_i = \arctan\left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i}\right) \tag{2.1}$$

Speed and direction are both translation invariant. Furthermore the speed is also rotation invariant and the direction is also scale invariant. The direction component is susceptible to noise, because of the nonlinear arctan operation. A third trajectory representation is the spatiotemporal curvature ρ which is defined as:

$$\rho = \frac{\sqrt{A^2 + B^2 + C^2}}{((x')^2 + (y')^2 + (t')^2)^{3/2}} \quad A = \begin{vmatrix} y' & t' \\ y'' & t'' \end{vmatrix} \quad B = \begin{vmatrix} t' & x' \\ t'' & x'' \end{vmatrix} \quad C = \begin{vmatrix} x' & y' \\ x'' & y'' \end{vmatrix} \tag{2.2}$$

Curvature is translation and rotation and scale invariant. An advantage of curvature is that it describes a trajectory with a single function, as apposed to the two functions for velocity or speed and direction. It does suffer from noise due to the use of a second derivative.

All mentioned trajectory parameterizations use absolute values for velocity, speed, direction and curvature. Cédras suggests that absolute values of motion may be inadequate and that it may be better to look at the relative motion of the segments participating in the movement. For example, the absolute velocity of a body part may be less significant to gesture perception, than the relative velocity between the moving parts of that body segment. Cutting and Proffitt [16] showed that the absolute motion of an object is perceived as the sum of common motion and relative motion. Common motion is the global motion that is shared by all parts of the object. Relative motion is the motion of a part of the object with respect to other parts. Cutting and Proffitt found that relative motion is usually extracted first by an observer and is therefore an important measure for the understanding of the motion.

Human model based features

In a human model-based representation, relative motion can for example be expressed in terms of joint angles between different body parts. These joint angels are another type of feature. They can be derived from a human model representation of the gesture. Angles can also be thought of as being translation invariant because it is a relative measure. When for example the camera is shifted to the right the angles between different body segments will stay the same. Joint angles are rotation and scale invariant as far as the model estimator is capable of compensating for this variation. This is not the case when an absolute position of the hands is taken. However, an absolute position can be made relative by measuring the distance to a fixed point on the human body instead of to a fixed point in the image.

Using a feature as context information

Instead of looking for features that are invariant to variation occurring within a gesture class, one can also take features which describe this variation and use them as context information. Wilson [64] suggests this idea by using a parametric form of HMM to recognize a "family" of gestures. For example, if you make a pointing gesture, relevant context information could be the direction in which you point. A parametric HMM could recognize all the pointing gestures as a single pointing gesture family. This way it is possible to deal with the variance, which occurs for example when pointing in different directions.

2.3.3. Available features

Now that we have seen some possible feature sets and their advantages and disadvantages we will look at features that can be derived from the already extracted video features. The feature extraction tool used in this project is a pose estimation system. The Posio system [46] is a tool that can estimate the human pose from a two dimensional image such as a camera frame. It matches a stick figure with the human form in the picture. This stick figure forms a model of the underlying picture, in terms of position of head and hands and the different joint angles. This research only focuses on a model of the upper part of the human body. In the meeting application area, all of the relevant gestures take place in this upper part. In the model of Figure 2.5 the different segments in black are connected by green joints. Each joint has one or more degrees of freedom (DOF) which describe the possible movement in the joint. The Posio model has a total of ten degrees of freedom, three in each shoulder, one in each elbow, one in the neck and one in the back.

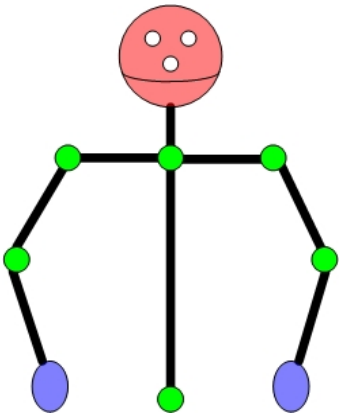


Figure 2.5 - Model of the upper part of the body. The segments (black) are connected by joints (green).

There are a number of features which can be extracted from a static body pose such as the one in Figure 2.5:

- The angles of each DOF in the joints.
- The head and hands position (in world-centered Cartesian / polar coordinates).
- The distances between the head and hands and between both hands.

There are also a number of features which can be extracted from a few transitions from one pose to another:

- The angular velocity for each DOF.
- The angular acceleration for each DOF.
- The velocity of the head and hand movement (Cartesian or polar).
- The acceleration of the head and hand movement (Cartesian or polar).
- The speed and direction of the head and hand movement trajectory.

Finally there are some features which apply to a complete gesture or gesture part:

- Locus, the part of the body where the gesture takes place.
- Intensity of the gesture, based on average velocity or acceleration.
- Duration of the gesture.

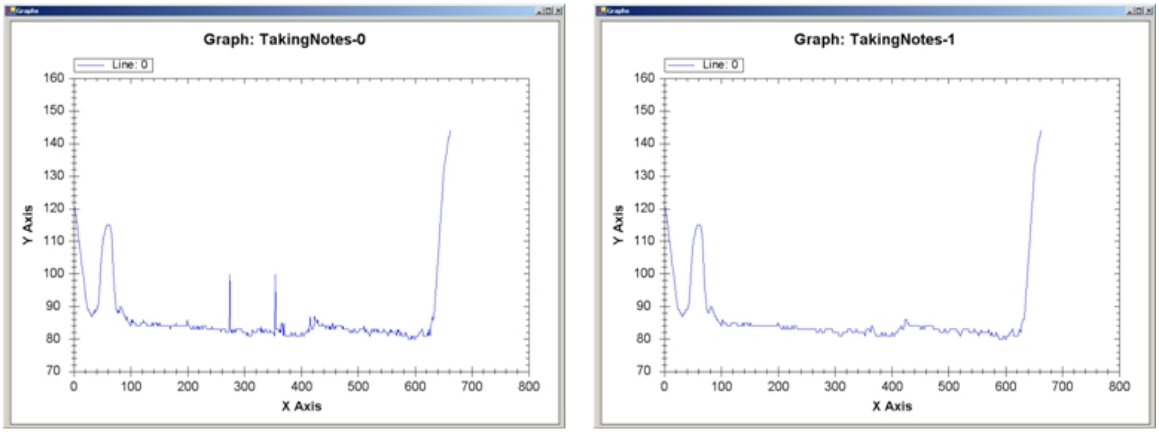
The complete list of available features can be found in Appendix C. All feature representations discussed in this paragraph can be derived from the output of the Posio system. The most important task of the feature extraction module is to estimate these features as accurately as possible. Precise features form the basis for a good recognition. It seems wise to select high level features for their invariance but these features won't be of any use if they cannot be estimated accurately. There are no quantitative results available about the performance of the Posio system. The output features of the Posio system are estimated from the 2d frames of a video. Because they are not directly measured, the head and hand positions and angles between joints are inherently noisy. Therefore we take a look at outlier and noise filtering in the next paragraph.

2.4. Outlier and noise filtering

The feature data extracted from the video images is an estimation of the underlying movement taking place. Estimations however can be wrong, resulting in spikes in the data or high frequency noise. An example is shown in Figure 2.6. This noise should not be present because the human movement which generates the data is a smooth movement. Humans cannot make these sudden changes because of physical limitations. A problem with noisy data arises when you want to make calculations on it. Derivates can amplify the existing signal noise. We use a few signal processing techniques to remove those outliers and smooth the data. These techniques will be discussed below.

2.4.1. Outlier filtering

First of all we want to remove the spikes in the data that clearly do not follow the apparent trend. These spikes can be caused by a wrong estimation (clipping) of the movement taking place, resulting in data values that don't follow the trend of its surrounding data points. These outliers can severely influence derivatives calculated from this data because of their often extreme values. To filter these outliers in the data a common data cleaning technique from the signal processing field is used called median-filtering. This technique looks at a number of data points, surrounding the current data point using a window. From this set of data points the median is computed. This median value replaces the old value of the point being currently examined. This technique results in removing the outlier spikes from the data as can be seen in Figure 2.6. We use a window size of three since this will remove the spikes that that are only one frame in duration, which is typical for a clipping error. A higher window size might result in the removal of useful data.



Original Graph

After median filter

Figure 2.6 - The results of applying outlier filtering

2.4.2. Noise filtering

The second type of noise we want to filter is high frequency noise. This noise can be caused by estimations that are slightly different from the values that they ideally should have. Assume for example, that there is no movement and the neck joint angle should have a "correct" constant value of 10°, for a number of frames. The estimation of this angle might be 9° in the first frame, 11° in the second and 10° in the third, resulting in variations that ideally should not be there.

This high frequency noise can be amplified as said before, when computing derivates from the feature data. We have implemented two data smoothing techniques, to cope with this noise namely: running mean and weighted average. These techniques fall in the category of low pass filters because they keep low frequency variations in the data and filter out the high frequency variations. Both techniques work in the space domain of the data and are based on averaging out the data points, resulting in a smoother signal.

Running mean

The running mean technique applies Formula 2.3. The new value of the current data point is comprised of the already computed value of the previous data point and the average increment measured over a certain window.

$$new_k = new_{k-1} + \frac{1}{window}(old_k - old_{k-window}) \tag{2.3}$$

A parameter on the running mean filter is the window size, over which the mean is calculated. The larger the window size the further you look back in the past to determine how you are going to update the present data point. The larger the window size, the more the data is averaged out. To determine this window size, the difference between gestures with long and short duration comes into play. For long gestures, you want a larger window size to smooth all the insignificant short changes. The window size has to be smaller for short gestures so you won't smooth out these gestures' shorter significant changes. The window size has to have some kind of relation with the length of the gesture. This relation will be empirically determined.

Weighted average

The weighted average technique is based on the recurrent Formula 2.4. Basically the new value consists of a percentage of the old value and a percentage of the new value of the previous point. This previously calculated point is also made up of these two percentages.

$$new_k = (1 - percentage)\, old_k + percentage * new_{k-1} \tag{2.4}$$

The percentage parameter determines how much of the new value is comes from the already calculated value of the previous point (past information) and how much is left of the original data value (present information). Since the weighted average formula is recurrent, the higher this percentage is, the longer the past information will play a role in the calculation of the new value. A high percentage will therefore smooth the data more than a low percentage. This parameter also has to be made relative to the length of the gesture.

Qualitative comparison

Smoothing cannot be tested quantitatively because there is no ground truth on how smooth a signal should be. We made a qualitative analysis on the two smoothing techniques mentioned above using a number of long and short gestures. The results of both methods show a similar smoother signal. We choose to use the running mean method because the linear effect of the window parameter is easier to understand than the recurrent effect of the percentage parameter. The best smoothing result for short gestures is achieved with a window size of 3 and for the long gestures with a window size of 12.

2.5. Dimensionality reduction

After feature selection we have a set of 84 possible features, see also Appendix C. This set is too large for practical use in classification and may contain some redundant features. In order to reduce this feature space we will look into different dimensionality reduction techniques. It is at this point not possible to test which reduction technique results in the best feature set for classification, because there is no classification approach yet. It is possible at this moment to use the inversed results of a dimensionality reduction technique, to pinpoint those features that add almost no information to a reduced feature set.

First a description is given of a number of aspects of dimensionality reduction. Secondly we decide which technique to apply and how to apply it. Following this a selection will be made on the feature set, leaving out those features which contain almost no useful information.

2.5.1. Aspects

There are many dimensions on which different dimensionality reduction techniques can be compared. We have selected a few main aspects. For each aspect a short explanation will be given. The aspects mentioned here should provide enough information to select a technique.

Feature extraction vs. feature selection

There are two main methodologies for dimensionality reduction, feature extraction and feature selection. In feature extraction, not to be confused with the same computer vision term, the original feature set is transformed into a new, smaller feature set. The objective is to retain as much of the meaningful information as possible, whilst the original feature set is discarded. A disadvantage is that the meaning of the original features is also discarded. The transformed features are some sort of combination of the original features and therefore their meaning is not intuitive anymore. In feature selection a subset of the original features is selected which best optimizes one or more criteria. This leaves the meaning intact and this way the features remain interpretable.

Supervised vs. unsupervised

Supervised methods use class information of the features, to select or extract a new feature set. Unsupervised methods on the other hand try to construct a feature set by using aspects of the input data itself. PCA for example is an unsupervised method that uses maximization of the variance of the input data as a selection criterion. More information can be found in the following dimensionality reduction surveys [11, 22]. Since the class information is available from the annotation of the different selected gestures, it is possible to use supervised methods. An advantage of supervised methods is that they can extract features that are useful for classification. These most discriminating features (MDF) are those features that ensure the best discrimination between different classes. Unsupervised methods can only extract the most expressive features (MEF), which retain the largest part of the available information. These most expressive features are not necessarily those features that can easily separate the different gesture classes. The difference between MDF and MEF features is studied by Swets and Weng [55].

Linear vs. nonlinear

A study by Backer et al. [18] shows that nonlinear methods, such as multidimensional scaling (MDS), Sammon's mapping (SAM), self-organizing maps (SOM) and auto-associative feed-forward neural network (AFN), perform better than linear methods such as principal component analysis (PCA) or linear discriminant analysis (LDA), at nonlinear feature reduction. The MDS and SAM methods however perform only well on a data set with a limited number of data points. Since there is a large data set of meeting data available, these two methods are impractical for the current situation. The SOM and AFN perform better on low dimensional data. The feature data obtained from applying image processing to the meeting videos, is reasonably low dimensional, but it is questionable if this dimensionality is low enough to efficiently apply SOM or AFN. This means that linear approaches cannot be left out of the equation.

2.5.2. Selected technique

The technique we selected to reduce our feature dimensionality is LDA. This paragraph describes this technique and how it is applied to determine the non-useful features.

Linear Discriminant Analysis

The linear discriminant analysis technique is a supervised method that uses two evaluation criteria. The first is the between-class scatter of the feature set and the second is the within-class scatter. The between class scatter criterion ensures that the extracted features separate the different gesture classes as good as possible. The within-class scatter ensures that the extracted features group occurrences of the same class close together. This is the first advantage of using LDA, both the MDF and MEF are taken into account. As mentioned before unsupervised methods such as PCA cannot take the MDF into account. Since the class labels are available it is no problem to also evaluate the discriminative aspect of the different features.

As the name suggest, LDA is a linear method. The main advantage of a linear method is the low processing time required to apply such a method. This is another advantage of LDA as opposed to non-linear methods such as SOM or AFN.

Because LDA projects data onto a new feature set it is a feature extraction method. The objective of this paragraph is to select features that are not useful from the feature set. For the segmentation and classification phase it is also more intuitive to work with interpretable features. Feature extraction discards the meaning of the original features. Therefore, the LDA method is used in an inverse way to select features from the original feature set. In order to properly explain how we use LDA in an inverse manner we first explain the standard LDA procedure briefly.

First the input data is normalized using the sample mean and standard deviation. Next the within-class and between-class scatter matrixes are calculated. From these matrices the correlation matrix and the eigenvalues are determined. The eigenvalues are used to calculate the new dimension *d*. These eigenvalues are sorted in decreasing order and transformed into a percentage. Each percentage shows which part the corresponding eigenvalue explains of the total variance. The first *d* eigenvalues are selected, by summing their percentages one by one, until a (large) given percentage is reached. This dimension *d* is used to transform the correlation matrix into a transformation matrix.

Inverse LDA

The first step of inverse LDA is applying the original LDA method to reduce the original feature set to a certain new dimensionality *d*. In this step the transformation matrix and the eigenvalues of this matrix are calculated. The transformation matrix is calculated on the entire data set for all 84 features. This results in a *dx84* matrix.

The second step is to determine the influence of an original feature in the reduced feature set calculated in the first step. This is done by analyzing the transformation matrix and the eigenvalues. The larger the absolute value of an element in the transformation matrix, the more it adds to a new feature value. The columns of the matrix indicate the original features. The rows indicate the new features of the reduced feature set. Each row also has an eigenvalue that indicates what part of the variance the new feature adds to the total variance. By multiplying the eigenvalues with the absolute values of the features in the different columns, a score for each original feature is obtained. This score indicates how much the original feature contributes to the new feature set.

Equation 2.5 illustrates the process. *M* is a *dx84* matrix containing the absolute values of the features in the transformation matrix. The λ indicate the eigenvalues of each row. The vector $\boldsymbol{v}$ $(v_1 \ldots v_{84})$ contains the result scores for each original feature.

$$\begin{bmatrix} v_1 & v_2 & \ldots & v_{84} \end{bmatrix} = \begin{bmatrix} \lambda_1 & \lambda_2 & \ldots & \lambda_d \end{bmatrix} M \tag{2.5}$$

The last step determines which set of the original features contribute enough to the new feature set constructed by applying LDA. By looking at the values in the vector $\boldsymbol{v}$ and evaluating each value against a fixed threshold we can determine which features add enough to the feature set that could be obtained by applying LDA. The new feature set consists of the features that lie above this threshold. In the next paragraph the inverse LDA method is applied to reduce the original feature set of 84 features.

2.5.3. Reducing the feature set

Before inverse LDA can be applied we have to take into account that there are many options to calculate a transformation matrix with LDA. It all depends on the desired dimensionality of the reduced feature set. These different transformation matrices are constructed in the first step where the original LDA method is applied. On our data set this first step generally produces a new feature set with a dimensionality ranging from one to six new features. Another factor is the optional smoothing of the feature data. Since this gives a total of only twelve test cases, six with and six without smoothing, we decided to apply the inverse LDA method to each of them.

The threshold that is used to select which features add too little information to be useful is determined empirically. We have chosen a threshold value of 0.18%. This means that each feature that adds less than 0.18% to the feature set is removed for a certain test case. The value of 0.18% clearly separates a few feature categories.

Results

The results of the twelve tests are given in Appendix C. The results clearly show that the angular velocities and angular accelerations are almost always below the threshold. Other feature sets below the threshold are the polar and Cartesian accelerations of the hands and the head. Only a few features of the polar and Cartesian velocity set lie below the threshold. This is not convincingly enough to discard these categories. The trend is obvious: the features to discard are all the accelerations and the angular velocities. This is a reduction from 84 to 50 features. Intuitively this result could be expected. Higher order derivates such as accelerations amplify the occurring noise resulting in an increase in within-class variation. Also, as mentioned earlier in 2.3.2, derivates trade in context information for invariance. Less context information will result in less between-class variation. Both effects explain why the discarded features add too little information to be useful. Table 2.6 lists the selected feature categories. In the following segmentation chapter and the later classification chapter we select a suitable subset from these categories.

Feature Category
Cartesian head & hand positions
Polar head & hand positions
Joint angles
Distances between head & hands
Cartesian head & hand velocities
Polar head & hand velocities
Speed and direction of head & hand
Duration
Intensity
Locus

Table 2.6 – The selected feature categories to be used in the remaining part of the project. Duration, intensity and locus are three uncategorized features.

Chapter 3 - Segmentation

In the previous chapter we manually annotated the occurring gestures in some of the available meeting recordings. The annotation separates the occurring meeting gestures in time from other parts of the meeting. We can use this manual segmentation result to train and test the actual gesture classification method. But this restricts the program to work with manually pre-annotated meetings. In order to have the program work with plain feature streams, the segmentation has to be automatic. This chapter describes the techniques we researched and their results on automatic gesture segmentation.

In the state of the art overview it is mentioned that explicit segmentation is not strictly necessary when using a state-space approach, such as an HMM, for classification. The idea is that the HMM will stay in a start state as long as no gesture takes place and will progress to an end state when a gesture does take place. So why do we want to research explicit segmentation methods? With a perfect segmentation the classification method only has to decide on the label of the gesture. If the gestures are not explicitly segmented, the classification method also has to determine the location of the gesture in addition to the label. Given the infrequent occurrence of gestures in a meeting, finding the correct location of the gesture implicitly will not be an easy task.

Ideally, we want to segment the gestures as a whole. This means that we want to generate boundaries in the same place as a human annotator would place them. This enables us to present the meeting data in chunks to the classifier. The classifier then only has to decide if the chuck contains a certain gesture or not. However, assuming we can solve this segmentation problem may be too ambitious. The segmentation approach has some pitfalls. Take for example a repeated nodding gesture. The problem of segmenting this as a whole and not stopping after the first one or two nods is not easy to solve. Another example is coping with the long duration of a writing gesture. Not placing any boundaries between the beginning and end of writing may also prove to be difficult.

We also want to research a less ambitious approach to the segmentation problem. From the previous examples we saw that it is reasonable to assume that a gesture is being segmented into smaller parts. In this less ambitious approach we want to allow the presence of these gesture parts as long as the begin and end parts are well aligned with the annotated gesture boundaries. This approach still has an advantage over no segmentation, because we still can present a sequence of gesture parts containing only the gesture to the HMM. The only problem lies in finding the correct sequence of gesture parts.

In this chapter we want to answer the question if we can segment gestures as a whole or in parts from a meeting feature stream. For this we first have to determine the features on which each gesture class is segmented. This selection is made in the next Paragraph 3.1. After this we look at two different segmentation methods in Paragraph 3.2. A method to compare segmentations is described in Paragraph 3.3. The two segmentation methods are tested in Paragraph 3.4, followed by the conclusions of this chapter in Paragraph 3.5.

3.1. Segmentation features

In order to recognize when a certain gesture begins and ends we have to know what characterizes the beginning and end of that gesture. To be able to find these characteristics we recall the annotation guidelines of Paragraph 2.2.1 used for the manual annotation. We use these guidelines as a starting point to determine for each gesture which features describe its boundary characteristics. These features are not necessarily the best features for future classification. For this you need features that describe the entire gesture. The features we are looking for in this paragraph only have to describe the gesture boundaries. For example the head features could be used to detect the boundaries of a writing gesture, because of the significant downward and upward movement during the beginning and end of this gesture. During writing the head generally remains in a fixed position thus providing less information for classification. The segmentation features selected from the categories listed in Table 2.6, are summarized in Table 3.1. A more detailed analysis of the selected segmentation features is provided in Appendix A.

Category Gesture	Cartesian position	Polar position	Joint angles	Cartesian velocity	Polar velocity	Speed / Direction
Writing	Head Y	Head R/D	Head X	Head Y	Head R/D	Head speed
SSG	Left/Right hand X/Y	Left/Right hand D			Left/Right hand D	
Nodding	Head Y	Head R/D	Head X	Head Y	Head R/D	
Shaking	Head X	Head R/D	Head X	Head X	Head R/D	
Standing up Sitting down	Head X/Y	Head R/D	Left/Right shoulder X	Head X/Y	Head R/D	Head speed

Table 3.1 – Selected segmentation features. For the polar sets R/D stands for the radius and delta feature.

3.2. Segmentation methods

In this paragraph we describe the two segmentation methods we use. The idea is to test both methods on whole gesture segmentation and gesture part segmentation. We based our two segmentation methods on previous research into automatic segmentation. Research studies on this topic can be clearly divided into two different points of view: one using a Bayesian information criterion, the other using an activity measure. We will refer to these two approaches as respectively BIC and AM. The rest of this paragraph will give a detailed description of the two methods and how we apply them.

3.2.1. BIC

The BIC approach to automatic segmentation is a probabilistic one. It does not explicitly use any *a priori* knowledge about gesture characteristics to find the gesture boundaries. In short, the segmentation is performed by taking a certain observation sequence and evaluating if it is more likely that this sequence is generated by one underlying process or by two successive smaller processes. The presence of a boundary is implied if it is more likely that two different processes have generated the observation sequence. This approach makes the segmentation problem a statistical model selection problem, using a decision rule to place the boundaries.

BIC is a frequently used decision rule for model selection problems. BIC segmentation has been used in the field of audio segmentation by Zhou and Hansen [68]. Also in the field of speech recognition, BIC is used for speaker segmentation by Tritschler and Gopinath [57]. Recently it has been applied in gesture segmentation by Zobl et al. [70]. The way we apply BIC to our problem is mainly based on this last article.

The first step in the BIC approach is to place a window of size n over a part of the meeting's feature stream. This results in a collection of n feature frames. It is assumed that within this window there is at most one boundary. The next step is to place boundary candidates at each position i within the range $(4,\ldots,n-4)$ of this window. We place the boundaries four frames from the edges of the window to ensure that there is enough data present to the left and right of each boundary candidate to estimate the Gaussian process. We assume four frames, because this number is also used by Zobl et al. in their BIC based gesture segmentation.

To determine which of these candidate boundaries are valid we perform a test for each candidate. The test determines if one Gaussian process generated the observation sequence of the entire window or if two Gaussian processes generated the observation sequences to the left and right of the candidate boundary. Out of the boundaries that are valid, the boundary candidate that scores best on this test is selected to be the actual generated boundary. If none of the candidate boundaries is valid the window slides a certain amount of frames ahead. From there the process is repeated.

The amount of frames that the window slides is specified by the slide parameter. If a boundary is detected the window is also placed a number of frames ahead. This position is determined by adding the value of the slide parameter to the position of the detected boundary. This process is repeated until the end of the entire meeting stream is reached. To summarize: with the window size parameter you define the range in which you expect at most one boundary. With the slide parameter you influence the minimal amount of frames between two subsequent boundaries.

To test whether one process is better than two we have to measure how well a process describes a generated observation sequence. To represent this Gaussian process we use the covariance matrix Σ of the features in the observation sequence. This results in three covariance matrices: Σ_w representing the observation sequence of the entire window, Σ_f representing the observation sequence left of the boundary and Σ_s representing the observation sequence right of the boundary. To reduce the covariance matrices to a single score, the determinant of each matrix is calculated. Each determinant represents the score for the likelihood that the process correctly explains its observation sequence. The lower the determinant, the better the process explains the observation sequence. To be able to compare the different determinant values they are transformed by taking the logarithm of the absolute value. Each of the resulting values is corrected for their model size. When the likelihood score of the two smaller models Σ_f and Σ_s together is lower than the score of the larger model, the data within the window has more likely originated from two different processes. As a consequence, the evaluated position contains a boundary.

The explanation above results in Formula 3.1. This formula is used to test if a boundary candidate is valid at position *i* within the current window.

$$0 < \frac{n}{2}\log\left\|\Sigma_w\right\| - \frac{i}{2}\log\left\|\Sigma_f\right\| - \frac{n-i}{2}\log\left\|\Sigma_s\right\| - \frac{1}{2}\lambda\left(d + \frac{d(d+1)}{2}\right)\log n \qquad (3.1)$$

n	= window size
i	= number of frames before the boundary candidate
n-i	= number of frames after the boundary candidate
d	= dimensionality of the feature vector of each observation
λ	= penalty parameter
Σ_w	= covariance matrix of the observation sequence in the entire window
Σ_f	= covariance matrix of the observation sequence in the first part
Σ_s	= covariance matrix of the observation sequence in the second part

The last term in the above formula is a penalty term which corrects the formula for the feature dimensionality *d*. The lambda parameter can be used to fine-tune this penalty. Suppose the complexity of the segmentation data increases while the dimensionality remains the same, for example when the summed left and right hand feature is used. A higher lambda is necessary in this case to compensate for this increased complexity. Effectively this penalty influences the number of boundaries that are placed. A higher lambda parameter results in fewer detected boundaries.

3.2.2. AM

The AM approach uses *a priori* knowledge of the begin and end characteristics of a gesture. The idea is to find a feature or combination of features that shows these characteristics and to explicitly search for the gesture boundaries. For example, you know that for a speech supporting gesture the hands start and end in a rest position. From this you assume that the velocity feature in these rest positions is close to zero. Gesture boundaries can then be placed by explicitly searching for points, where the hand velocity crosses a certain threshold that is close to zero. Generally, methods originating from this point of view use some form of activity measure to detect gesture boundaries.

The AM method finds gesture boundaries by explicitly searching for points that might indicate a gesture's begin or end. In Paragraph 3.1 we listed the features that show the characteristic begin and end movements of a gesture. We will use these features as our measure of activity. We now have to find a way to search for the characteristic boundaries. Figure 3.1 below shows the trajectory of a certain feature. It also shows four possible categories of interest that might indicate a gesture boundary: local minima, local maxima, zero crossings and threshold crossings

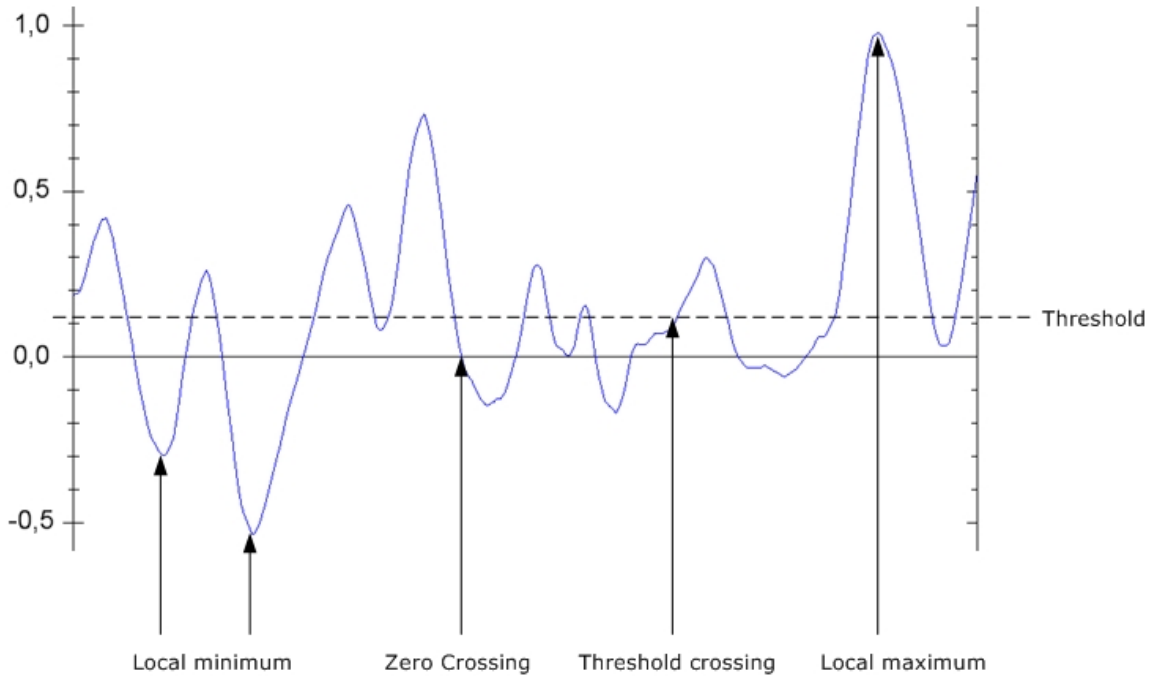


Figure 3.1: Variation of a feature over time showing points of interest.

The basic idea behind the AM method is to search for occurrences of one or more of the categories listed above, in the trajectory of a certain feature. These points of interest might indicate a gesture start or end, as the examples in the following paragraphs make clear.

Local minima and maxima

Local minima and maxima are actually occurrences of the same phenomena. For example let's say that the graph above represents the movement of the head in the *y* direction. In this scenario a local minimum means that the head was moving down and is now at its lowest position before moving up again. The same can be said for a local maximum, only in this case the head was moving up and is now starting to move down again. Both categories indicate a change in direction when you look at position features such as Cartesian or polar coordinates. A writing gesture starts for example with a sudden downward movement of the head. This characteristic change in direction could be detected by looking at points where a local maximum occurs in one of the position features.

A downside to using local minima and maxima is that noise will cause much of these minima and maxima in the data. To limit the resulting set of boundaries found when searching for local minima and maxima a threshold can be defined. This threshold can be used to include only the very high and low peaks in the data. The smaller minima and maxima generated by noise, random movement or smaller gestures such as nodding are left out. This can be useful when searching for the boundaries of gestures with larger amplitudes such as writing, standing up and sitting down. Another possibility is to define a window where only one minimum or maximum may occur and keep only the minimum or maximum with the lowest or highest value within this window.

Zero crossings

Zero crossings in velocity features have a significant meaning. A zero crossing indicates that the velocity of the occurring movement was slowing down is now zero and is going to speed up. This shows a moment of rest in the occurring movement that might indicate a gesture start or end.

Threshold crossings

Threshold crossings can be useful when you have observed, for example that at the end of standing up, the *Y* position of the head always exceeds a certain value whilst during the rest of the meeting this feature is always below this value. By defining a threshold on this value and looking for the points where the *Y* position of the head crosses this threshold you might find the end boundaries for standing up.

It is also possible to define a threshold on velocity features. Consider for example the absolute velocity as a measure of occurring activity. You could define a threshold on the minimum amount of velocity and look for the points where the velocity exceeds this threshold. In this case a gesture begins when the measured activity becomes higher than the threshold and ends when the activity drops below the threshold again.

The resulting set of boundaries can be limited by increasing the threshold on a certain feature. If for example a high boundary on a velocity feature is defined, the detected boundaries are limited to include only the gestures which have high variations in velocity.

3.3. Comparing two segmentations

Before we can test which segmentation approach and corresponding parameter settings gives the best segmentation result, we need a method to compare two generated segmentations. The first step in this process is to compare a generated segmentation with the corresponding annotated segmentation. The method we use for this is described in the next paragraph. The second step is to give an error score to the result of this comparison. The error score for whole gesture segmentation is determined in Paragraph 3.3.2. The error score for gesture part segmentation is determined in Paragraph 3.3.3. Error scores of different generated segmentations can be compared with each other to determine which segmentation is better.

3.3.1. Comparing with annotation

This paragraph gives a description of how we compare a generated segmentation with our own manual annotations. A comparison is made for each annotated gesture, to evaluate how well a generated segmentation matches the annotated gesture. Independent of a whole gesture or gesture part approach, the start and end boundary of the annotated gesture both have to be matched with an automatically generated boundary. We have to consider that it is very unlikely that a generated boundary will be present at exactly the same frame as the annotated boundary. Therefore, we have to decide when a generated boundary is close enough to be coupled to an annotated boundary. A range *r* has to be defined in which the closest generated boundary must lie in order to be coupled to an annotated boundary. With this range we create two situations as shown in Figure 3.2. We call the situation where a boundary is within range a 'match'. We call the other situation where there is no boundary within range a 'deletion'.

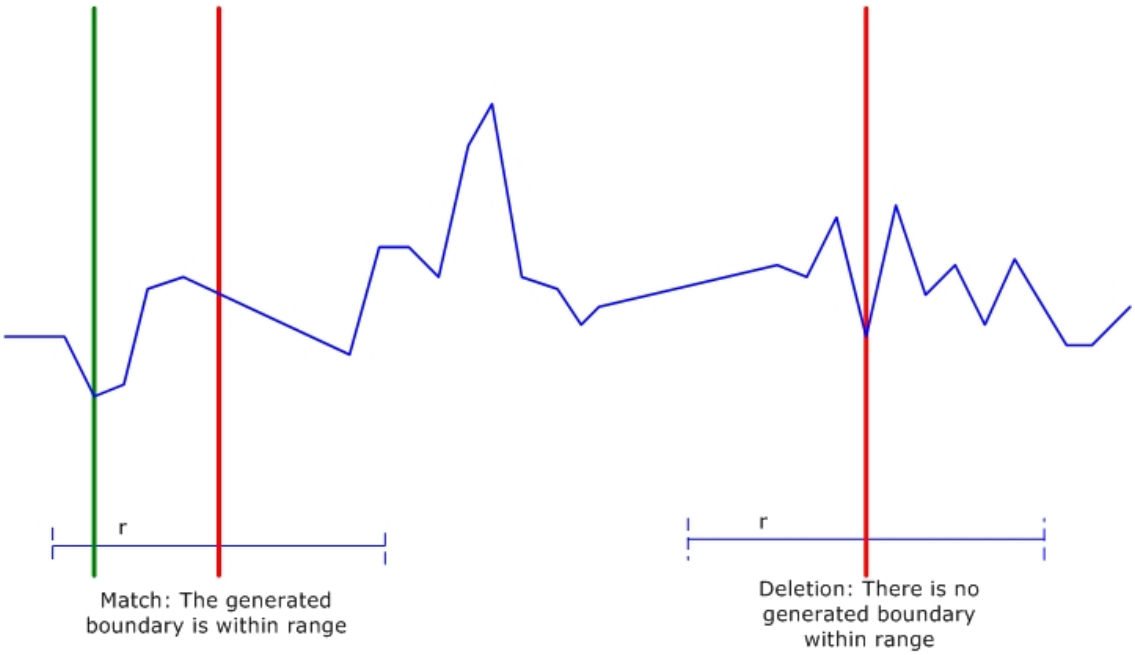


Figure 3.2 - The situation where a generated (green) boundary is within the range *r* of an annotated (red) boundary and the situation where no boundary is within range.

When both annotated boundaries of a certain gesture have been matched correctly with generated boundaries, other generated boundaries in between the two matched boundaries can still make the segmentation unsuccessful. This third situation is called an 'insertion'. Note that we consider boundaries to be an insertion, only if the begin and end boundaries of the currently examined gesture are already matched. When either is not matched we do not speak of an insertion because there is not a possibly, correctly segmented gesture to make the insertion in.

A last situation is caused by boundaries in between the gestures. Because we have not annotated all occurring movement, these boundaries could indicate the presence of gestures that are not considered by us or could indicate any other type of movement. Suppose you are looking for nodding boundaries, at points where the head starts to move down. This method might very well find nodding boundaries but it will most likely also find boundaries of actions that also start with a downward movement of the head, for example looking down at the table. Both gestures have the same begin characteristics, but only the nodding gesture is annotated and looking down is not. The segmentation method cannot be blamed for not being able to make a distinction between these movements. Because we do not have any information about these movements we cannot say if a certain situation would be a match, insertion or deletion. Therefore we will not consider these situations any further. A consequence of this approach is that the uninteresting data between two gestures will be segmented into smaller parts, leaving the classification method to deal with classifying these parts as uninteresting.

3.3.2. Error score for whole gesture segmentation

When using a whole gesture segmentation approach we want the automatically placed boundaries to be only present in the vicinity of the annotated boundaries. This means that no insertions should occur in between the two boundaries that match with the beginning and end of a gesture. So the success criterion is to find as many matches as possible and as little deletions as possible with the least amount of insertions. The rest of this paragraph describes the calculation of an error score which represents this success criterion.

Match score

For the situations where there is a match, we can determine an error score by calculating the difference in frames between the generated and annotated boundary. The formula for the calculation of this score is presented below.

$$Em = Abs(\#FrameAnnotatedBoundary - \#FrameGeneratedBoundary) \quad (3.2)$$

To determine the maximum range, in which *Em* should lie, we use the boundary agreement between annotators described in Paragraph 2.2.2. The amount of disagreement on the location of boundaries gives a good indication of the differences that might occur between two segmentations.

The result of the boundary agreement test describes how much overlap there is between gestures annotated by two different annotators. Table 2.4 in Paragraph 2.2.2 also lists the disagreement between annotators, expressed by the total number of frames that the gestures in the test set did not overlap. By dividing this total disagreement by the number of gestures in the test set we calculate the average number of frames the annotators disagree on per gesture. A gesture always has two boundaries.

To get the average disagreement per boundary, we must divide the disagreement on the whole gesture by two. The resulting value of seven frames is used as the maximum range between the generated and annotated boundary. The calculation is given below.

$$r_{\max} = \frac{3005 / 228}{2} \approx 7 \tag{3.3}$$

Deletion score

When there is a deletion, there is no generated frame within the maximum range r_{max}. Therefore the error score for this situation should at least be greater than the maximum match error which is this maximum range of 7. But we do not know if the closest generated boundary is just out of range or far away. Therefore, we add one to this maximum range as the error score for a deletion.

$$Ed = r_{\max} + 1 \tag{3.4}$$

Insertion score

The last situation of an insertion arises when one or more boundaries are generated in between the matched begin and end boundaries. You could say that it does not matter how much boundaries are inserted because the gesture is already incorrectly segmented with only one insertion. We still choose to calculate a score based on the number of insertions because this way the total error score can reflect whether the number of insertions is diminishing or not. To determine the penalty for an insertion we look at the penalty we determined for a deletion. An insertion has the same effect as a deletion. Both cause an incorrect segmentation of the whole gesture. Because deletions and insertions both result in incorrectly segmented gestures, we should give both the same error score. However, we still want to take the number of insertions into consideration. Taking all this into account we give the following error score to an insertion where *#i* is the number of insertions between two matched gesture boundaries. This results in the same penalty as a deletion, when one insertion occurs, with a small increase for multiple insertions.

$$Ei = r_{\max} + \#i \tag{3.5}$$

Total score

We now have error scores for the three situations that can occur per gesture. These must be grouped together in a single score for the whole segmentation pass. This is done by summing these errors of all gestures together and dividing this score by the total number of counted matches, deletions and insertions. Em_a Ed_b Ei_c in the formula denote the different error scores for each situation and *Nm Nd Ni* respectively the number of matches, deletions and insertions.

$$Ew = \frac{\left(\sum_{a=1}^{Nm} Em_a + \sum_{b=1}^{Nd} Ed_b + \sum_{c=1}^{Ni} Ei_c\right)}{(Nm + Nd + Ni)} \tag{3.6}$$

3.3.3. Error score for gesture part segmentation

The gesture part segmentation approach allows the gesture to be split up into smaller blocks. This means that the generated segmentation only has to segment parts of a gesture. The smaller gesture units could be classified first, before they are combined to form a gesture. A similar approach is taken by Wang et al. [63]. Parallels can be drawn with speech recognition, where the phonemes are also identified first and then used to classify a certain word or phrase. For sign language recognition this approach is applied by Murakami and Taguchi and by Birk et al. [42, 8].

Because a gesture is split up in smaller parts, insertions are allowed to occur. No error score is assigned to insertions. The begin and end boundary of an annotated gesture should however still be matched by an automatically generated boundary. The success criterion for this approach is just to find as many matches, close to the annotated boundaries and as a consequence, have as little deletions as possible.

Match and deletion scores

We calculate the error scores for the matched boundaries and the deletions in the same way as we did in whole gesture segmentation. See Formula 3.2 and 3.4.

Total score

The only difference between this approach and the whole gesture approach is that we ignore the insertions. The score for this approach can be derived from the whole gesture approach Formula 3.6 by simply leaving out the insertions. This reduces the formula to the one presented below.

$$Ep = \frac{\left(\sum_{a=0}^{Nm} Em_a + \sum_{b=0}^{Nd} Ed_b\right)}{(Nm + Nd)} \tag{3.7}$$

The above error score is designed to find the optimal combination of features and segmentation settings that generates the most matches as close as possible to the annotated boundaries. In other words minimizing this error will result in the optimal combination of features with the best matching precision.

Gesture part baseline test

A possible problem with ignoring the insertions might be that the segmentation method just places as much boundaries as possible, to get the most precise matches and the least deletions. As a consequence the average size of the gesture parts may become too small to represent anything significant. To test if the average size of the gesture parts is too small, we devised a baseline test. The idea behind the baseline test is to place boundaries a certain fixed amount of frames apart, resulting in gesture parts with a fixed size. The average size of the gesture parts generated with BIC or AM must be larger than this baseline size.

The baseline test can match all annotated boundaries as long as it places the boundaries close enough together. This is because a boundary is considered to be a match when it lies within the range of seven frames before or after the annotated boundary. In theory this means that if you just place boundaries every 14 frames apart, you can get a 100% match. However this match would not be very precise. For the example of 14 frames the minimum deviation from the annotated boundaries is 0 and the maximum deviation is 7 so the expected precision of a match is an average of 3.5 frames.

To make a fair comparison between the baseline test and the BIC or AM result both must have the same match precision. For example if BIC generates matches with an average precision of 2 frames, the baseline test with the same precision would result in gesture parts of 2*4=8 frames. The average size of the parts generated by the BIC example must lie above these eight frames to be better than the baseline method. Summarized the baseline test approach is as follows. First determine the match precision of the currently evaluated BIC or AM result. Next determine the baseline part size that corresponds to this precision. Finally evaluate if the BIC or AM part size exceeds the baseline part size.

3.4. Testing

In this paragraph the segmentation performance of the two previously described methods is evaluated for the whole gesture and gesture part approach. The purpose of the test phase is to find an answer for each gesture class to the following questions:

- Which segmentation method is better, BIC or AM?
- What are the best parameter settings for this method?
- On which feature or features is this best result achieved?

The best combination of parameters and features is determined automatically, by testing a whole range of different combination of features and parameter settings. The setup with the lowest error score will have the best combination of features and parameters. This is because the error scores for whole gesture and gesture part segmentation are designed to reflect the desired behavior of each approach. The tested parameters and feature setups are described below.

3.4.1. Test setup

To determine the best segmentation method the best BIC result is compared with the best AM result. The approach with the lowest error score is the best segmentation approach.

BIC parameters

When we recall the explanation of BIC in 3.2.1, there are three BIC parameters: a window width, a window slide parameter and the λ penalty value for inserting a boundary. We assume that on a large data set the optimal values for these parameters can be determined independently. This saves a lot of processing and testing time because no combinations of parameters have to be tested. If we for example test 5 values for each of the three parameters we now have to perform 15 tests instead of 125 (5^3). These assumptions are based on preliminary tests which show that the error score determined with independent testing deviates only a few percent from the optimal score possible. This deviation is not significant. We therefore conclude that we can optimize the values for these parameters independent of each other.

AM parameters

In the description of the AM method in Paragraph 3.2.2 we mentioned four different types of boundaries: Local minima, local maxima, zero crossings and threshold crossings. Most these boundaries have additional parameters. The threshold crossings have a threshold parameter indicating the position of the threshold. The detected minima and maxima can also be restricted with a threshold. The minima and maxima also have an optional boundary filter, which selects the lowest local minimum or highest local maximum within a certain window. Also the minima and maxima can be made absolute, when it is required that the maxima must lie above and the minima below zero.

Feature setups

For the BIC method there are three feature setups available:

- The first setup is to determine an individual gesture part and whole gesture score for each single feature that has been suggested in Paragraph 3.1.
- Based on these individual results some of the best features can be combined to form the second test setup.
- The last setup is to sum some logical individual features together, such as the left and right hand for example. The performance is determined on the new summed feature.

For the AM method a similar approach is taken. It is however not possible to test a combination of features because the AM method only works on one dimensional data. Only the first and last setup mentioned in the BIC approach will be applied for the AM method.

All of the tests in this chapter are performed on features which are smoothed, normalized and filtered for outliers. In the next two paragraphs the best test results are given for whole gesture segmentation in Paragraph 3.4.2 and gesture part segmentation in Paragraph 3.4.3. Furthermore, the test results are evaluated to determine if the best results are good enough.

3.4.2. Whole gesture segmentation results

Table 3.2 provides all the answers to the questions we asked in the beginning of this test paragraph, for whole gesture segmentation. For each gesture: which method performs best AM or BIC, on which features is this result achieved and with which parameter settings.

Gesture	Method	Feature set	Parameter settings	
Writing	AM	Sum of: Cartesian coordinate head Y direction, polar coordinate head Radius / Delta	Boundary types: Threshold value: Min/Max threshold: Min/Max filter: Min/Max absolute:	Threshold Maxima Zero Cross 2 0.8 False True
SSG	BIC	Sum of: polar coordinate Delta of the left and right hand	Window size: Window slide: λ penalty:	13 1 2
Nodding	BIC	Sum of: Cartesian velocity head Y direction, polar velocity head Radius / Delta	Window size: Window slide: λ penalty:	24 2 3
Shaking	BIC	Sum of: Cartesian velocity head X direction, polar velocity head Radius / Delta	Window size: Window slide: λ penalty:	16 2 5
Standing up	AM	Sum of: left and right shoulder X angle	Boundary types: Threshold value: Min/Max threshold: Min/Max filter: Min/Max absolute:	Minima Maxima Not applicable 0.8 False True
Sitting down	AM	Sum of: left and right shoulder X angle	Boundary types: Threshold value: Min/Max threshold: Min/Max filter: Min/Max absolute:	Threshold Maxima 2.6 0 False False

Table 3.2 – Summary of the best test results for whole gesture segmentation.

We can also make a few additional observations from these test results. The first observation is that the shorter gestures (SSG, nodding and shaking) use the BIC method to segment the entire gesture. The longer gestures (writing, standing up and sitting down) achieve the best segmentation results with the AM method. This observation may be explained by the characteristics of the gestures. The nodding and shaking gesture can have a repeated pattern within the gesture itself. This means that an AM method that segments on an observable aspect may find boundaries within the gesture, because the observable aspect is repeated within the gesture. These insertions add to the error score. This may explain why AM scores worse than BIC for the small gestures. The larger gestures, however, generally lack this repeated aspect and may therefore be easier to segment with the AM method.

Another observation we can make is that the best results for both BIC and AM are based on summed features. This seems reasonable since a sum of features contains more information to segment on than its separate parts. Suppose that one feature gives the best description of the beginning of a gesture whilst another best describes the end. The sum of these two features would be suitable to segment the whole

gesture on. This observation is in line with the one Rigoll et al. [49] made for the BIC method. They state that the best results can be achieved by using the energy (sum) of each feature vector instead of the feature vectors themselves.

Segmentation performance

Based on Table 3.2 we now know what combination of segmentation method, features and parameter settings results in the lowest error score. But is this lowest error score good enough to say that we can segment the gestures as a whole? To answer this question we estimate the percentage of gestures in the test set that can be segmented as a whole. This performance percentage is obtained by dividing the matched gestures without any insertion by the total number of gestures in the test set. This is approached by Formula 3.8. The number of matches and deletions together form the total amount of gesture boundaries. This number must be divided by two to get to total number of gestures in the test set. The matched gestures are estimated by dividing the matched boundaries by two. Subtracting the insertions from this total will leave the matched gestures without any insertion. Table 3.3 lists the performances of whole gesture segmentation.

$$\left(\frac{Match}{2} - Insertion\right) \Big/ \left(\frac{Match + Deletion}{2}\right) \tag{3.8}$$

Whole Gesture	Match	Insertion	Deletion	Performance
Writing	95	24	63	29,75%
SSG	1224	550	12	10,03%
Nodding	678	226	84	29,66%
Shaking	120	52	0	13,33%
Standing up	14	3	6	40,00%
Sitting down	15	5	5	25,00%
Total (weighted)	2146	860	170	18,39%

Table 3.3 – Performance evaluation of the whole gesture segmentation results. The match and deletion columns list the matched and deleted boundaries. The insertion column lists the number of gestures where one or more insertions occur.

The optimal balance between the insertions and deletions is found for each gesture using the evaluation method described in Paragraph 3.3.2. Despite this optimal balance, the average number of gestures that we can segment as a whole without having any insertions or deletions turns out to be only 18%, as can be seen in Table 3.3. This means that we cannot use the researched segmentation approaches to segment entire gestures.

An explanation for the results lies with the characteristics of the different gestures. The longer gestures have more frames so the chance is higher for an insertion to occur and make the segmentation unsuccessful. This means that writing, standing up and sitting down are difficult to segment as a whole. The nodding and shaking gestures can have a repeated pattern in them. A boundary could very well be inserted between these repeated parts. The SSG class is very diverse, which makes it difficult to find a general setting with which the wide range of gestures can segmented correctly. Each gesture class has pitfalls that make whole gesture segmentation difficult.

3.4.3. Gesture parts segmentation results

In the previous paragraph we saw that whole gesture segmentation is not possible with the approaches we have taken towards automatic segmentation. In this paragraph we look at the results, suggested by our automatic test, for the gesture part segmentation. Table 3.4 below shows the best suggested method, feature set and parameter settings results.

Gesture	Method	Feature set	Parameter settings	
Writing	AM	Head speed	Boundary types: Threshold value: Min/Max threshold: Min/Max filter: Min/Max absolute:	All 4 types 2 0 False False
SSG	AM	Sum of: polar velocity Delta of the left and right hand	Boundary types: Threshold value: Min/Max threshold: Min/Max filter: Min/Max absolute:	All 4 types 1 0 False False
Nodding	AM	Sum of: Cartesian velocity head Y direction, polar velocity head Radius / Delta	Boundary types: Threshold value: Min/Max threshold: Min/Max filter: Min/Max absolute:	All 4 types 0.2 0 False False
Shaking	AM	Sum of: Cartesian velocity head X direction, polar velocity head Radius / Delta	Boundary types: Threshold value: Min/Max threshold: Min/Max filter: Min/Max absolute:	All 4 types 0.2 0 False False
Standing up	BIC	Sum of: Left and right shoulder X angle	Window size: Window slide: λ penalty:	10 1 1
Sitting down	BIC	Sum of: Left and right shoulder X angle	Window size: Window slide: λ penalty:	10 1 1

Table 3.4 – Summary of the best suggested test results for gesture part segmentation.

For whole gesture segmentation we observed that a summed feature set gave the best results. This is also true for gesture part segmentation. Almost all gesture classes except writing use a summed feature. The same explanation, that a summed feature potentially contains more information to segment on, also applies for gesture part segmentation.

Segmentation and baseline performance

Just as for whole gesture segmentation we look at the performance of the gesture parts approach. The only criterion here is to have the most precise matches and the least deletions possible. When we look at the performance of this approach it is a lot better than the whole gesture performance and results in almost 96% match of a generated boundary with a segmented boundary, see Table 3.5.

Gesture Parts	Match	Deletion	Performance
Writing	156	2	98,73%
SSG	1214	22	98,22%
Nodding	687	75	90,16%
Shaking	116	4	96,67%
Standing up	20	0	100,00%
Sitting down	20	0	100,00%
Total (weighted)	2213	103	95,55%

Table 3.5 – Segmentation results for gesture part segmentation. The performance percentage is obtained by dividing the match by the sum of match and deletion.

In Paragraph 3.3.3 we argued that a possible problem with ignoring the insertions might be that the segmentation method just places as much boundaries as possible, to get the most precise matches and least deletions. As a consequence the average size of the gesture parts may become too small to represent anything significant. To test this hypothesis we suggested a baseline test in that paragraph. This test gives a lower limit to the average part length. The length of the parts generated by the segmentation method should be at least larger than this baseline. We determined the part length with the settings of Table 3.4. The results are given in Table 3.6, which gives the average part length and the baseline part length. The baseline average part length is based on the precision of the AM or BIC segmentation.

Gesture	Average part length	
		Base
Writing	4	6
SSG	5	4
Nodding	3	5
Shaking	3	5
Standing up	10	6
Sitting down	10	7

Table 3.6 – Average part length using the suggested results of Table 3.4. The Base column is the result of the baseline method.

The first four gestures, which are segmented with the AM method, have a low average part length. Especially for writing, nodding and shaking where the part length is even lower than the baseline length. This results in a high average number of parts per gesture and a high number of total generated bounds. For writing, nodding and shaking the average number of parts and number of generated boundaries would be even lower when you use the baseline segmentation method. The score for speech supporting gestures is also only marginally better than the baseline test. In other words ignoring the insertions leads to too small gesture parts for writing, nodding and shaking.

Re-evaluation

In order to increase the average part length for the writing, speech supporting, nodding and shaking gesture, the settings for the best AM result and the best BIC result are re-evaluated. The results for standing up and sitting down show that the segmentation with BIC generates larger gesture parts. This indicates that BIC in general might generate fewer boundaries. Therefore we re-evaluated not only the best AM result but also the best BIC result. The objective is to decrease the number of generated boundaries and thereby increase the average part length, while the percentage of matched boundaries does not decrease significantly. The AM setting with which the number of generated boundaries can effectively be reduced is the boundary type selection. An example is to leave out all the threshold crossings. For BIC, the penalty parameter directly influences the number of boundaries. The penalty value is increased to reduce the boundaries generated by the BIC method.

The scatter plot of Figure 3.3 shows the result of this fine-tuning evaluation for writing. The yellow point belongs to the suggested setting of the automatic evaluation method. Although this point does have the highest match percentage it also has the highest number of generated boundaries, which resulted in the low average part length for writing. The red and blue points are the test results of the fine tuning test for AM and BIC. Most tests result in significantly fewer generated boundaries whilst still having a high match percentage. The green point indicates the chosen fine-tuned setting. The same approach has been used for speech supporting, nodding and shaking.

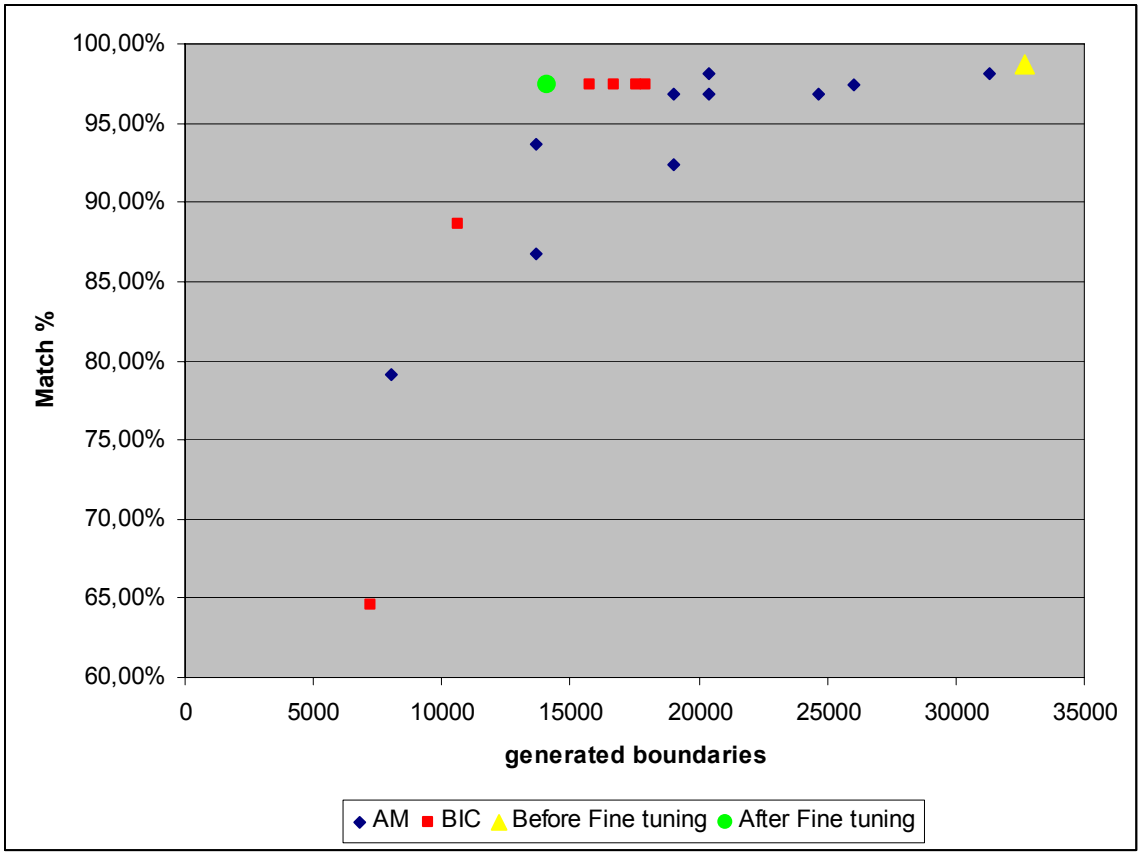


Figure 3.3 – Scatter plot of the fine-tuning test results for writing. The yellow point is the best result before fine-tuning. The green point is the best result after fine-tuning

Table 3.7 lists all the fine-tuned settings and Table 3.8 shows the new average part length compared to the baseline length. With the fine tuned settings, all gestures perform better than their corresponding baseline tests. The average part length has increased significantly for writing, speech supporting, nodding and shaking. There is only a decrease of 0.04 percent point in performance compared to the old total segmentation performance given in Table 3.5. The new total performance score is 95.51%. The indication that BIC generally produces larger gesture parts for a similar performance, proved to be true. The BIC method with the parameters from Table 3.7 will be used to segment gesture parts for all gesture classes.

Gesture	Method	Feature set	Parameter settings	
Writing	BIC	Sum of: Cartesian head Y position, polar head Radius / Delta	Window size: Window slide: λ penalty:	10 1 6
SSG	BIC	Sum of: polar position Delta of the left and right hand	Window size: Window slide: λ penalty:	13 1 6
Nodding	BIC	Sum of: Cartesian head Y position, polar head Radius / Delta	Window size: Window slide: λ penalty:	12 1 7,5
Shaking	BIC	Sum of: Cartesian velocity head X direction, polar velocity head Radius / Delta	Window size: Window slide: λ penalty:	11 1 6
Standing up	BIC	Sum of: Left and right shoulder X angle	Window size: Window slide: λ penalty:	10 1 1
Sitting down	BIC	Sum of: Left and right shoulder X angle	Window size: Window slide: λ penalty:	10 1 1

Table 3.7 – Summary of the fine-tuned test results for gesture part segmentation.

Gesture	Average part length	
		Base
Writing	10	8
SSG	12	9
Nodding	11	10
Shaking	9	9
Standing up	10	6
Sitting down	10	7

Table 3.8 – The fine tuned average part length.

3.5. Segmentation conclusion

In this chapter we evaluated the performance of two segmentation methods, BIC and AM, on whole gesture and gesture part segmentation. By giving a single score to a generated segmentation we were able to automatically determine the optimal parameter settings for each method and segmentation approach. The test results are summarized below in Table 3.9.

Gesture	Whole gesture performance	Gesture part performance
Writing	29,75%	98,73%
SSG	10,03%	98,22%
Nodding	29,66%	90,16%
Shaking	13,33%	96,67%
Standing up	40,00%	100,00%
Sitting down	25,00%	100,00%
Total (weighted)	18,39%	95,55%

Table 3.9 – Summarized test results

The average performance for segmenting entire gestures is only 18% due to insertions. Given this performance we can conclude that it is not possible to reliably use whole gesture segmentation for the classification phase. We have however determined for each gesture class which features are the most suitable for segmenting entire gestures. The other observation is that we achieve the best results by segmenting the longer gestures using the AM method and the shorter gestures using BIC. See Table 3.2 for the test results of each gesture class.

It is possible to segment gestures in parts. Almost 96% of the generated boundaries are correctly matched with the annotated boundaries. However, using the settings generated by the automatic evaluation method resulted in very small gesture parts. By fine-tuning these settings, the gesture part sizes increased to sizes between nine and twelve frames or 0.36 – 0.48 seconds per part. The segmentation score only decreased slightly by this fine-tuning to 95.51%. A downside of the gesture part segmentation is that the whole meeting is just split up in smaller parts. Although the annotated boundaries are matched there are also a lot of boundaries in between the gestures.

The re-evaluation step also showed that BIC generally produces larger gesture parts than AM on our data. The BIC method is used on all six gesture classes to segment the gesture parts. Another observation is that the best results are achieved on summed features. An explanation for this is that a sum of features contains more information to segment on than its separate parts. The final features and settings we use to segment each gesture are listed in Table 3.7.

All in all we can say that we did not succeed in segmenting gestures as a whole. We can however segment the gestures in parts of decent size with high reliability. This means that in the classification phase we can process the data in parts and we don't have to process the data frame by frame.

Chapter 4 - Feature clustering

This chapter describes the approaches we have taken to cluster the feature data in space. There are several reasons for clustering. In general a reduction from a continuous to a discrete feature space could potentially leave almost all context information intact but reduce the search space drastically. This means that clustered data can follow the trend of the original data whilst leaving out the potentially non-interesting smaller variations. A reduced search space helps to simplify the training of a classifier. Also, a number of classification methods require, or work better, on discrete data.

Clustering will be applied for each class, on a combination of input features in the classification phase. This means that data points, discussed in the clustering approaches, consist of feature vectors. We have chosen to implement and test two different clustering methods, K-means and expectation maximization (EM), which will be described below. This is followed by a short test that compares both methods and a discussion which method is best for our purposes.

4.1. Algorithms

The first algorithm we have chosen is a commonly used unsupervised clustering method K-means, described in Alpaydin [3]. The K-means algorithm finds clusters by choosing *K* data points at random, as initial cluster centers. Each data point is then assigned to the cluster that is closest to that point. Next, each cluster center is replaced by the mean of all the data points that have been assigned to that cluster. The next step is to reassign data points that are now closer to a different cluster center. This process is iterated until no more data points are reassigned. The reason for choosing this algorithm is that it is fast and robust to use.

We also apply a probabilistic method, the expectation maximization algorithm with Gaussian components. The EM method can be seen as a generalized version of K-means clustering. The main difference is the distinction between hard versus soft memberships. A hard membership is adopted in the K-means algorithm where a data point is assigned to only one cluster. This is not the case with the EM algorithm, which uses a soft membership, where each data point can contribute to multiple clusters. The formulae for the expectation and the maximization step are also described in Alpaydin [3]. Expectation maximization does have its disadvantages. It generally takes longer to converge to a stable solution than K-means and the chance that the clustering won't converge at all is present for EM, whilst K-means always converges. Also, the EM method has to calculate a covariance matrix for each cluster. The difficulty of this calculation increases when more clusters are used or when clustering is applied on a more complex data set. This makes the EM method less scalable.

A possible cluster difficulty is that different gestures cause very different variations in the data. Nodding gestures for example cause small variations in the data but do happen often. Standing up gestures cause large variations in the data but do not happen so often. Both gesture classes have to be represented by their own clusters. It might be possible that the larger gestures overshadow the smaller gestures resulting in fewer clusters for the smaller gestures and more clusters for the larger gestures. Because EM uses soft label memberships it may be better suited to cope with this difficulty. This problem cannot be solved with normalization because the

different variations are present in the same feature. Normalization can bring different features in the same range but does not change the relative differences within a single feature. In the next paragraph we will evaluate how EM and K-means perform on our dataset.

4.2. Testing and conclusions

To evaluate the performance of K-means versus EM we looked at how good the generated clusters approach the original data for different cluster sizes. This performance is measured with the summed city block error (Manhattan distance), being the sum of the absolute difference between the data points before and after clustering. The Manhattan distance does not take a squared distance, making it less susceptible to single large differences (outliers). The lower the reconstruction error, the better the clustering result approaches the original data. The results of these tests are shown in Figure 4.1. The more clusters you add the more precise the clustering method can approach the data, resulting in a lower error score. However, at a certain point, adding more clusters will not result in a significantly lower error score. To find the optimal amount of clusters one has to look for the dent in the error graph. This is the point where the decrease in error significantly decreases, the so called elbow criterion. Because the feature sets to cluster are determined in the classification phase, we cannot test the effect of different cluster sizes on the classification performance at the moment. The optimal number of clusters has to be determined in the classification phase.

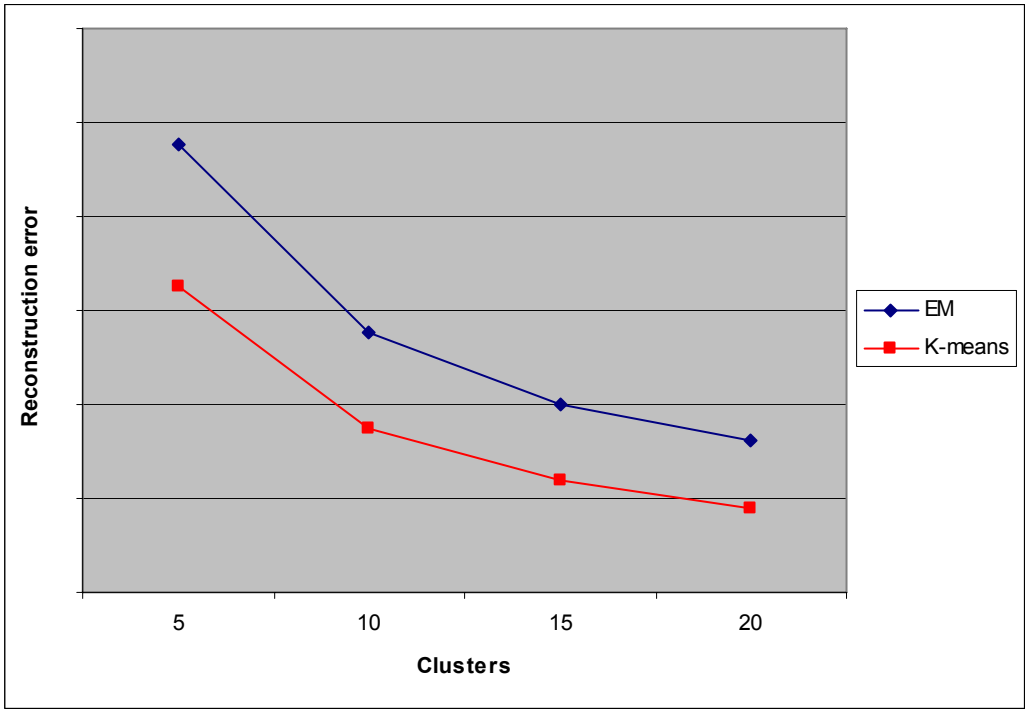


Figure 4.1 – Example of reconstruction errors of EM and K-means clustering for different cluster sizes. The original data which was clustered consisted of one feature namely the summed head features: *Y* position, polar position *R* and *Delta*. Using the dent method one can say that the optimal number of clusters is 10 in this case.

The example graph of Figure 4.1 shows that K-means clustering has a lower reconstruction error than EM clustering. K-means therefore approaches the original data better than the EM method. Given this result and the advantages of K-means on calculation speed and convergence the conclusion would be to use K-means. However this reconstruction error is measured over a whole meeting and doesn't say anything about our concern that the larger gestures might overshadow the smaller gestures. To test if there is reason for this concern we have examined the clustering result on nodding gestures. An example result is shown in Figure 4.2.

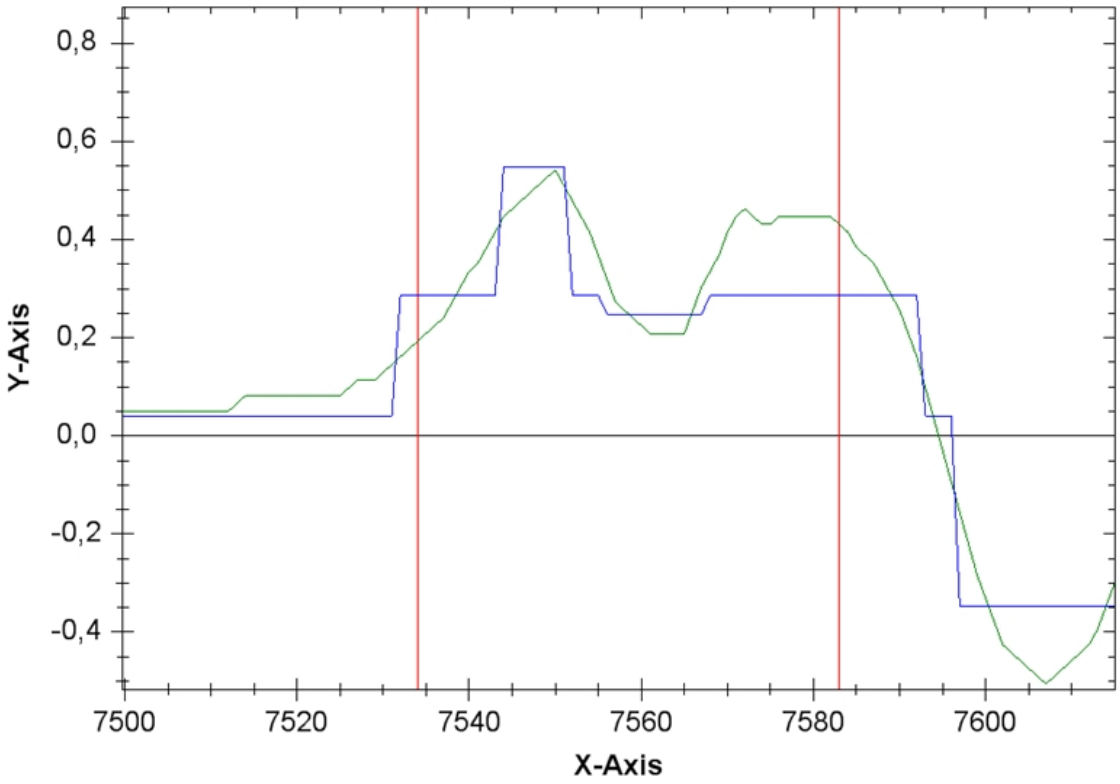


Figure 4.2 – Example of the K-means clustering result on a nodding gesture using 10 clusters. The red vertical lines show the gesture boundaries. The green line shows the original data (summed head features: *Y* position, polar position *R* and *Delta*). The blue line shows the clustering result.

When we look at the graph we see that the concern we had, that this small nodding gesture would be flattened out into one cluster, is unfounded. The trend of the green line is reasonably followed by the clustered blue line. To conclude this chapter on clustering we can say that it is safe to use K-means clustering as the clustering method for the remainder of this project.

Chapter 5 - Classification

In the previous two chapters on segmentation and clustering we concluded that it is possible to partition a whole meeting into small gesture parts. The data in these gesture parts can be left continuous, or made discrete using K-means clustering. The optionally clustered gesture parts form the input for this chapter on the final phase of the roadmap, classification.

First the choice of classifier is documented in Paragraph 5.1. Here we comment on the considerations we had for using an HMM as a classifier. Furthermore the developed HMM toolkit is described in this paragraph. In Paragraph 5.2 we make an analysis of the different problems we have in this classification phase. The paragraph starts with the options we have for presenting data to the HMM. This is followed by the method we suggest on how to find gestures in a partitioned meeting. In Paragraph 5.3 we determine the test space. This is divided into three parts namely the gestures, feature sets and the classification parameters that will be tested. Paragraph 5.4 starts with a test plan on how to test the options of the test space. In the remainder of that paragraph the phases of this test plan and their results are documented. This chapter ends with the conclusion and evaluation of the test results in Paragraph 5.5 and 5.6 respectively.

5.1. Why classify with HMMs

To make a choice for a classification method, we looked at what we want to classify, the kind of data we have at this point and what a classifier should be capable of. We will illustrate our choice for an HMM classifier, based on these considerations. In addition we also take into account that HMMs are the most commonly used method in the literature for different gesture recognition applications

The whole idea of this project is to find and classify the occurring gestures in a certain meeting recording. The logical way to do this would be to identify the data interesting data chunks and let a classifier decide what kind of gesture it is. This approach assumes that it is possible to segment whole gestures from a meeting recording. The conclusions of the segmentation chapter state that it is only possible to reliably divide the whole meeting into smaller segments which may or may not be part of a gesture. Since not all segments are part of a gesture the first aspect of a possible classification approach is that it must be capable of indicating that a piece of data doesn’t contain any gesture at all. Although this is not a distinctive aspect it has to be reckoned with.

The second aspect of a possible classification approach is flexibility. Since we want to classify gestures with very different characteristics it is preferable that we can use a different classifier for each gesture. By this we mean not so much a totally different classification approach, but for example a different topology. It would be preferable if a classification approach would provide this flexibility.

A third important aspect is that different instances of the same gesture class can have different durations. This means that we are dealing with data with varying duration. An example of this is the writing gesture where it is possible that someone writes for a short or longer period of time. As a consequence it is necessary that the classification approach is able to cope with variable observation lengths.

A last aspect that limits the number of possible classifiers is the fact that at the start of this project we have decided to work towards a solution that uses machine learning techniques. This leaves for example static template based classifiers out of the question. By altering templates based on the gesture samples you can argue that you can learn the best template. However, the main consideration for using machine learning approaches is that we assume that these techniques can model the varying duration aspect better than templates or similar methods.

Mainly based on the different duration aspect we have chosen to use HMMs for classification. An HMM can deal with variable observation lengths, as we have already seen the state of the art overview (Chapter 1). It also fulfills all other aspects mentioned above. To handle the non-gesture parts it may be possible to use a garbage-HMM or a simple threshold. Also, an HMM is flexible, because you can vary the number of states or the topology of how these states are connected. And last but not least, it is a machine learning technique. In the rest of this chapter the possibilities of HMMs and especially the possibilities we use will be covered more extensively.

5.1.1. HMM toolkit

Before we can use HMMs for classification we first have to make a toolkit that allows us to build one. We have based our implementation on Rabiner's tutorial on HMMs and Alpaydin's book on machine learning [47, 3]. The features that are available in this toolkit are described below. This description assumes a general knowledge of HMMs. We will not describe all the ins and outs of hidden Markov models here. For this we refer to Rabiner's tutorial [47].

The toolkit can be used to construct both discrete and continuous HMMs. This means that the observation sequences used to train and test an HMM can consist of clustered or unclustered data. The HMM parameters we have implemented for these two types are:

- The number of states
- The number of mixtures per state (only for continuous HMM)
- Different state topologies

The number of states influences the modeling capacity of the HMM. It might be possible that more complex gestures such as writing require more states than nodding.

A continuous HMM is a generalization of a discrete HMM. The observation probabilities are calculated using a number of mixtures for each state and the distribution of those mixtures. Each mixture is a multivariate normal distribution with a certain mean vector and covariance matrix. By increasing the number of mixtures per state you influence the capacity of what each state can model. For example with one mixture you can model data with one Gaussian distribution and with two mixtures the data is approached using two Gaussian distributions.

Because the different gestures have different characteristics, we have implemented two HMM topology options: fully connected and left-right. Some of the gestures have a repeated character which requires the topology to be fully connected or at least cyclic. In a fully connected topology you can go from the last state to the first state to model a repeated characteristic. Other gestures which are not repeating could suffice with a left-right topology. This topology is less complex and easier to learn.

Applied extensions

In our toolkit we have applied two extensions which are not commonly found in HMM literature. We describe these two below.

Because some of the gestures, especially writing, can be performed for a relative long time, we have implemented scaling as mentioned in Rabiner [47]. This technique applies normalization on the forward and backward variables used in the forward and backward procedures. The probability multiplications involved in such long observation sequences would otherwise become too small and go beyond a computers number range. As a consequence the output of the HMM for a certain observation sequence is the negative log of the actual probability of this observation sequence. Because you take the negative log, the output of the HMM is no longer a direct probability. Instead, the lower the HMM error score the better this HMM explains the observation sequence.

The restriction on the observation probabilities mentioned in Rabiner's tutorial [47] is also implemented in our toolbox. When a certain alphabet symbol is rarely observed during the training phase, the observation probabilities for that symbol will approach zero. When the same symbol occurs during the testing phase within a certain observation sequence the probability of that observation sequence will be too low, because of the high influence of one very low observation probability. To prevent this from happening we have restricted the observation probabilities within the toolbox to be at least greater or equal to 10^{-4}. Note that when for example 200 symbols are used the restriction takes up 20% of the total observation probability, leaving only 80% to be divided between the most probable symbols. It might be necessary to lower this restriction when the number of alphabet symbols increases.

5.2. Options for classification

The overall objective is to classify the gestures that occur in a meeting. There are a few topics that will have to be discussed, before we can process this meeting data. The first topic is the input representation of the gesture parts constructed in the segmentation phase, followed by an elaboration on the gesture building blocks (GBBs). The other topic discusses how to classify gestures in the data stream of an entire meeting.

5.2.1. Input data options

This paragraph discusses how the feature stream of an entire meeting is partitioned and the options we implemented to represent the data within these parts.

We can use the time indices of the boundaries generated by the segmentation process, to partition the classification feature stream. This has to be done for each gesture separately because each gesture has a different set of features and settings, for segmentation and for classification. After applying segmentation we have six different partitioned classification streams, one for each gesture. The advantage of partitioning is that you can search more efficiently for gestures in the meeting stream. Parts enable us to walk through the data stream part by part instead of frame by frame reducing processing time. The method we use to walk through the gesture parts will be discussed in Paragraph 5.2.3.

We have implemented three options to represent the feature data within the gesture parts. The first option is to classify the data within a gesture part to one label, before classifying the entire gesture. This is a two layer classification approach. When the gesture parts are classified in the first layer they become GBBs. In this case the input for the HMM consists of a sequence of a few GBBs, one for each gesture part. The options on how to construct these GBBs are discussed in the next paragraph.

When the gesture parts are not classified beforehand, the second and third options are to leave the data continuous or make it discrete. To make the data discrete we cluster each feature frame of a gesture part to a discrete label, using the clustering approach described in Chapter 4. When the data is clustered the input for the HMM consists of a sequence of discrete labels for every frame of a gesture. The continuous gesture parts result in a sequence of continuous feature data for every frame of a gesture. To summarize, the input data options are:

- Pre-classified GBBs
- Discrete gesture parts
- Continuous gesture parts

5.2.2. Constructing GBBs

In this paragraph we describe the options we have to label gesture parts and turn them into GBBs. There are several options to create GBBs from the gesture parts. When using GBBs as input, one of the options described here will be chosen to classify the first layer of gesture parts to GBBs. The options are divided in supervised and unsupervised

Unsupervised

The first option is to use the most frequently occurring cluster number in a gesture part as a label for that gesture part. The approach is to cluster each feature frame of a gesture part to a discrete label using clustering described in Chapter 4. The cluster label that occurs most frequently is the label of the GBB. In essence this is a voting method. We assume that each gesture part has a clear main cluster. The reason for this assumption is that the segmentation method, used to construct the gesture parts, segments the data on significant changes. As a consequence we expect that within a part no significant change takes place and most of the data is clustered to one cluster label.

The second option is a direct clustering of the entire trajectory within a gesture part to one cluster, instead of clustering each frame of the gesture part. To do this we have to find a method that can cope with the differences in length of the gesture parts. We choose to represent a gesture part by one feature frame, which is the average of all feature frames in that part. This average feature frame is then clustered to a label which forms the label of the GBB. A problem with using an average feature frame could be that we throw away too much information beforehand resulting in a poor first classification step.

Supervised

In order to use a supervised labeling approach, a set of gesture parts has to be manually labeled with a set of predefined labels. These annotated gesture parts can be used with a machine learning method to learn the labeling of gesture parts to GBBs. The labels assigned to the gesture parts can also have a semantic meaning. This semantic information could aid the classification. You can label the gesture parts of nodding for example with “head up” and “head down” labels.

The advantage of a supervised method is the possibility to test the classification step of gesture parts to GBBs explicitly. This ensures that this first step performs optimally before the result is used in the following step, of classifying gestures from these GBBs. The downside is that the current annotation doesn’t suffice for such a supervised learning approach. In the current annotation only whole gestures are annotated, not gesture parts. To annotate gesture parts you have to define new gesture part labels and annotation guidelines. One problem is how to tell what labels to use. Another problem is the actual labeling because the movement taking place in the parts isn’t always very clear. This could lead to different interpretations between annotators. Next to annotation the problem extends to segmentation. The parts generated by the automatic segmentation method may not have the same semantic meaning as the annotated parts. We think that the manual labeling of the parts could turn out to be too difficult so that a supervised approach eventually performs worse than an unsupervised approach. Given the problems mentioned above, we choose not to consider the supervised approach any further.

5.2.3. How to process an entire meeting

At this stage we have a meeting partitioned in consecutive gesture parts. The HMM classifier will be trained on whole gestures because that is what we annotated. Since gestures cannot be segmented as a whole, there is a discrepancy between the smaller gesture parts and the objective to classify entire gestures. Therefore we have to come up with a solution on how to present a set of gesture parts to a classifier.

Gesture size chunks

The classification stream of a certain meeting could be offered to an HMM as a whole, leaving the HMM to find out which parts together form a gesture. This would leave the HMM to cope with a large amount of redundant information, because gestures do not occur frequently during a meeting. It would be better to present the entire classification stream to the HMM in chunks of gesture size. This simplifies the classification task to the decision whether a certain chunk of gesture parts is a gesture or not.

Because the typical length differs between the gesture classes, different chunk sizes are needed for each gesture class. The length within a gesture class is also too variable to have a fixed chunk size for each gesture class. This means that we have to vary the chunk size for each gesture between a certain minimum and maximum size. These sizes are determined by taking the minimum and maximum length of the entire set of annotated gestures. 10% of the largest and smallest gestures are left out of this set to get a more average minimum and maximum. Table 5.1 lists the results. As a result of this measure the exceptionally short and long gestures can still be classified correctly but not as accurate anymore. The end performance will not be compensated for this.

Gesture	Minimum	Maximum
Writing	104	422
SSG	12	35
Nodding	15	73
Shaking	14	75
Standing up	41	70
Sitting down	44	74

Table 5.1 – The minimum and maximum chunk size, in number of frames.

Sliding and expanding window

In order to obtain the different chunks from the partitioned classification stream we use a sliding and expanding window. Depending on the chosen input data representation, this window expands and slides through the GBBs or gesture parts. This principle is shown in Figure 5.1. The window expands from the minimum size in a number of expansion steps to the maximum size in order to capture different gesture lengths. Each time the window expands, one ore more gesture parts or GBBs are added to the chunk that is presented to the HMM. When the maximum window size is reached, the window slides forward through the stream. Each time the window slides, it slides one gesture part forward and the window size is reset to the minimum size.

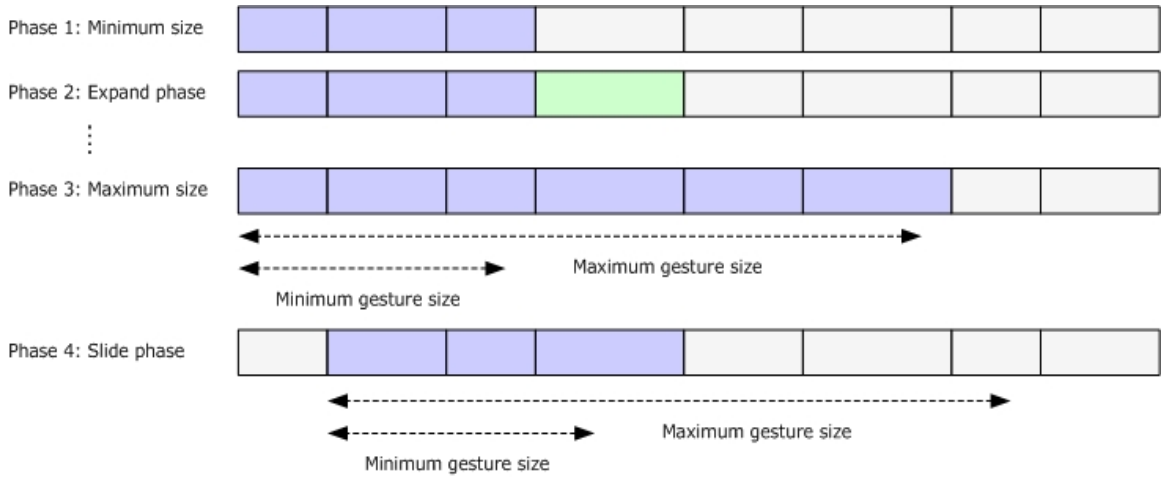


Figure 5.1 – Sliding and expanding window. The blocks are the gesture parts or GBBs depending on the input data representation. Blue indicates the selected parts that form a chunk. Green indicates the part that will be added to the chunk in the expand phase.

Locating the gestures in a meeting

The result after processing a meeting with the sliding and expanding window is a list of HMM error scores generated for each data chuck. These scores are generated from the beginning till the end of the examined meeting. We use a threshold on the error score to determine if an HMM has classified a chunk as a gesture.

Because of the sliding and expanding window the same gesture parts are examined multiple times. Therefore a certain gesture can also be examined multiple times. This can be seen in Figure 5.2. A gesture can be examined partially when the window is not aligned with the gesture (situation 1 and 5). It can be examined entirely but along with other parts (situation 2 and 4), or entirely without any additional data parts (situation 3). This means that not only one error score, but also the surrounding scores should lie under the threshold to indicate the presence of a gesture. When only one score is lower than the threshold we define this as an incident. This could for example be a small piece that resembles part of a gesture. If a consecutive range of lower HMM scores is present as shown in Figure 5.2 (situation 2, 3 and 4), the lowest error score indicates the chunk that most likely matches the annotated gesture.

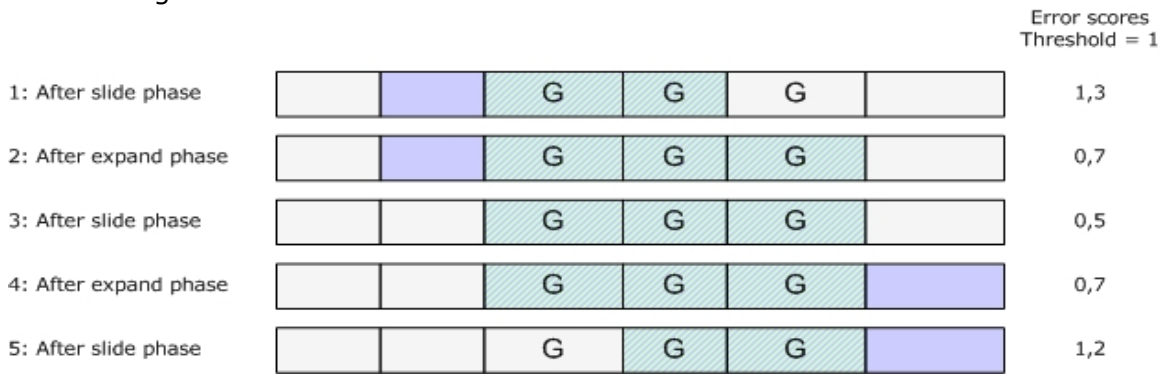


Figure 5.2 – Multiple examinations of the same gesture. The parts that belong to a gesture are marked with G. The colored parts are examined. The numbers are the HMM error scores.

Length compensation

If we want to locate the chunk with the lowest error score a problem arises when we compare the error scores of chunks in one expand phase with each other. Examples of this are situations 3 and 4 of Figure 5.2. In general, if the HMM error scores of two chunks are compared and the gesture parts of the first chunk form a subset of the second chunk, the smaller (first) chunk will always have a lower error than the larger (second) chunk. As a consequence the chunk similar to the minimum gesture size will always have the lowest error score, because it is the smallest chunk of one expand phase.

This under-fitting behavior explained above is a problem for gestures with high length variation. Because of this problem, the classified gesture boundaries will never both be matched with the annotated boundaries of longer gestures. All gestures, which are not similar in size to the minimum gesture size, will therefore not be matched. If a gesture has less length variation the problem is smaller, because more gestures will be similar in size to the minimum gesture size.

An obvious solution for the length problem would be to use a fixed window size equal to the average size of a gesture. When a fixed window is used there is no need to expand from the minimum to the maximum gesture size. The classified gestures will now all have a length similar to the average size of that gesture. However this still is not a sufficient solution for the gestures with a high variation in length. For these gesture classes, the smaller than average and the larger than average gestures, will still be missed.

The solution we applied for the length problem is a form of length compensation on the HMM error scores. With this compensation we can determine which chunk in the expand phase has the lowest error score, given its length. The idea, in terms of probabilities, is to multiply the HMM probability of a chunk with the chunk's length. For example, assume that chunk *X* is twice as long as chunk *Y*. To get equal corrected probabilities, the original HMM probability of chunk *X* may be twice as low as the HMM probability of chunk *Y*. Now it is possible that a longer chunk is selected. Because our toolkit doesn't work with probabilities but with error scores this compensation is also transformed. In terms of error scores the compensation translates to subtracting the log of the chunks length from the original error score to get the corrected error.

Determining the threshold and range parameter

The last classification issue is how to determine when a range of consecutive low error scores is low enough to clearly indicate a gesture. In other words how to determine the threshold and the number of chunks scores that have to lie below this threshold. The optimal settings for these two parameters can be determined by examining the relation between false positives and true positives. Having too few true positives indicates that the chosen threshold may be too low. Having too many false positives indicates that the chosen threshold may be too high. The challenge is to find the threshold where you have a desired balance between true and false positives. The solution to this problem will be discussed in the test Paragraphs 5.4.2 and 5.4.3.

5.3. Test space selection

The selection of the test space for the testing phase of this chapter consists of three parts. First, a subset of the gesture classes is selected for testing. Second, the feature sets that will be tested for the different models are chosen. For this selection the results of Chapter 2 and experiences of the segmentation phase are used. Next the possible values for the different model parameters are considered. Based on the characteristics of the gestures a decision is made for each parameter. This selection of gestures, features and parameters forms the input for the test phase.

5.3.1. Gestures

To limit the time needed for the remaining tests of this classification phase we have decided to select three gestures for the different tests. The first gesture we decided not to test is the sitting down gesture. This gesture shows many similarities with the standing up gesture, so there is little added value in researching the classification performance of both gestures. The nodding and shaking gesture class is also not further researched. During the project it became clear that many nodding and shaking gestures have small amplitudes and are hard to distinguish from other small head movements or noise in the data. Finding a model that is capable of distinguishing between this noise and the gestures will be a very difficult or even impossible task. As a consequence we have decided to focus on the three remaining gestures: writing, speech supporting gestures and standing up.

5.3.2. Features

In this paragraph we select the feature sets we want to test per gesture. To make this selection we first recapitulate on a few of the conclusions made in the feature selection chapter.

In the feature selection Paragraph 2.3.2 we looked at the properties an ideal feature should have. One of the remarks there was that velocity based features should perform better on classification than position based features, because of their invariance to translation and rotation. We want to test if we can also reproduce this observation with our classification results. Therefore we will test each gesture on position features and their velocity counterparts. Another distinction made in Paragraph 2.3.2 is the difference between the Cartesian and polar representation of the position and velocity features. We also want to test the differences between these representations on our classification results. This results in four feature sets:

1. Cartesian position set
2. Polar position set
3. Cartesian velocity set
4. Polar velocity set

In the feature selection chapter we determined different feature sets. We have looked at the variations the gestures cause in these features. Using these observations we have decided whether or not to use them in the position or velocity test sets. Some of the general observations for all gestures are described below. The chosen features per gesture, can be found in the gesture description Appendix A.

The suggested joint angles do show a reasonable amount of variation within a gesture. This is why they were most likely suggested by the LDA method during feature selection. In most cases however this variation is inconsistent for different examples of the same gesture. A certain angle feature does not always show the same behavior for all of the gestures of one class. A reason for this might be that the angles are estimates, derived from a 2d picture of a video-frame using complex vision techniques. Since estimates can be wrong the data from these features will likely be more inconsistent even when the data is smoothed. Another reason might be that joint angles cannot describe different gesture examples of the same class consistently because the angles are just different every time, due to a high within-class variation.

A second notion that became apparent in this and previous chapters is that the speed-direction feature set is noisier than the Cartesian or polar velocity feature sets. The velocities are obtained by taking the first derivative of the position features. The speed and direction features use nonlinear operations in their calculation, making them more susceptible to noise. These noisier features will most likely have a negative influence on the recognition performance, which is also the conclusion of Campbell et al. [9], from their research on features for gesture recognition. Therefore we have decided to leave the speed-direction feature set out of the test sets.

Features that can be measured more precisely and consistent are the position of the hands and head. This notion can also be seen in the selected features for segmentation in Paragraph 3.1. Almost all of the selected segmentation features are hand and head features or their derivates. Because of the better precision of the hand and head features we have decided to use these and their derivates for classification. Table 5.2 summarizes the features we have chosen for each.

Gesture	Cartesian Position	Polar Position	Cartesian Velocity	Polar Velocity
Writing	Left hand X,Y Right hand X,Y Head X,Y	Left hand R,D Right hand R,D Head R,D	Left hand X,Y Right hand X,Y Head X,Y	Left hand R,D Right hand R,D Head R,D
SSG	Left hand X,Y Right hand X,Y	Left hand R,D Right hand R,D	Left hand X,Y Right hand X,Y	Left hand R,D Right hand R,D
Standing up	Head X,Y Root Y	Head R,D Root Y	Head X,Y	Head R,D

Table 5.2 – Classification features per gesture, divided in four sets. In the polar sets *R* stands for radius and *D* for direction.

5.3.3. HMM options

Because it is impossible to test all combinations of HMM parameters we choose to narrow down this test space and only test those options that seem to make sense. For each gesture we have to decide how to represent the input data and what kind of HMM should be used. To do this the following questions must be answered.

Input data representation:

1. Is the data to be classified in the form of discrete gesture parts, continuous gesture parts or GBBs?
2. If the input is GBBs, are they constructed using the main cluster or direct clustering method? (Paragraph 5.2.2)

HMM options:

3. Given the answer to question 1, do you need a continuous or discrete HMM?
4. How many states should the HMM consist of?
5. What kind of HMM topology is likely to perform best?

We try to find an answer for each of these questions by looking at the characteristics of the different gestures. This means that the decision on what to test is not based on hard facts but on experiences from working with the meeting data. A selection of what to test, based on the questions above, is summarized in Table 5.3. The answers to the five questions can be found in the gesture description Appendix A.

Gesture	Input Data	GBB option	Discrete / Continuous	States	Topology
Writing	• GBBs • Discrete GP	• Direct clustering • Main cluster	Discrete	3-6	Left-Right
SSG	• Discrete GP • Continuous GP		• Discrete • Continuous	4-8	Fully connected
Standing up	• GBBs • Discrete GP	• Direct clustering	Discrete	5-9	Left-Right

Table 5.3 – Summary of the test options for classifying the different gestures. The term gesture part is abbreviated as GP.

5.4. Testing

The purpose of this paragraph is to test and select the models that are most suited for classifying the different gestures. This selection is made in three stages which are presented below. Before the tests can be performed, the data has to be divided into three sets. The largest set, the train set, will be used for training the different models. The second set, the validation set, will be used for testing the effects and determining the values of different classification parameters. The last set, the test set, is used for measuring the performance of the selected models and their corresponding parameters. For the exact division of the total sample set see Appendix D.

The purpose of the first test stage is to reduce the test options given in Table 5.3 and the suggested feature sets of Table 5.2. Because the combination of features and classification parameters gives a large set of options, the evaluation cannot be performed by hand. Therefore we use 5x2 cross validation, as proposed by Dietterich [20], on the set of annotated gestures. The reason for using 5x2 cross validation as well as the test and evaluation methods and test results are presented in Paragraph 5.4.1.

The target of the second test stage is to determine the classification performance on isolated samples of each gesture. The performance is measured on the annotated gestures, avoiding the need for segmentation. This phase selects the model and parameters best suited to classify a certain gesture without having to process an entire stream of meeting data. The isolated sample performance is represented by a confusion matrix showing how well a set of gestures can be distinguished from a set of non-gestures. The model and model parameters are selected from the results of the 5x2 test where most options have been removed. A more detailed description of this test stage and the results are documented in Paragraph 5.4.2

In the last test stage we test the performance of classifying gestures in the data stream of an entire meeting, using the method of Paragraph 5.2.3. The model and parameters of the isolated sample stage are used as a starting point to the settings for this streamwise test stage. The results of this test, given in Paragraph 5.4.3, will provide insight in the performance loss of classifying gestures from an entire meeting stream compared to classifying pre-segmented gestures. A measure that copes with the imbalance between negative and positive samples is used to determine this performance loss.

5.4.1. 5 x 2 cross validation test

In Paragraph 5.3 we discussed the test space consisting of four feature sets and the selected HMM options for each gesture class. The feature sets and HMM options can be varied, creating a different classification setup each time. In addition when using discrete input, the number of clusters used to make the continuous data discrete can also be varied. The combination of all these options results in a large test space. It would be virtually impossible to test all these combinations extensively in the isolated sample and streamwise test phases. Therefore, the goal of this cross validation test phase is to reduce the test space as much as possible.

The topology aspect is not varied in this test phase because we selected one fixed topology for each gesture class in Paragraph 5.3. The number of clusters is also not varied because the HMM error scores for different cluster sizes cannot be compared with each other. Reducing the cluster size simplifies the input data because the continuous data is mapped on a smaller number of clusters. This simplification is at the cost of how accurate the discrete data matches the continuous data. As a result the average HMM error score will also decrease because there is less variation in the input data. How much the error score decreases is not known beforehand. This means that we cannot compensate an error score of a 10 cluster input so that we can compare it with an error score of a 20 cluster input. We do test the remaining four classification parameters listed below.

- The input option: continuous gesture parts, discrete gesture parts or GBBs.
- Velocity or position based features.
- Cartesian or polar feature representation.
- The number of HMM states.

To make a selection between the options of these four classification parameters we make a pair-wise comparison. Take, for example, the decision between continuous gesture parts and discrete gesture parts. To make this decision we compare the setup "continuous gesture parts - velocity - Cartesian - 7 states" with its counterpart "discrete gesture parts – velocity – Cartesian - 7 states". This comparison is made for all combinations of the last three parameters. If the setup with continuous gesture parts significantly outperforms discrete gesture parts, we can discard discrete gesture parts as an option for the examined gesture.

In the rest of this paragraph we describe the method we use to test a certain classification setup and how we determine if one setup performs significantly better than another setup.

5x2 cross validation

In order to apply 5x2 cross validation a subset of all the annotated data is needed. In our case we use for this subset the combined train and validation set, see Appendix D. This subset is divided in two sets of equal size. The first set is used to train an HMM using a certain classification setup. The second set is used to test the performance of that classification setup. Then we swap the roles and the second set is used to train the HMM whilst the first set is used to test the performance. This division into two sets is made randomly five times. This results in a total of ten error scores for a certain classification setup. According to Dietterich [20], the 5x2 cross validation method ensures that you get an accurate estimation of the error score for that classification setup.

The first step is to validate if all ten scores give a reasonable estimation of the true error score. We assume that the ten scores are independent of each other and normal distributed. We estimate the sample mean and standard deviation. If all scores give a reasonable estimation of the true error score all values should be close to the estimated mean and the set should have a low variance. This assumption does not hold when a single score varies more than two standard deviations from the estimated mean. In this case a new series of scores has to be calculated.

The second step is to compare two sets of ten error scores of two different classification setups. Comparing two sets of ten scores directly is quite difficult. Therefore we have chosen to compare the means of the scores. Because of the validation made in the first step, the ten scores can be represented by their mean. To make a statistically sound comparison of these two means we use a method described by Tarpey [56]. The suggested method is as follows:

The null hypothesis states that the two means are equal and the alternative hypothesis states that the means are not equal.

$$\begin{aligned} &H_0 : \mu_1 = \mu_2 \\ &H_a : \mu_1 \neq \mu_2 \end{aligned} \tag{5.1}$$

We use a two-tailed t-test statistic on these hypotheses, see Formula 5.2. This statistic is a measure of the standardized difference between the two means. The significance level α we use is 0.05 and the degree of freedom is *n1+n2-2 = 10+10-2 = 18*. The critical *t* value for 0.025 ($\alpha/2$ because test is two tailed) and 18 degrees of freedom is 2.4450. When *t*, calculated using Formula 5.2, is higher than the critical value the null hypothesis is rejected. When the hypothesis is rejected there is a significant difference between the two means of the error scores.

$$t = \frac{\mu_1 - \mu_2}{S_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \qquad S_p{}^2 = \frac{(n_1 - 1)S_1{}^2 + (n_2 - 1)S_2{}^2}{n_1 + n_2 - 2} \tag{5.2}$$

The first formula calculates the *t* value based on the two means (μ_1 and μ_2), their combined standard deviation (S_p) and the number of samples (n_1 and n_2). The second formula calculates the combined variance ($S_p{}^2$) based on the variances of the two separate test series ($S_1{}^2$ and $S_2{}^2$). This combined variance can only be calculated when the two variances of the series are considered equal. In order to test this we perform an F-test on these two variances. The null hypothesis is that the two variances are equal and the alternative hypothesis is that they are not equal.

$$\begin{aligned} &H_0 : S_1 = S_2 \\ &H_a : S_1 \neq S_2 \end{aligned} \tag{5.3}$$

The hypothesis is validated by performing a two-tailed F-test with a significance α of 0.05. The probability of the F-test is based on the F statistic value of Formula 5.4 and the two given degrees of freedom (n_1 and n_2). When this probability lies within the range 0.025…0.975 ($\alpha/2$…1-$\alpha/2$), the null hypothesis is accepted. When the hypothesis is accepted the means can be compared, otherwise we can directly conclude that the two test setups differ significantly.

$$F = \frac{S_1^2}{S_2^2} \tag{5.4}$$

Thus in order to validate whether the difference between two tests is significant, first the variance and then the means are compared. When either the variances or the means aren't equal there is a significant difference between two tests.

Test results

The table below lists the results from the 5x2 cross validation tests for the four tested aspects of classification setup. We choose the best option for a certain parameter when at least 50% of the tested comparisons show a significant difference between the two options. An *X* indicates that no significant difference could be detected between the options of that parameter.

Gesture	Input Data	Position vs. Velocity	Polar vs. Cartesian	States
Writing	Discrete GP	Velocity	Cartesian	X
SSG	Discrete GP	Velocity	Cartesian	8+
Standing up	Discrete GP	Velocity	X	X

Table 5.4 – Test results of the 5x2 setup tests.

The first thing we observe is that the GBBs aren't used for any of the gestures. The GBBs clearly perform worse than the gesture parts. The idea behind using the GBBs was to construct shorter and more generic observation sequences. However, when the contents of a single gesture part can't correctly be matched to a single label, too much context information is lost when the GBBs are constructed. We suspect that a supervised approach for constructing the GBBs, as suggested in Paragraph 5.2.2, may be needed in order to use a two layered classification approach. With the unsupervised approach we took for constructing the GBBs it is not possible to correctly reduce gestures to a more generic description.

The choice between position and velocity based features can easily be made for all gestures. All gestures prefer the velocity based features. A possible explanation for this is that the hand and body movements suffer more from translational variance. This makes it is harder to model a gesture's pattern using position based features. Velocity based features are translation invariant and therefore perform better on these gestures. The choice between Cartesian and polar based features is more difficult to make. For the writing gesture and for the speech supporting gesture the Cartesian velocities perform slightly better than the polar velocities. For standing up it is not possible to make a significant distinction between Cartesian and polar features.

The 5x2 cross validation also gave no conclusive answers concerning the number of states. The only significant claim that can be made from the test results is that from the range of 4-8 states for speech supporting gestures, eight states perform best. It may be possible that speech supporting gestures require more states. This will be tested in the isolated sample test phase.

5.4.2. Isolated sample performance test

As mentioned in the introduction of the testing paragraph, the purpose of this test is to determine the classification performance on isolated samples. This test also determines which model and parameters are best suited for classifying the different gestures. The first paragraph shows how an isolated sample test is performed. After this, the most promising models are chosen, for each model an appropriate threshold is determined and the isolated sample performance is verified on the test set.

Test approach

At this stage we want to know how well a certain model can separate a set containing gestures of one class, from a set of random meeting data of similar size. These random data chunks are called non-gestures. The reason for separating gestures from non-gestures and not from the other annotated gestures is that the ultimate target is to determine where a certain gesture occurs within a data stream. The target is not to decide if a certain chunk of data is gesture *x* or gesture *y*. In the 5x2 test the performance was measured based on the scores of how good a model explained the gestures in the validation set. In this phase we also want to know how a certain model scores on the non-gestures. Therefore, the validation set consists not only of gestures but also of non-gestures of similar size.

The model options that remain from the 5x2 test are those options with no significant differences on the test results of the 5x2 tests. Since the number of remaining options that needs to be examined is relatively small we have chosen to select the most promising model by hand.

Garbage model

After comparing the results of the first few models it became obvious that there were far too many false positives for all models. Analyzing the HMMs and data showed that the different models of the gestures also more or less explain the non-gesture data. Within the gestures there is almost always a phase of minor activity that is also very common in most non-gesture data. These phases are also described as holds by McNeill [38] in his study on the temporal characteristics of gestures. Because these phases are part of a gesture and therefore part of the training data, they are also modeled by the HMM that tries to explain the gesture. As a result, this HMM also explains the non-gesture data quite good. This makes the model less capable of making the distinction between gestures and non-gestures.

The observation of the minor activity phase led to the notion of filtering out all data that contains only common activity (garbage), leaving the data that contains uncommon activity (gestures). To accomplish this, a simple HMM is trained on all the data that contains no gestures. This results in a garbage model that models the common activity during a meeting. Because large amounts of non-gesture data are available it is no problem to train such a garbage model. Initial tests with the garbage model showed that the common activities in the non-gesture data can be explained and filtered out by the garbage model. An example of the effect of a garbage model on speech supporting gestures shows an 82% decrease in false positives at the cost of 12% decrease in correct classifications.

Model selection results

The table below shows the most promising features and model parameters for each gesture. The simple guideline we used to compare and select a certain model is as follows. For each model the number of true positives is set to a fixed number by varying the model's threshold. The model with the smallest amount of incorrect classifications is chosen as the best performing model.

Gesture	Feature set	Data from	States	Topology
Writing				
Garbage model 1	Hand polar velocities	Discrete 20 clusters	10	Fully connected
Garbage model 2	Head polar velocities	Discrete 20 clusters	4	Left Right
Gesture model	Hand polar velocities	Discrete 20 clusters	10	Left Right
SSG				
Garbage model	Cartesian velocity	Discrete 30 clusters	8	Fully connected
Gesture model	Polar velocity	Discrete 30 clusters	18	Fully connected
Standing up				
Garbage model	Polar velocity	Discrete 30 clusters	8	Fully connected

Table 5.5 – The garbage and gesture models and their parameters which are chosen based on the classification performance on the validation set.

The first thing that we should point out is that the writing gesture setup doesn't use the suggested six features of its feature set together. Instead the four hand and two head features are separated. It turned out that two separate garbage models filtered the non-gestures better than one combined model. An explanation for this is that it is easier to train the clustering and the HMM on feature sets with lower dimensionality. Initial tests showed that using the better trained simpler models outweighed the loss of the relation between the hand and head features. For the gesture model only the model trained on the hand features is used. Adding a model based on the head features didn't improve the classification performance on the validation set.

The speech supporting results of the 5x2 tests indicated that this gesture might have needed more than the tested eight states. The isolated sample test results clearly support this case. The best gesture model even has eighteen states. Because of the large increase in the number of states we decided to retest the polar velocity feature set even though this option was already discarded in the 5x2 test. This turned out to be a good decision since the polar velocities outperformed the Cartesian velocities for the gesture model with eighteen states.

The standing up gesture tests with the garbage model alone showed good results. An explanation for this observation is that this gesture causes variations in the features that are very different from the average variations modeled by the garbage model. Because when a person stands up, he moves out of his normal seated position. Therefore, the standing up gestures will have a much higher error score on the garbage model than random non-gesture data. This allows for an easy distinction with a threshold between the standing up gestures and the non-gesture data. Adding

a gesture model didn't improve the classification performance on the validation set. This explains why Table 5.5 only lists a garbage model and no gesture model.

Determining the threshold

Now the models are known but the last parameter, the threshold has not been determined yet. From a set of confusion matrices alone it is hard to determine which threshold ensures the best performance. For this we use an ROC curve as described in Alpaydin [3]. An ROC curve displays the hit rate versus the false alarm rate. The hit rate indicates which part of the positive samples (gestures) is correctly classified as a gesture. The false alarm rate indicates which part of the negative samples (non-gestures) is incorrectly classified as a gesture. Equation 5.5 shows the formulae for the hit rate and the false alarm rate.

$$\text{Hit Rate} = \frac{TP}{TP + FN} \qquad \text{False Alarm Rate} = \frac{FP}{TN + FP} \tag{5.5}$$

The *TP* indicate the true positives, *FP* the false positives, *FN* the false negatives and *TN* the true negatives. How these variables correspond with a confusion matrix is shown below in Table 5.6

Predicted / Actual	Positive	Negative
Positive	True Positives (TP)	False Negatives (FN)
Negative	False Positives (FP)	True Negatives (TN)

Table 5.6 – Example confusion matrix

Figure 5.3 shows an example of a ROC curve. A hit and false alarm rate of 1 corresponds to classifying all positive and negative samples as positive. A hit and false alarm rate of 0 corresponds with classifying everything negative. The optimal point in this graph is the upper left corner, where the hit rate is 1 and the false alarm rate 0. All positives are classified as positive and all negatives as negative. But how do we determine the optimal points on the actual ROC curve? As an example we compare the upper three points with the highest hit rates of the curve. Since the most left one is clearly the closest to the upper left corner of the graph it has the best performance of the three compared points. The four points in Figure 5.3 that show a better performance compared to their neighboring points are indicated by an arrow. Which point and corresponding threshold we should choose is still not clear. Therefore, we have decided to define two scenarios which determine for us what relation between the hit rate and false alarm rate is required. The two scenarios are presented below.

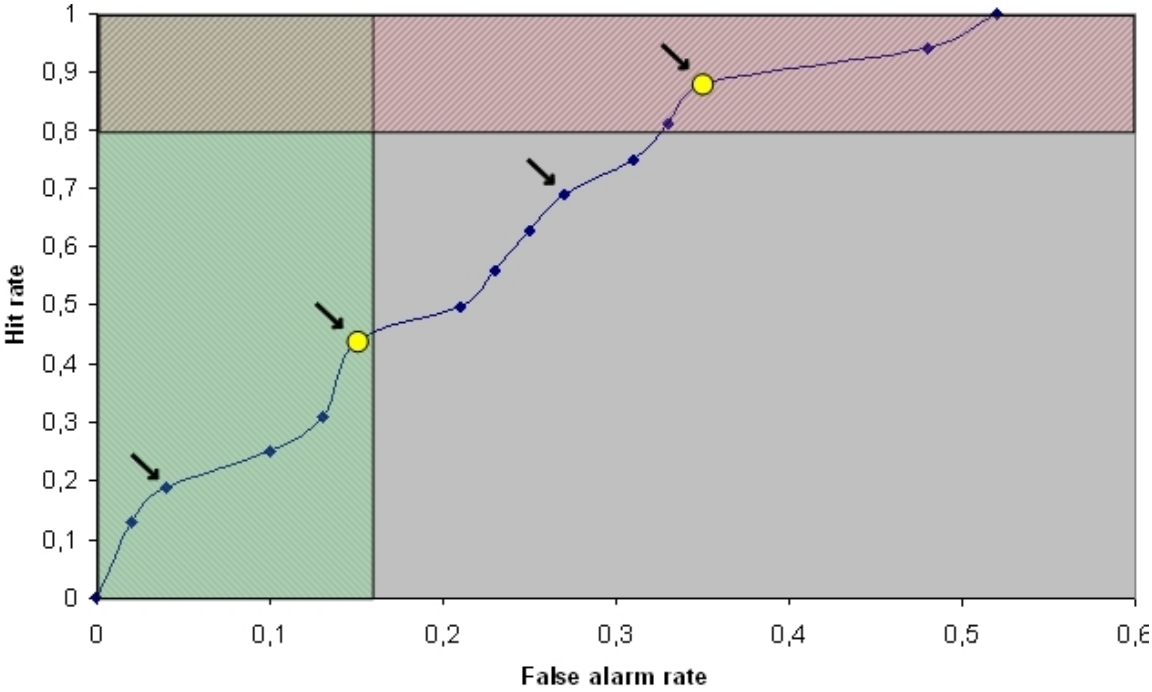


Figure 5.3 – An example ROC curve the shaded green area indicates the low false alarm rate region. The shaded red area indicates the high hit rate region. The two yellow dots indicate the relative best performing threshold settings for the two regions.

Accurate scenario

In an accurate scenario, the costs of false positives are high. When a gesture is recognized the chance that it actually is a gesture should be high. This first requirement means that the gesture should have a low false alarm rate. For selecting the correct point on the ROC curve we first select a region of the graph where the false alarm rate is low enough. Figure 5.3 shows this as the green region. Within this region the yellow point shows the best relative performance. The threshold that corresponds with this point is chosen as the threshold for the accurate scenario.

In the streamwise test, where an entire meeting is processed there is an additional requirement for the accurate scenario. The begin boundary and end boundary of the recognized gesture should lie close to the actual annotated boundaries. The measure we use for deciding when a boundary is close enough is the inter-annotator agreement calculated in Paragraph 2.2.3. The annotator disagreement on a certain gesture defines the range within which the boundary should lie. Table 5.7 lists these deviation ranges for the writing, speech supporting and standing up gesture. The idea behind this is that the automatic classification results may disagree with an annotated gesture as much as the annotators disagreed on it. This restricts the accuracy of this scenario to be at least equal or better than the human observer accuracy.

Gesture	Range
Writing	20
SSG	7
Standing up	20

Table 5.7 – boundary deviation ranges in number of frames for the accurate scenario

Tolerant scenario
In a tolerant scenario, the costs of false negatives are high. The tolerant scenario focuses on a high hit rate, indicating that when there is a gesture it should be recognized. The downside of this approach is of course that the false alarm rate will also increase. Figure 5.3 shows the high hit rate region in red. The threshold that corresponds to the yellow point within this region is selected for the tolerant scenario.

The precision of the match with the annotated begin and end boundary doesn't have to meet precise requirements in the tolerant scenario as long as there is overlap between the annotated and recognized gesture. This means that the recognized gesture should have at least one frame overlap with the annotated gesture. Note that this introduces an additional discrepancy in the streamwise test between the scenarios. The first discrepancy is that the accurate scenario still focuses more on a low false alarm rate and the tolerant scenario more on a high hit rate. The second discrepancy is that in the accurate scenario the match between annotation and classification must meet precision requirements whilst in the tolerant scenario overlap is considered sufficient. This makes a direct comparison of the streamwise performance of the two scenarios more difficult. The measured difference in performance between the accurate and tolerant scenario in the streamwise test will be caused by the two discrepancies between the scenarios. However there is information available from the isolated sample test about the effect of the first discrepancy. With this information we can deduct the performance effect of the second discrepancy.

Isolated sample performance results

Now the correct thresholds have been determined on the test data, the performance of the gesture models and garbage models can be verified. The classification performance for a certain gesture is determined on a test set consisting of gesture and non-gesture data. Below the confusion matrices for the three gestures are presented for the accurate and tolerant scenario.

Writing

Accurate scenario

Predicted / Actual	Writing	Garbage
Writing	10	14
Garbage	1	46

Tolerant scenario

Predicted / Actual	Writing	Garbage
Writing	20	4
Garbage	20	27

The confusion matrix of the accurate scenario shows that there is only one false positive. This is clearly at the cost of the number of correctly recognized gestures. When for the tolerant scenario the number of correct recognitions is increased, the number of false positives also increases significantly.

Speech supporting gesture

Accurate scenario

Predicted / Actual	SSG	Garbage
SSG	48	127
Garbage	1	163

Tolerant scenario

Predicted / Actual	SSG	Garbage
SSG	149	26
Garbage	26	138

Although for the accurate scenario there is only one false positive, the number of false negatives is high. Decreasing this number of false negatives in the tolerant scenario does not introduce too many false positives as we saw with writing.

Standing up

Accurate scenario

Predicted / Actual	Standing up	Garbage
Standing up	4	0
Garbage	0	31

Tolerant scenario

Predicted / Actual	Standing up	Garbage
Standing up	4	0
Garbage	0	31

The confusion matrices for both scenarios of the standing up gesture show no false negatives or false positives. This indicates a perfect separation of gestures from non-gestures. In the next paragraph we will compare the above isolated sample performances with the performances of the streamwise processing test.

5.4.3. Streamwise performance test

In this third test phase we test the performance of trying to find and classify gestures in an entire meeting, instead of classifying pre-segmented gestures. To test this streamwise performance we use the method with a sliding and expanding window as suggested in Paragraph 5.2.3. In the isolated sample phase we tested the performance on pre-segmented gestures. The degradation in performance between this test and the isolated sample test will tell us the impact of not having pre-segmented gestures available.

Re-evaluation of parameters

The first step in the streamwise performance test is the re-evaluation of the threshold parameter for both the accurate and tolerant scenario. We use the values for the threshold parameter determined in the isolated sample test as a starting point and fine-tune these values on the validation set. The reason for this fine-tuning step is that the thresholds established on the isolated samples might not work equally well in this streamwise scenario. In the isolated sample test the threshold parameter is used directly to determine if a certain pre-segmented chunk of data is a gesture. In the streamwise method the threshold is used together with another parameter we call the range parameter. As illustrated in Paragraph 5.2.3 we expect that the data chunks surrounding the chunk with the lowest error score are also chunks with low error scores. The range parameter controls the minimum number of consecutive chunks that must lie below the given threshold in order to classify a certain chunk as a gesture.

Because the threshold parameter is now used in combination with the range parameter we have more options than in the isolated sample test. You can for example take a low threshold and a low range setting. This will find the gestures that have a low error score on the gesture model for a small number of consecutive chunks. You can also increase the threshold and the range setting. In this case more chunks will fall below the higher threshold, but also more consecutive chunks must fall below this threshold. Because of this flexibility we re-evaluated the thresholds established in the isolated sample test. Next to this re-evaluation we also determined the best setting for the range parameter. The approach for determining the best setting for threshold and range is the same approach used in the isolated sample test, namely choosing the best point on the ROC curve for a given scenario as explained in the previous paragraph in Figure 5.3.

Streamwise performance

The next step is to measure the performance on the test set, using the selected garbage and gesture models with their optimal thresholds and range settings. This results in the confusion matrices shown below. For each confusion matrix we discuss briefly the models we used and some observations that can be made from the results.

Writing

Accurate scenario

Predicted / Actual	Writing	Garbage
Writing	0	24
Garbage	0	6238

Tolerant scenario

Predicted / Actual	Writing	Garbage
Writing	10	14
Garbage	18	6220

For the writing gesture we used the writing garbage model as a classifier in both scenarios. This is different from the isolated sample test where we used the garbage model as a filter and a gesture model as the classifier. In the isolated sample test, the gesture model was able to classify some samples that still came through the garbage filter as garbage. This resulted in a better performance because false positives were reduced without a large impact on the correct classifications. However in this streamwise test the gesture model was not able to show the same behavior. The confusion matrix for the accurate scenario shows no gestures that are classified correctly. This means that it is not possible to find the precise location of the writing gestures in an entire meeting with the approach we used.

Speech supporting gestures

Accurate scenario

Predicted / Actual	SSG	Garbage
SSG	9	166
Garbage	53	4632

Tolerant scenario

Predicted / Actual	SSG	Garbage
SSG	123	52
Garbage	189	4496

For the speech supporting gesture the garbage model is used as a filter in the accurate scenario. The gesture model is used to classify the data chunks that were not marked as garbage. This resulted for the accurate scenario in a better performance, instead of using the garbage model alone as we did in the isolated sample test. However the performance of the accurate scenario is very low, nine correct recognitions out of 175 with 53 false positives. For the tolerant scenario we used the speech supporting garbage model directly as the classifier just as in the isolated sample test.

Standing up

Accurate scenario

Predicted / Actual	Standing up	Garbage
Standing up	2	2
Garbage	1	10376

Tolerant scenario

Predicted / Actual	Standing up	Garbage
Standing up	3	1
Garbage	2	10375

The standing up garbage model is used directly as a classifier in the accurate and tolerant scenario, just as in the isolated sample test. Using a gesture model in combination with the garbage model didn't result in a better performance.

Comparison with isolated sample results

The last step is to evaluate the performance loss of the streamwise approach compared to the isolated sample approach. We have the confusion matrices of both tests. However, to compare two confusion matrices with each other a single measure is needed. This measure gives a performance score to both confusion matrices. There are several methods to measure performance based on a confusion matrix. We have analyzed the accuracy, Kappa, and F-measure.

All three measures can be used as a performance measure. Which one is more suited depends on the situation. When you compare an isolated sample confusion matrix with a streamwise confusion matrix you immediately see that a streamwise confusion matrix has far more true negatives than true positives. Because an entire meeting is processed the gestures are far rarer than the non-gesture occurrences. The accuracy and the Kappa measure both have the true negative component in their equations, but the F-measure has not. Therefore, we think that we can make the best comparison between the isolated sample and the streamwise performances with the F-measure.

The F-Measure leaves the true negatives out of the equation. This measure is defined as the harmonic average of the precision (P) and recall (R) measures. Precision is defined as the proportion of the predicted positive cases that are correct. Recall is defined as the proportion of the actual positive cases that are correctly identified. The three measures are determined using the formulae in Equation 5.6. The results of the F-Measure comparison are given in Table 5.8

$$FM = \frac{2 \cdot (P \cdot R)}{(P + R)} \qquad P = \frac{TP}{TP + FP} \qquad R = \frac{TP}{TP + FN} \tag{5.6}$$

Gesture	Isolated sample F-Measure %	Streamwise F-Measure %	Performance difference
Writing			
Accurate scenario	57%	0%	57%
Tolerant scenario	63%	38%	25%
SSG			
Accurate scenario	43%	8%	35%
Tolerant scenario	85%	51%	34%
Standing up			
Accurate scenario	100%	57%	43%
Tolerant scenario	100%	66%	34%

Table 5.8 – Comparison between isolated sample and streamwise performance

From Table 5.8 we can see that the performance loss with the streamwise approach is at least 25% and even 57% in the accurate writing scenario. For this scenario and also for the accurate scenario of speech supporting gestures the streamwise performance is very low, respectively zero and eight percent.

In the previous isolated sample test paragraph we mentioned two discrepancies that are responsible for the performance in the streamwise test. The first discrepancy is that the accurate scenario focuses more on a low false alarm rate and the tolerant scenario more on a high hit rate. The second discrepancy is that in the accurate scenario the match between annotation and classification must meet precision requirements whilst in the tolerant scenario overlap is considered sufficient

For writing the difference between the accurate and tolerant scenario in the isolated sample test is 6%. In the streamwise test this difference increases to 38%. Because the first discrepancy is also present in the isolated sample test, the increase of 32% is mainly due to the second discrepancy of having to match the annotated boundaries precisely in the accurate scenario. For speech supporting gestures the difference in the isolated test was already 42%. This difference only increases slightly to 43%, meaning that the additional precision requirement doesn’t have a large impact on the streamwise performance. For standing up the difference between the scenarios increases from 0% in the isolated case to 9% in the streamwise case.

The main observation we can draw from this is that the longer the gesture is, the more its performance suffers from the additional accuracy restrictions in the accurate scenario. In the evaluation paragraph we will give a possible explanation for this observation.

5.5. Conclusion

To conclude the classification topic, a short summary of the findings made in the three test stages is given, starting with the reductions made on the test space with the 5x2 cross validation test. This is followed by the classification performance on isolated samples and the impact on this performance when using the suggested streamwise classification approach.

5.5.1. Test space reduction

The objective of the 5x2 cross validation test was to reduce the selected test space as much as possible. This was done by evaluating if one option for a certain classification parameter performed significantly better than all the other options for that parameter. To make this comparison we used a combination of the F-test and t-test on the mean and variances of the 5x2 cross validation test results.

The most obvious observation that could be made in the 5x2 test was that the GBBs performed significantly worse for all gestures. In the evaluation paragraph we will explain why we think this happened. Another apparent observation could be made regarding velocity or position based features. As expected the velocity features outperformed the position features because of the writing, SSG and standing up gestures' translational variance.

The test results of the 5x2 cross validation test could not give a conclusive answer for the last two tested classification parameters: whether a Cartesian or a polar feature representation is better and the ideal number of HMM states.

5.5.2. Isolated sample classification performance

The first objective of the isolated sample test was to select the best model for each gesture class. The second objective was to evaluate the performance of this model by determining the confusion matrix on a set of manually pre-segmented gestures and non-gestures. In determining the performance we used two scenarios. The first was an accurate scenario where we wanted as little false positives as possible. The second was a tolerant scenario where the emphasis lay more on a high number of true positives at the cost of some false positives.

The first observation we made when selecting the best model for each gesture class was that it is difficult to train a good gesture model. The gesture model was unable to make a good distinction between the gesture examples and the non-gesture examples in the validation set. This led to the use of garbage filtering. We trained a garbage filter on the all the data of a number of meetings. This resulted in a model for the most common movement in a meeting. With these garbage models we were able to mark a reasonable amount of the non-gestures as garbage, whilst leaving most of the actual gestures unmarked. This resulted in the following confusion matrices of Table 5.9 for the three tested gesture classes.

Accurate scenario

Predicted / Actual	Writing	Garbage
Writing	10	14
Garbage	1	46

Tolerant scenario

Predicted / Actual	Writing	Garbage
Writing	20	4
Garbage	20	27

Accurate scenario

Predicted / Actual	SSG	Garbage
SSG	48	127
Garbage	1	163

Tolerant scenario

Predicted / Actual	SSG	Garbage
SSG	149	26
Garbage	26	138

Accurate scenario

Predicted / Actual	Standing up	Garbage
Standing up	4	0
Garbage	0	31

Tolerant scenario

Predicted / Actual	Standing up	Garbage
Standing up	4	0
Garbage	0	31

Table 5.9 – Confusion matrices of the isolated sample test

For writing the confusion matrix of the accurate scenario shows only one false positive. But only 10 out of the total of 24 gestures are recognized correctly. Increasing the number of true positives in the tolerant scenario leads to a large increase in false positives. With 20 correctly classified gestures there are also 20 false positives.

For the speech supporting gestures there is also only one false positive in the accurate scenario. But again only 48 of the 175 gestures are correctly classified. However, increasing the true positives in the tolerant scenario, doesn't lead to the same dramatic increase in false positives that we see for writing. The tolerant scenario performance results in 148 true positives against 26 false positives.

The confusion matrices for both scenarios of the standing up gesture show no false negatives or false positives, indicating a perfect separation of gestures from non-gestures. Furthermore we used only the garbage model for this gesture to get this result. This works because the standing up gesture is really different from the common movement in a meeting. This enables the garbage model to make a perfect distinction between the actual gestures and the garbage.

5.5.3. Impact of the streamwise classification approach

The objective of the streamwise test was to evaluate the performance impact of not having pre-segmented gestures. This impact is determined by comparing the isolated sample performance with the streamwise performance. We first determined the new confusion matrices for the three tested gestures, summarized in Table 5.10.

Accurate scenario

Predicted / Actual	Writing	Garbage
Writing	0	24
Garbage	0	6238

Tolerant scenario

Predicted / Actual	Writing	Garbage
Writing	10	14
Garbage	18	6220

Accurate scenario

Predicted / Actual	SSG	Garbage
SSG	9	166
Garbage	53	4632

Tolerant scenario

Predicted / Actual	SSG	Garbage
SSG	123	52
Garbage	189	4496

Accurate scenario

Predicted / Actual	Standing up	Garbage
Standing up	2	2
Garbage	1	10376

Tolerant scenario

Predicted / Actual	Standing up	Garbage
Standing up	3	1
Garbage	2	10375

Table 5.10 – Confusion matrices of the streamwise test

Next we used the F-measure to give a performance score to these confusion matrices and the isolated sample confusion matrices. Using this F-measure we compared the streamwise performance with the isolated sample performance. The conclusion of this comparison is that a streamwise classification approach results in at least 25% performance loss and even 57% for the accurate writing scenario. In the next paragraph we evaluate why we think the performance loss is this large, especially for the accurate scenario.

5.6. Evaluation

The idea of this evaluation paragraph is to give a possible explanation for the observed test results, based on our experiences acquired during testing. The claims made in this paragraph are speculations. They are not validated with test results, but they do point out what the problem areas are.

5.6.1. GBBs performance

As mentioned in the results of the 5x2 test paragraph, the GBB approach performs a lot worse than the gesture part approach. The idea behind using GBBs is that they would provide a more uniform description of the gestures within one gesture class. We will discuss here what we believe are the two main reasons why constructing GBBs out of the gesture parts has failed.

The first reason is the displacement of the boundaries within a gesture. The gestures are separated into parts in the segmentation phase of the recognition process. Because of the precise match of generated boundaries with the annotated boundaries we expected that the boundaries inserted within the gestures are also placed at meaningful positions. These are positions where the insertions split the gesture into semantically meaningful parts. It is however very well possible that too few or too many boundaries are inserted or that they aren't inserted at a logical position at all. Since we use an unsupervised method to construct the GBBs there is no control mechanism on whether the placement of these boundaries is correct. Suppose the boundaries within a gesture are not placed at a similar location for each gesture. This makes the division of the gesture into parts inconsistent. Constructing GBBs from these inconsistent parts will not result in a more uniform description for each gesture.

The second reason lies with the dependency on the clustering of the data. The main cluster method uses the clustered data to determine the label of the GBB. The direct clustering method calculates an average vector on the continuous data and then clusters this vector to determine the label. When the movement within a part cannot clearly be represented by one label, because the data is matched to two or more clusters, both methods choose only one of those clusters as a label. This could mean that very similar pieces of data will not always be matched to the same GBB label, but to a set of two or sometimes more labels. In the end this leads to a large set of inconsistent label sequences for the same gesture and not to the intended uniform description.

These two reasons together explain in our opinion why the GBB strategy doesn't perform well in the 5x2 tests. On one hand the search space isn't reduced effectively whilst on the other hand too much context information is lost.

5.6.2. Gesture modeling difficulties

Paragraph 5.4.2 explains that a garbage model is used to filter out the pieces of data that clearly contain no gesture. The results of the isolated sample tests show however, that for some gesture classes it is still very difficult to produce a good gesture model. The two gesture classes that show these problems are the speech supporting gesture and the writing gesture. We think that the main problem lies in the gesture classes themselves for these two gestures.

The writing gesture is a very long gesture that actually consists of three main parts. These parts are: the starting body movement forward towards the writing position, the actual writing part and the body movement backwards. The results that show the splitting of the feature sets in a set for the hand features and a set for the head features indicate that these movements together are hard to classify. Also the long and highly variable duration of this gesture has an impact on the performance. While an HMM is suitable to cope with variable observation lengths, the average length of 254 frames for the writing gesture is probably too much to fit into one HMM. We think that trying to recognize the writing gesture with the three sub movements together and with such a long duration as a whole is too much for one HMM.

For the speech supporting gesture we suspect that the diversity of the gesture is too high to fit into a single model. Because a model is constructed based on such a diverse set of data it is very likely that this model explains a wide variety of hand movements. This means essentially that other hand movements that are no speech supporting gesture will also be explained by the model. When the two confusion matrices of the isolated sample tests are compared, it shows that a small set of false positives can be removed but at the cost of many true positives. We think that these false positives, that are hard to remove, are other hand movements that are also easily explained by the gesture model.

Another indication for the fact that the speech supporting gesture model explains a too diverse set of data is given by the results of the 5x2 and isolated sample tests. The initial prediction of the number of states given in Table 5.3 was four to eight states. The 5x2 tests already showed clearly that eight or more states were needed. The ultimately selected model in the isolated sample test even has eighteen states. The high capacity of eighteen fully connected states allows the modeling of a large set of observations sequences, including some non-gesture observations.

5.6.3. Streamwise performance loss

In Paragraph 5.4.3 the performance loss of the streamwise approach versus the isolated sample approach was determined. Based on the F-measure the streamwise approach has at least a performance loss of 25%. In the accurate scenario for writing this loss is even 57%. In this evaluation we try explain this observation.

First of all there is a difference in difficulty between the isolated sample test and the streamwise test. In the isolated sample test, the validation set consists of pre-segmented gestures and garbage chunks. The only decision that has to be made using the output of the garbage model and gesture model is whether a certain data chunk is a gesture or garbage. In the streamwise test there are also a number of other difficulties. First the validation set does not consist of pre-segmented data chunks but it consists of an entire meeting. The streamwise method described in Paragraph 5.2.3 has to search for occurrences of a gesture using the selected models and a sliding and expanding window. The second difficulty is that for the accurate scenario the match between the annotated and classified gestures must also meet a certain precision requirement. Both the begin and end boundary must lie close to the annotated boundaries. This extra difficulty for the accurate scenario also has its impact on the performance. For all three gestures the accurate scenario performs worse and has a higher performance loss than the tolerant scenario.

In the analysis Paragraph 5.2.3 we mentioned the problem of under-fitting when trying to find a precise match between classified and annotated gestures. The higher the length variation within a gesture class is the bigger this problem is. To cope with this problem we suggested a form of length compensation. In terms of probabilities the HMM probability of a chunk is multiplied with its length. However, the under-fitting problem still remains even with this length compensation. This can be observed from the streamwise performance scores. The writing gesture with the highest variation in length has a performance of 0% on the accurate scenario.

In retrospect, the reason why the under-fitting problem remains is that the length compensation we used is flawed. This idea is derived from the Bayes' rule, given in Equation 5.7.

$$P(G \mid chunk) = \frac{P(chunk \mid G) * P(G)}{P(chunk)} \tag{5.7}$$

The term *P(chunk|G)* is the HMM output probability of gesture G for this chunk of data. As said in Paragraph 5.2.3 this probability can not be compared for chunks of different length. We tried to solve this problem by multiplying the HMM output probability with the length of the chunk. However, the right solution according to Formula 5.7 is to compensate a chunks HMM probability for the chance of that chunk *P(chunk)*. This gives the term *P(G|chunk)* which is the probability of gesture G given a certain chunk. This probability can be compared for chunks of different lengths because it is corrected for the *a priori* chance of the chunk. In our case we do not know the *a priori* distribution to determine *P(chunk).* So in order to use this length compensation approach a solution has to be found on how to estimate *P(chunk)* from the available data samples. This will be further discussed in the recommendation Paragraph 6.2.3.

Chapter 6 - Conclusions and recommendations

In the last chapter of this thesis the original research objective is evaluated. Based on this evaluation we give some recommendations for future work.

6.1. Conclusion

Gesture recognition is a relatively new field of research. The lack of standard fully-developed approaches made our research challenging and served as a personal motivation. The main research motivation was the discrepancy between recognizing gestures in a controlled environment versus a more natural environment such as meetings. Assumptions that hold in a controlled environment cannot always be made for real life applications of gesture recognition.

This discrepancy inspired our research objective. In this research project we wanted to identify the problem areas in gesture recognition and determine the recognition performance when we apply existing machine learning techniques, used in gesture recognition, to recognize a set of predefined gestures in the more natural meeting setting.

Having a set of isolated samples to be recognized is an assumption often made. The confusion matrices and F-measure scores for the isolated sample test show that you can indeed get a reasonable performance under this assumption. But if the assumption of isolated gestures is abandoned the problem gets much harder. Having to locate the gestures in a meeting with a certain precision is an additional challenge in the recognition approach. This difficulty is reflected in the performance scores of the streamwise test. Especially for accurate scenarios and even for the tolerant scenarios the performance drops significantly.

The second part of the research objective was to point out where the challenges lie in gesture recognition. We identified three challenges in our research. The first challenge is finding consistent features to describe the gesture with. The second challenge is segmentation, identifying when something interesting occurs in a meeting without classifying what. The third challenge is modeling a gesture class based on the chosen features. The next page gives a summary of the main test results. This is followed by a description of the three challenges and their problems.

Segmentation results

Gesture	Whole gesture performance	Gesture part performance
Writing	29,75%	98,73%
SSG	10,03%	98,22%
Nodding	29,66%	90,16%
Shaking	13,33%	96,67%
Standing up	40,00%	100,00%
Sitting down	25,00%	100,00%
Total (weighted)	18,39%	95,55%

Table 6.1 – Summarized test results

Classification test results

Accurate scenario

	Writing	Garbage
Writing	10	14
Garbage	1	46
	SSG	Garbage
SSG	48	127
Garbage	1	163
	Standing up	Garbage
Standing up	4	0
Garbage	0	31

Tolerant scenario

	Writing	Garbage
Writing	20	4
Garbage	20	27
	SSG	Garbage
SSG	149	26
Garbage	26	138
	Standing up	Garbage
Standing up	4	0
Garbage	0	31

Table 6.2 – Confusion matrices of the isolated sample test

Accurate scenario

	Writing	Garbage
Writing	0	24
Garbage	0	6238
	SSG	Garbage
SSG	9	166
Garbage	53	4632
	Standing up	Garbage
Standing up	2	2
Garbage	1	10376

Tolerant scenario

	Writing	Garbage
Writing	10	14
Garbage	18	6220
	SSG	Garbage
SSG	123	52
Garbage	189	4496
	Standing up	Garbage
Standing up	3	1
Garbage	2	10375

Table 6.3 – Confusion matrices of the streamwise test

Gesture	Isolated sample F-Measure %	Streamwise F-Measure %	Performance difference
Writing			
Accurate	57%	0%	57%
Tolerant	63%	38%	25%
SSG			
Accurate	43%	8%	35%
Tolerant	85%	51%	34%
Standing up			
Accurate	100%	57%	43%
Tolerant	100%	66%	34%

Table 6.4 – Comparison between isolated sample and streamwise performance

6.1.1. Consistent features

The challenge here is to find the ideal features. These are features, invariant to translation, rotation and scaling, which are at the same time highly informative. On the other hand it must also be possible to reliably extract these features from the video recording of a gesture in some way. This extraction process has to be consistent so that the same gestures are described in the same way.

Because of the natural environment we don't have the luxury of precise feature measurement by means of a data glove for example. The extracted features in this project are derived from a dynamic stick-figure representation of the actors in a meeting. The estimation of this stick-figure representation results in more noise and less consistent features than one would get with direct sensors. As a consequence, gestures with small amplitudes might not be distinguishable anymore from noise. We encountered this in our project with the nodding and shaking gestures. Also the usefulness of more invariant higher level features derived from these base features such as accelerations will be compromised due to derivate noise amplification. This became apparent in the inverse LDA analyses of the feature selection chapter. All the derived accelerations and the angular velocities showed too little consistency on this test and were discarded.

Good features that can consistently be extracted from the gestures form the basis for a good segmentation and classification performance. We think there is a challenge in improving feature extraction techniques. This will result in an increased quality of the original data and improve the performance of segmentation and classification.

6.1.2. Segmentation

The challenge for segmentation is to automatically identify the begin and end boundaries of the occurring gestures in a meeting. This simplifies the classification task because the location of the gesture will already have been determined in the segmentation phase. Data between two boundaries only has to be labeled with the correct gesture label.

We researched two segmentation approaches, BIC and AM. Both approaches performed insufficiently to reliably segment the gestures as a whole. The best setup resulted in an average of only 18% correctly segmented gestures. Even with a perfect classifier the maximum classification performance would be 18%. Therefore, we also researched a less ambitious approach that segments the gestures in parts. With this approach we can reliably match 96% of the annotated boundaries with an automatically generated boundary. However, the downside of gesture parts is that the generated boundaries do not always indicate a gesture start or end. This means that each sequence of gesture parts might contain a gesture and has to be processed in the classification phase. Instead of only labeling a sequence, the classifier now also has to determine which sequence of parts is mostly likely a gesture. The data can be processed more efficiently in parts but the location of the gestures is still not known. In this project the impact of this additional problem was significant, at least 25% performance loss on the researched gestures.

Segmentation is still a big challenge. An approach which can reliably perform whole gesture segmentation will simplify classification and improve the overall classification performance significantly.

6.1.3. Gesture modeling

The challenge in gesture modeling is to find a good descriptive model of the gesture class. This representation should model or explain all the different gesture samples of a certain gesture class. Additionally the model should be discriminative enough to distinguish gestures from non-gesture samples.

We used HMMs to model the gestures. Preliminary results in the isolated sample tests produced a lot of false positives. This showed us that the trained gesture models had difficulties with separating the gestures from the non-gestures. We solved part of this problem with the use of garbage models, which model the average meeting movements. These models were able to filter out part of the non-gestures beforehand. For the standing up gesture this filtering was even 100% because this gesture is very different from the average movement in a meeting.

However, the gesture model will still give a reasonable explanation for the non-gestures that slip through the garbage filter. For an accurate scenario the threshold on the gesture models output has to be very strict to leave out those remaining non-gestures or false positives. As a result many of the gesture samples are also removed. This can be seen when classification results of the tolerant and accurate scenario of the isolated sample test are compared. Removing 19 of the 20 false positives that occur for writing in the tolerant scenario also removes 10 of the 20 true positives. Removing nearly all false positives in the accurate scenario for speech supporting gestures leaves only 48 out of the 149 true positives. The challenge is to find a better gesture model, so the removal of the last false positives does not have such a large impact on the number of correct classifications.

All three challenges mentioned above are not separate challenges. They heavily influence each other. Without precise consistent features the segmentation and classification performance will also suffer. And without a good segmentation you will also need a more accurate gesture model to still be able to find the correct gesture locations. In the following recommendations paragraph we present some ideas for future work to possibly overcome the challenges we identified.

6.2. Recommendations

The previous paragraph mentions the challenges in gesture recognition. In this paragraph we give a few suggestions which could improve the classification of gestures. These recommendations are divided into three main parts, annotation and feature selection, segmentation and classification.

6.2.1. Annotation and feature selection

A first recommendation for this phase of the recognition process is to use a feature extraction approach that is more precise and robust. A better feature extraction that permits the nodding and shaking data to be separated from the noise in the data allows an attempt at recognizing these gestures. Also the performance in the other stages of the project would improve when the original features are more consistent and less noisy.

A possible solution for the gesture modeling problem is simplifying the gesture classes. The recommendation is to make the annotation more specific so that the gestures in one class become more alike. For the writing gesture it is possible to divide the movement into three parts: the forward body movement towards the writing position, the actual writing hand movement and the ending body movement backwards. Annotating these three parts separately allows for a more flexible recognition. As long as you recognize the begin and end parts, the writing part itself doesn't have to be recognized to correctly classify the entire gesture.

The SSG class is a very diverse class, where many different hand movements are grouped together. One option is separating the iconics and metaphorics from the beats. The speech itself and precise annotation guidelines will be needed, to decide if a hand gesture contains communicative information (iconic or metaphoric) or only emphasizes the speech (beat). Another option to split the SSG class is annotating if a gesture is performed with the left hand, the right hand or with both hands.

For the nodding and shaking gesture class it is possible to split a repeating gesture into multiple nods or shakes. However, it is possible that nods, which were originally part of a repeated nodding gesture, don't contain a clear begin or end phase.

6.2.2. Segmentation

For segmentation, the recommendation in general is to continue research on whole gesture segmentation. The first specific recommendation is to group together gesture parts using BIC and work toward a whole gesture representation. BIC is used to model the two parts separately and together. When the model for the parts together gives a better score than the two models for the separate parts, the boundary between the parts can be removed.

The second recommendation is to apply garbage filtering before segmentation. With the garbage model you can reduce the area, where segmentation should look for gesture boundaries. When the regions where a gesture could be present are identified, it is possible to allow only two boundaries within this area. AM or BIC can be used to place the begin and end boundary.

The last recommendation is to use a more intricate measure for the activity. The measures we have researched are only individual, summed or combined features. A good measure should balance the input of several features so that a consistent representation of the activity is obtained. Kahol et al. [32, 31] for example, calculate the force, kinetic energy and momentum for different body segments. These segments are connected in a hierarchical structure, where parent segments inherit the aggregate characteristics of each child segment and child segments inherit the motion of the parent segment. This structure can be used to derive more intricate head, hand and body measures to segment the different head, hand and body gestures with.

6.2.3. Classification

The first recommendation for the classification phase is based on the results of the 5x2 tests. These results show that the unsupervised construction of building blocks and the two layered classification approach doesn't work. A supervised construction of the building blocks could produce a low dimensional but highly informative representation of a gesture. We think that a two layered classification approach in general has a very high potential for gesture recognition. Because gesture recognition is a spatio-temporal problem, one layer can be used for the spatial problem and the other for the temporal problem.

The results of the isolated sample tests show that a model doesn't only explain the gesture but also some non-gesture data. To make a better distinction, we think that more emphasis is needed on the sequence within the data and less on the data itself. The recommendation is to determine how well the most probable state sequence Q within HMM λ explains the observation sequence O, the probability $P(Q|O, \lambda)$. Instead of evaluating how well the entire HMM explains the observation sequence, the probability $P(O|\lambda)$. By using the Viterbi algorithm this state sequence can be determined and observed. Other classification methods that focus more on the pattern or small set of patters of a gesture class are also recommended.

The streamwise tests show that it is difficult to precisely match the classified gestures with the annotated gestures, due to the problems with variable lengths. Paragraph 5.6.3 explains that the *a priori* distribution of a gesture is needed to compensate for this length problem. The actual *a priori* distribution is unknown. A recommendation for estimating this distribution is to use the length distribution within a gesture class.

The last recommendation is to use domain knowledge after different gestures are recognized in a certain data stream. When the results show two gestures occurring at the same time that cannot overlap, because of physical restraints to the human body, the one with the highest likelihood can be selected. Another example is the sequence of standing up and sitting down gestures. The restriction that these two should be alternated ensures that no two standing ups or sittings downs are recognized after each other. A last example of using domain knowledge is using the input of other modalities such a speech to influence the *a priori* probability of certain gesture classes. For example the probability of nodding and shaking gestures increases in a discussion. When someone is giving a presentation the writing chance increases for the listeners. For the speaker the chance of pointing gestures increases. In general, information of the gesture recognition domain or other domains could be used to enhance the recognition result.

References

[1] J. K. Aggarwal and Q. Cai, *Human motion analysis: a review*, Computer Vision Image Understanding, 73 (1999), pp. 428-440.

[2] T. Allison, A. Puce and G. McCarthy, *Social perception from visual cues: role of the STS region*, Trends in Cognitive Sciences, 4 (2000), pp. 267-278.

[3] E. Alpaydin, *Introduction to Machine Learning*, MIT Press, 2004.

[4] D. Altman, *Practical Statistics for medical research*, Chapman and Hall, London, 1991.

[5] AMI, *Augmented Multi-party Interaction,* available from: http://www.amiproject.org, 2005

[6] B. Bauer and K. F. Kraiss, *Towards an Automatic Sign Language Recognition System Using Subunits,* in GW '01: Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction, Springer-Verlag, 2002, pp. 64-75.

[7] S. Bengio, *Challenges in Multi Channel Sequence Processing*, 2005.

[8] H. Birk, T. B. Moeslund and C. B. Madsen, *Real-time recognition of hand alphabet gestures using principal component analysis,* in Proceedings of the 10th Scandinavian Conference on Image Analysis, 1997, pp. 261-268.

[9] L. W. Campbell, D. A. Becker, A. Azarbayejani, A. F. Bobick and A. Pentland, *Invariant features for 3-D gesture recognition,* in Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition (FG '96), IEEE Computer Society, 1996, pp. 157.

[10] A. Camurri, B. Mazzarino, M. Ricchetti, R. Timmers and G. Volpe, *Multimodal Analysis of Expressive Gesture in Music and Dance Performances,* in GW '03: Revised Papers from the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 2003, pp. 20-39.

[11] M. A. Carreira-Perpiñán, *A Review of Dimension Reduction Techniques*, University of Sheffield, Sheffield, 1997, pp. 1-69.

[12] C. Cédras and M. Shah, *Motion-Based Recognition: a Survey*, Image and Vision computing, 13 (1995), pp. 129-155.

[13] R. Chellappa, A. Roy-Chowdhury and S. Zhou, *Recognition of Humans and Their Activities Using Video*, 2005.

[14] A. Corradini, *Real-Time Gesture Recognition by Means of Hybrid Recognizers,* in GW '01: Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction, Springer-Verlag, 2002, pp. 34-46.

[15] M. P. Craven and K. M. Curtis, *GesRec3D: A Real-Time Coded Gesture-to-Speech System with Automatic Segmentation and Recognition Thresholding Using Dissimilarity Measures,* in GW '03: Revised Papers from the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 2003, pp. 231-238.

[16] J. E. Cutting and D. R. Proffitt, *The Minimum Principle and the Perception of Absolute, Common and Relative Motions*, Cognitive Psychology, 14 (1982), pp. 211-246.

[17] J. W. Davis and A. F. Bobick, *The Representation and Recognition of Action Using Temporal Templates,* in Proceedings Computer Vision and Pattern Recognition, 1997, pp. 928--934.

[18] S. De Backer, A. Naud and P. Scheunders, *Non-linear dimensionality reduction techniques for unsupervised feature extraction*, Pattern Recognition Letters, 19 (1998), pp. 711-720.

[19] C. Dell, *A primer for movement description: Using effort-shape and supplementary concepts*, Dance Notation Bureau, New York, 1970.

[20] T. G. Diettrich, *Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms*, Neural Computation, 10 (1998), pp. 1895-1923.

[21] G. Fang, W. Gao, X. Chen, C. Wang and J. Ma, *Signer-Independent Continuous Sign Language Recognition Based on SRN/HMM,* in GW '01: Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction, Springer-Verlag, 2002, pp. 76-85.

[22] I. K. Fodor, *A Survey of Dimension Reduction Techniques*, Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, 2002, pp. 1-18.

[23] S. Gaffney and P. Smyth, *Trajectory clustering with mixtures of regression models,* in KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM Press, San Diego, California, United States, 1999, pp. 63-72.

[24] D. Gavrila, *The visual analysis of human movement: a survey*, Computer Vision Image Understanding, 73 (1999), pp. 82-98.

[25] R. Gherbi and A. Braffort, *Interpretation of Pointing Gesture: The POG system,* in GW '99: Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 1999, pp. 153-157.

[26] A. J. Howell and H. Buxton, *Gesture Recognition for Visually Mediated Interaction,* in GW '99: Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 1999, pp. 141-151.

[27] A. J. Howell, K. Sage and H. Buxton, *Developing Task-Specific RBF Hand Gesture Recognition,* in GW '03: Revised Papers from the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 2003, pp. 269-276.

[28] IDIAP, *M4 public scripted recordings,* available from: http://mmm.idiap.ch/publicMeetings.html, 2002

[29] G. Johansson, *Visual perception of biological motion and a model for its analysis*, Perception and Psychophysics, 14 (1973), pp. 201-211.

[30] D. Jurafsky and J. Martin, *Speech and Language Processing*, Prentice Hall, New Jersey, 2002.

[31] K. Kahol, P. Tripathi and S. Panchanatan, *Automated Gesture Segmentation From Dance Sequences,* in Sixth IEEE International Conference on Automatic Face and Gesture Recognition, IEEE, 2004, pp. 883-888.

[32] K. Kahol, P. Tripathi and S. Panchanatan, *Gesture segmentation in complex motion sequences,* in International Conference on Image Processing, IEEE, 2003, pp. 105-108.

[33] A. Kendon, *How gestures can become like words*, in F. Potyados, ed., *Cross cultural perspectives in nonverbal communication*, New York, 1988, pp. 131-141.

[34] A. Kendon, *Some Relationships between Body Motion and Speech*, in B. Poppe and A. Wolfe, eds., *Studies in Dyadic Communication*, Pergamon Press, 1972, pp. 177-210.

[35] J. Martin, D. Hall and J. L. Crowley, *Statistical Gesture Recognition Through Modeling of Parameter trajectories,* in GW '99: Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 1999, pp. 129-140.

[36] G. McClaun, F. Althoff, M. Lang and G. Rigoll, *Robust Video-Based Recognition of Dynamic Head Gestures in Various Domains - Comparing a Rule-Based and a Stochastic Approach,* in GW '03: Revised Papers from the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 2003, pp. 180-197.

[37] I. McCowan, D. Gatica Perez, S. Bengio and G. Lathoud, *Automatic Analysis of Multimodal Group Actions in Meetings*, IDIAP Switzerland, 2003.

[38] D. McNeill, *Hand and Mind: What Gestures Reveal about Thought*, University of Chicago press, 1992.

[39] T. B. Moeslund and E. Granum, *A survey of computer vision-based human motion capture*, Computer Vision and Image Understanding, 81 (2001), pp. 231-268.

[40] D. Moore, *The IDIAP Smart Meeting Room*, IDIAP, Martigny, 2002, pp. 1-15.

[41] K. H. Munk, *Development of a Gesture Plug-in for Natural Dialog Interfaces,* in GW '01: Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction, Springer-Verlag, 2002, pp. 47-58.

[42] K. Murakami and H. Taguchi, *Gesture recognition using recurrent neural networks,* in CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM Press, New Orleans, Louisiana, United States, 1991, pp. 237-242.

[43] J. Nespoulous, P. Perron and A. R. Lecours, *The Biological Foundations of Gestures: Motor and Semiotic Aspects*, Lawrence Erlbaum Associates, Hillsdale, 1986.

[44] H. Noot and Z. Ruttkay, *Gesture in Style,* in GW '03: Revised Papers from the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 2003, pp. 324-337.

[45] F. E. Pollick, *The Features People Use to Recognize Human Movement Style,* in GW '03: Revised Papers from the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 2003, pp. 10-19.

[46] R. Poppe, D. Heylen, A. Nijholt and M. Poel, *Towards real-time body pose estimation for presenters in meeting environments,* in Proceedings of the WSCG'2005, Plzen,Czech Republic, 2005.

[47] L. Rabiner, *A Tutorial on Hidden Markov Models and Selected Application in Speech Recognition*, Murray Hill, 1989, pp. 30.

[48] S. Reiter, *Proposal for Annotation for Gestures and Individual Actions*, Technical University of Munchen, 2004.

[49] G. Rigoll, A. Kosmala and S. Eickeler, *High Performance Real-Time Gesture Recognition Using Hidden Markov Models,* in Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction, Springer-Verlag, 1998, pp. 69-80.

[50] G. Rizzolatti, L. Fogassi and V. Gallese, *Neurophysiological mechanisms underlying the understanding and imitation of action*, Nature Reviews Neuroscience, 2 (2001), pp. 661-670.

[51] N. Rossini, *The Analysis of Gesture: Establishing a Set of Parameters,* in GW '03: Revised Papers from the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 2003, pp. 124-131.

[52] K. Sage, A. J. Howell and H. Buxton, *Developing Context Sensitive HMM Gesture Recognition,* in GW '03: Revised Papers from the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 2003, pp. 277-287.

[53] G. S. Schmidt and D. H. House, *Model-Based Motion Filtering for Improving Arm Gesture Recognition Performance,* in GW '03: Revised Papers from the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 2003, pp. 210-230.

[54] D. Stephens, *Hemispheric language dominance and gesture hand preference*, Department of Behavioral Sciences, University of Chicago., 1983.

[55] D. L. W. Swets, J., *Using Discriminant Eigenfeatures for Image Retrieval*, IEEE Transaction Pattern Analysis and Machine Intelligence, 18 (1996), pp. 831-836.

[56] T. Tarpey, *Unpublished - No Title*, Wright State University, 2005, pp. 158.

[57] A. Tritschler and R. Gopinath, *Improved Speaker Segmentation and Segments Clustering using the Bayesian Information Criterion,* in EuroSpeech '99 - Sixth European Conference on Speech Communication and Technology, Budapest, Hungary, 1999, pp. 679-682.

[58] M. Turk, *Gesture Recognition*, in K. M. Stanney, ed., *Handbook of Virtual Environments: Design, Implementation, and Applications*, Lawrence Erlbaum Associates, 2002, pp. 223-238.

[59] C. Vogler and D. Metaxas, *Handshapes and Movements: Multiple-Channel American Sign Language Recognition,* in GW '03: Revised Papers from the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 2003, pp. 247-258.

[60] C. Wang, W. Gao and J. Ma, *A Real-Time Large Vocabulary Recognition System for Chinese Sign Language,* in GW '01: Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction, Springer-Verlag, 2002, pp. 86-95.

[61] J. Wang and S. Singh, *Video analysis of human dynamics - a survey*, Pann Research - University of Exeter, 1999.

[62] L. Wang, W. Hu and T. Tan, *Recent Developments in Human Motion analysis*, National Laboratory of Pattern Recognition China, 2003.

[63] T. S. Wang, H. Y. Shum, Y. Q. Xu and N. N. Zheng, *Unsupervised Analysis of Human Gestures,* in PCM '01: Proceedings of the Second IEEE Pacific Rim Conference on Multimedia, Springer-Verlag, 2001, pp. 174-181.

[64] A. D. Wilson, *Adaptive Models for the Recognition of Human Gesture*, Massachusetts Institute of Technology, 2000, pp. 140.

[65] H. Wu and A. Sutherland, *Dynamic gesture recognition using PCA with multiscale theory and HMM,* in T. Zhang, B. Bhanu and N. Shu, eds. Proceedings SPIE: Image Extraction, Segmentation, and Recognition, 2001, pp. 132-139.

[66] Y. Wu and T. S. Huang, *Vision-Based Gesture Recognition a Review,* in GW '99: Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 1999, pp. 104-115.

[67] L. Zhao, *Synthesis and Acquisition of Laban Movement Analysis Qualitative Parameters for Communicative Gestures*, University of Pennsylvania CIS, 2001.

[68] B. Zhou and J. Hansen, *Unsupervised Audio Stream Segmentation and Clustering via the Bayesian Information Criterion,* in ICSLP-2000: International Conference on Spoken Language Processing, Beijing, China, 2000, pp. 714-117.

[69] M. Zobl, R. Nieschulz, M. Geiger, M. Lang and G. Rigoll, *Gesture Components for Natural Interaction with in-car devices,* in GW '03: Revised Papers from the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction, Springer-Verlag, 2003, pp. 448-459.

[70] M. Zobl, F. Wallhoff and G. Rigoll, *Action Recognition in Meeting Scenarios using Global Motion Features,* in PETS-ICVS - Proceedings Fourth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, Graz, Austria, 2003, pp. 32-36.

Appendices

A: Gesture description

B: Annotation tool comparison

C: Available features

D: Sample sizes

Appendix A - Gesture description

This appendix describes the seven meeting gestures chosen in this project. The characteristics of each gesture are described according to their location, temporal, amplitude and parametric properties. This is followed by each gestures annotation guidelines. The tested segmentation features are described for all gestures except pointing. The tested classification features and the tested HMM options are given for the writing, SSG and standing up classes.

Pointing

The pointing gesture is a hand and arm gesture. The gesture starts when the hand is moved away from an arbitrary rest position. Next the hand moves to where the person wants to point to. This is almost always followed by a retraction to an arbitrary end position. The duration of this gesture is relatively fixed. The amplitude depends on whether the hand starts close to the pointing position or not. Additional parameters for this gesture are the pointing direction (north, east etc.) and if the gesture is performed repeatedly.

Annotation guideline

Gesture	Begin movement	End movement
Pointing	Moving the hands away from a rest position.	Ending with the hands in a rest position.

Writing

The writing gesture is in general a hand gesture, but also the head and body are involved. This gesture starts when the person moves towards the writing position. This means that the head moves downward, the body leans forward and the hands move towards the pen and paper. Most of the time, this gesture ends with the inverse of this movement. The duration of this gesture is arbitrary; it all depends on how long the person is actually writing. This can change from 50 to even 500 frames. Because this gesture involves a change to a writing position the amplitude is relatively large especially for the head. The hands do not always have large amplitudes because they can already be near the pen and paper before the gesture starts.

Annotation guideline

Gesture	Begin movement	End movement
Writing	Moving hand, head and body from a rest position towards the object on which will be written.	Moving hand, head and body backwards toward the rest position.

Tested segmentation features

The writing characteristics of the annotation guidelines above are most clearly present in the different head features. This is because the head almost always moves significantly downward when someone begins to write and upward when they are finished. We observed that the following head features show the characteristic begin and end movements of a writing gesture:

Feature set	Selected feature(s)
Cartesian coordinates	Head Y
Polar coordinates	Head Radius / Delta
Joint angles	Head X joint
Cartesian velocity	Head Y
Polar velocity	Head Radius / Delta
Speed/Direction	Head Speed

One head feature is not present, namely the direction feature. This feature is just too noisy due to the use of the nonlinear arctan operation in its calculation.

The expected hand and arm (shoulder and elbow) movements do appear in the data but not as clearly as the head movement. This is because the hands and arms do not always have to be moved to get them to their writing position. It could very well be the case that the hands and arms are already in their correct positions. The body movement which could be measured by the back joint is also not clearly present. The first possible explanation is that someone is already leaning forward on the table. The second explanation is that the back at the position of the joint moves less clearly than the head when a writing gesture starts or ends.

The head movement primarily takes place in the vertical Y direction. Polar coordinates combine the Cartesian coordinate system in the X and Y direction. Because the X component has no effect the polar coordinates will show the same variation as the Cartesian Y coordinate at the same time. So it is possible to sum these features and amplify their characteristics. The same can be done with the Cartesian and polar velocities. Speed and joint values are different from the other features in scale and behavior and cannot be grouped together.

Tested classification features

Writing is a hand gesture, but a part of the gesture also involves a change in body position, moving the head downward. Therefore the hand features as well as the head features are part of the tested feature sets.

Gesture	Cartesian Position set	Polar Position set	Cartesian Velocity set	Polar Velocity set
Writing	Left hand X,Y Right hand X,Y Head X,Y	Left hand R,D Right hand R,D Head R,D	Left hand X,Y Right hand X,Y Head X,Y	Left hand R,D Right hand R,D Head R,D

Tested HMM options

1. Because of the length of this gesture, it consists of a relatively large amount of gesture parts. It should be possible to create enough GBBs out of these gesture parts. The problem that you discard information by using GBBs should therefore not be much of a problem. If the classification with GBBs doesn't perform well, discrete gesture parts are a second option.
2. By constructing the GBBs based on the direct clustering of the average vector of a gesture part we expect that the repetitive part occurring in the hand movement is smoothed out. The risk of this smoothing is however that too much information is thrown away. Therefore the second option, constructing the GBBs based on main cluster, is also considered.
3. A discrete HMM is necessary, since we intend to use GBBs or discrete gesture parts.
4. The number of states for this HMM lies probably somewhere between three and six. Less is unlikely because there are three distinct stages in this gesture: the begin head movement, writing hand movement and end head movement. More than six is also unlikely because all three gesture phases described above are not very complex and should not require more than two states.
5. If the repetitive writing part is pre-classified to one or two GBBs, a left right HMM topology could be suitable for the top level classification. When too many labels are used for the writing part and they show a repetitive pattern a left-right HMM might not be sufficient. In this case you need a fully connected HMM.

SSG

The speech supporting gestures are purely hand gestures. This gesture starts when one or both hands are moved away from their rest positions. This is followed by a short stroke where the speech is supported followed by a retraction to another rest position. The gestures in this group can be divided in beats which only emphasize speech and iconics or metaphorics which also illustrate the speech. The duration of this gesture is relatively fixed, but the beats tend to be shorter than the iconic and metaphoric gestures.

Annotation guideline

Gesture	Begin movement	End movement
SSG	Moving the hands away from a rest position.	Ending with the hands in a rest position.

Tested segmentation features

Speech supporting gestures are not easily distinguishable from other occurring movement because the average amplitude of this gesture is very small. Because these gestures are made solely with the hands we have examined the available hand features. The features that do show some characteristic speech supporting movements are:

Feature set	Selected feature(s)
Cartesian coordinates	Left/Right Hand X Left/Right Hand Y
Polar coordinates	Left/Right Hand Delta
Polar velocity	Left/Right Hand Delta

The other hand features either show no clear characteristic movement at all, or show inconsistent characteristics when you look at different examples of the same gesture.

It is possible to sum the coordinate or velocity data of the left and right hand together to get the movement of both hands in one feature. This might however cause problems when for example both hands make the same movement but in opposite direction. Adding the features of both hands in this situation might cancel out the occurring movement in the resulting feature. It is possible to work around this problem by taking the absolute velocity for example. It has to be tested if this summation has any effect on the segmentation performance.

Tested classification features

Speech supporting gestures are purely made with the hands so we consider only the hand features in the tested feature sets.

Gesture	Cartesian Position set	Polar Position set	Cartesian Velocity set	Polar Velocity set
SSG	Left hand X,Y Right hand X,Y	Left hand R,D Right hand R,D	Left hand X,Y Right hand X,Y	Left hand R,D Right hand R,D

Tested HMM options

1. Because this gesture is relatively short it only has a few gesture parts, roughly two or three. Classifying these parts to GBBs would probably result in a high loss of information. The SSG is a short, fast and expressive gesture. Our expectation is that this expressiveness separates this gesture from other hand movements. Clustering the data with too few clusters could result in abstracting from this expressiveness. It is however possible that with enough clusters the search space is reduced quite effectively while most information remains within the discrete feature data. We want to try the discrete gesture part approach because a discrete HMM is easier to train. If clustering proves to be difficult for this gesture we can fall back on continuous gesture parts.
2. Not applicable here since GBBs are not used.
3. Considering question one, we need either a discrete or continuous HMM.
4. Taking the complexity of the gesture into account we expect that an HMM needs four to eight states.
5. Again given the complexity we expect to need a fully connected topology.

Nodding and shaking

Nodding and shaking are head gestures. The nodding gesture starts when the head begins to move up- or downward. For the shaking gesture this movement is in a sideward direction. Both gestures end when this oscillating movement ceases. This can take an arbitrary amount of time depending on how long the person keeps nodding. This can be anywhere within 10 to 100 frames. The amplitude of this movement is small compared to the other gestures. The largest amplitude is found in the beginning of these gestures, it fades out toward the end. An additional parameter is, if it involves a single or repeated nods or shakes.

Annotation guideline

Gesture	Begin movement	End movement
Nodding	Beginning of the up or downward head movement.	End of the last up or downward head movement.
Shaking	Beginning of the sideward head movement.	End of the last sideward head movement.

Tested segmentation features

Nodding gestures have the same problem as speech supporting gestures, in that they are not easily distinguishable from other occurring movements. The features to take into consideration are of course those features that describe the movement of the head. Although not all head features show the expected nodding characteristics. The speed feature contains too little information to segment on and the direction feature is as mentioned before too noisy. The features which do somewhat show the typical up and downward movements of the head are:

Feature set	Selected feature(s)
Cartesian coordinates	Head Y
Polar coordinates	Head Radius/Delta
Joint angles	Head X
Cartesian velocity	Head Y
Polar velocity	Head Radius/Delta

The same conclusions can be made for the shaking movement. In general this gesture is the same as the nodding gesture only moving the head sideways instead of up and down. Therefore the same features can be used for this gesture class. Only the joint feature cannot be used since this feature is only measured in the up and down direction and not in the sideways direction. This leaves the following features:

Feature set	Selected feature(s)
Cartesian coordinates	Head X
Polar coordinates	Head Radius/Delta
Cartesian velocity	Head X
Polar velocity	Head Radius/Delta

Note that the characteristics for nodding and shaking are only very faintly present in these features and show only for the clear gestures. The reason for this is that the amplitude of the up, down and especially the sideward movement of the head is very small during these gestures, in the order of one or two pixels. It is possible to amplify the characteristic movements just as in writing by summing the Cartesian and polar coordinates and Cartesian and polar velocities together.

Standing up and Sitting down

Standing up and sitting down are whole body gestures. When preparing to stand up a person moves his arms backward and body forward. The gesture ends when the person is fully upright. The sitting down gesture starts when a person moves down and ends when he or she is seated again. The duration depends a bit on how much a person hesitates during these gestures but in general the duration is relatively fixed. The amplitude caused by these gestures is large, because it involves an entire change of place from a seated to a standing position.

Annotation guideline

Gesture	Begin movement	End movement
Standing up	Moving arms backward and body forward.	Ending in a (straight) standing rest position.
Sitting down	Begin movement downwards.	Ending with the body in a seated rest position.

Tested segmentation features

The observation of moving the arms backward is best seen in the shoulder X joint feature of both arms (flexion extension in the longitudinal plane). This data shows the same data variation for both arms, noting that the data of the left arm is inversed compared to the data of the right arm. This inversion occurs when one arm is places on the arm rest of a chair and the other on the table. Other features that describe arm or hand movement do not show the same characteristics for both hands or differ too much between different samples. This is due to the fact that the arms and hands start in an arbitrary start location. For example one person might keep his hands on the table when standing up to support his weight, while another person might not.

The downward movement of the body at the beginning of standing up and the rise of the body at the end of the gesture are most clearly seen in the features that describe the position of the head. The head typically moves down first, when a person prepares to stand up and then moves up very fast during the actual standing up movement. The suggested features to use are:

Feature set	Selected feature(s)
Cartesian coordinates	Head X/Y
Polar coordinates	Head Radius/Delta
Joint angles	Left/Right Shoulder X
Cartesian velocity	Head X/Y
Polar velocity	Head Radius/Delta
Speed / Direction	Head Speed

The sitting down movement is very similar to the standing up movement. It practically is the inverse movement of standing up. Therefore we assume that the features found useful for standing up are also the features to use for sitting down.

Tested classification features

Standing up will cause variations in almost all the measured features because of its large amplitude. However these variations are not always consistent for different examples of this gesture. The only consistent movement is the large upward and downward movement of the head and the change in the root Y position. These features are considered in the tested feature sets.

Gesture	Cartesian Position set	Polar Position set	Cartesian Velocity set	Polar Velocity set
Standing up	Head X,Y Root Y	Head R,D Root Y	Head X,Y	Head R,D

Tested HMM options

1. The standing up gestures are relatively long gestures, so the construction of GBBs should be possible. If it turns out that the usage of GBBs restricts the data too much, the clustered gesture parts are also an option. We expect that the upward movement should still be present in the clustered data. These two options, GBBs and discrete gesture parts, are chosen to be tested.
2. When the GBBs are used direct clustering of the average of the gesture parts should leave most information of the upward movement intact. Therefore we chose this method for constructing the GBBs.
3. For both the GBB strategy and the discrete gesture part approach we need a discrete HMM.
4. We suspect that the number of states needed to model the movement depends for on the number of clusters or GBBs. The HMM should model the upward aspect, based on a number of possible sequences of GBBs or clusters. Because the number of clusters cannot be too small the number of states should at least be five.
5. Because this gesture has no repeated character a left-right topology should suffice.

Appendix B – Annotation tool comparison

Within the AMI project Reiter [48] suggests two tools suitable for annotation of the video data. Another tool which has been used in the AMI project is the Nite toolkit. Of course more tools are available, but the given three should be good enough for our annotation purposes. By means of a short list of pros and cons we will determine which tool is going to be used for annotation. The aspects on which we evaluated the tools are:

1. Is it possible to annotate multiple gestures in the case that more gestures are performed simultaneously?
2. Is it possible to add attributes to a gestures annotation? (e.g. to indicate whether the performed gesture is clear or vague)
3. Is it possible to save the annotation data in an easy to use format such as XML?
4. Are basic video playback functions available? (e.g. stop, start and pause)
5. Is the program intuitive and easy to use?

The table below lists the results of the short review.

Tool / Criteria	Anvil	TASX	Nite
1	+	+	+
2	+	+	-
3	+	+	+
4	+	+	+
5	+	-	+

Table B.1 – Evaluation results of the three annotation tools

When you just look at the number of + marks, the Anvil tool scores the highest. In our opinion this is the best tool to use for our annotation purposes since it supports all the required features and also some more that aren't listed.

The disadvantage of the TASX tool is primarily its usability. It too supports all the necessary features but it is cumbersome to work with. This has primarily to do with the lack of a good coupling between the annotation window and the video display. This makes it more difficult and time consuming to annotate precise gesture boundaries. Another smaller disadvantage is the rather unintuitive output format of the TASX tool which makes a combination with other gesture recognition tools more difficult.

A disadvantage of the Nite tool is that it is a labeling tool and doesn't support the addition of attributes to the label. It is easy to use because you can annotate "on the fly" using key-shortcuts while the video plays. In the other tools you have to pause, select the correct fragment and label it. This on the fly annotating however has a disadvantage because a human has a certain reaction time. Because of this the begin boundary will start too late, when the gesture has already begun and the gesture will also end too late. These deviations will also differ between annotators, because everyone has a different reaction time. Even the reaction time of one annotator will change during annotation because he or she will get used to the program. This requires another, difficult correction pass to correct all the boundaries.

Appendix C – Available features

#	Name	#	Name
	Head & hand position		Cartesian accelerations
1	Head X	42	Cartesian Acceleration Head X
2	Head Y	43	Cartesian Acceleration Head Y
3	Left Hand X	44	Cartesian Acceleration Left Hand X
4	Left Hand Y	45	Cartesian Acceleration Left Hand Y
5	Right Hand X	46	Cartesian Acceleration Right Hand X
6	Right Hand Y	47	Cartesian Acceleration Right Hand Y
	Root position		Polar acceleration
7	Root X	48	Polar Acceleration Head R
8	Root Y	49	Polar Acceleration Head Delta
9	Root Z	50	Polar Acceleration Left Hand R
		51	Polar Acceleration Left Hand Delta
	Joint angles	52	Polar Acceleration Right Hand R
10	Head joint X	53	Polar Acceleration Right Hand Delta
11	Head joint Y		
12	Back joint		Speed and direction
13	Left shoulder X	54	Speed Head
14	Left shoulder Y	55	Direction Head
15	Left shoulder Z	56	Speed Left Hand
16	Left elbow	57	Direction Left Hand
17	Right shoulder X	58	Speed Right Hand
18	Right shoulder Y	59	Direction Right Hand
19	Right shoulder Z		
20	Right elbow		Angular velocity
		60	Angular velocity Head joint X
	Polar coordinates	61	Angular velocity Head joint Y
21	polar Head R	62	Angular velocity Back joint
22	polar Head Delta	63	Angular velocity Left shoulder X
23	polar Left Hand R	64	Angular velocity Left shoulder Y
24	polar Left Hand Delta	65	Angular velocity Left shoulder Z
25	polar Right Hand R	66	Angular velocity Left elbow
26	polar Right Hand Delta	67	Angular velocity Right shoulder X
		68	Angular velocity Right shoulder Y
	Head/hand distances	69	Angular velocity Right shoulder Z
27	distance Head	70	Angular velocity Right elbow
28	distance Head		
29	distance Left Hand		Angular acceleration
		71	Angular acceleration Head joint X
	Cartesian velocities	72	Angular acceleration Head joint Y
30	Cartesian Velocity Head X	73	Angular acceleration Back joint
31	Cartesian Velocity Head Y	74	Angular acceleration Left shoulder X
32	Cartesian Velocity Left Hand X	75	Angular acceleration Left shoulder Y
33	Cartesian Velocity Left Hand Y	76	Angular acceleration Left shoulder Z
34	Cartesian Velocity Right Hand X	77	Angular acceleration Left elbow
35	Cartesian Velocity Right Hand Y	78	Angular acceleration Right shoulder X
		79	Angular acceleration Right shoulder Y
	Polar velocities	80	Angular acceleration Right shoulder Z
36	Polar Velocity Head R	81	Angular acceleration Right elbow
37	Polar Velocity Head Delta		
38	Polar Velocity Left Hand R		
39	Polar Velocity Left Hand Delta	82	Duration
40	Polar Velocity Right Hand R	83	Intensity
41	Polar Velocity Right Hand Delta	84	Locus

Table C.1. – The complete set of available features.

Redundant features

#rows	unsmoothed	smoothed
1	1, 8, 9, 11, 30, 32-36, 38-53, 55, 59-81	1, 8, 9, 11, 25, 30, 32-35, 39-40, 42-44, 46, 48-49, 51-52, 55, 57, 60-67, 69-74, 76-81
2	8, 9, 11, 30, 32-36, 38-53, 60-81	9, 11, 30, 32-33, 35, 38-40, 42-44, 46, 48-49, 52, 55, 57, 60-63, 65-67, 69-74, 76-81
3	9, 11, 30, 32-36, 38-40, 42-53, 60-81	9, 11, 30, 33, 35, 39-40, 42-44, 46, 48-49, 52, 55, 57, 60-63, 65-67, 69-74, 76-81
4	9, 11, 32-35, 38-40, 42-53, 60-81	9, 11, 39-40, 42-44, 46, 48-49, 52, 55, 57, 60-63, 65-67, 69-74, 76-81
5	9, 11, 32-35, 38-40, 42-53, 60-81	9, 11, 39-40, 42-44, 46, 48-49, 52, 55, 57, 60-63, 65-67, 69-74, 76-81
6	9, 11, 32-35, 38-40, 42-53, 60-81	9, 11, 39-40, 42-44, 46, 48-49, 52, 60-63, 65-67, 69-74, 76-81

Table C.2. – The numbers of the features that are below the threshold of the feature reduction test. The feature numbers are given for the reduced dimensions one to six and the smoothed and unsmoothed situation.

Appendix D – Sample sizes

Gesture	Train (50%)	Validation (20%)	Test (30%)	Total
Writing	39	16	24	79
SSG	268	175	175	618
Standing up	4	2	4	10

Table D.1 – The sample sizes for the train, validation and test set for the tested gesture classes.

Glossary

AM	-	Activity Measure
AMI	-	Augmented Multiparty Interaction
BIC	-	Bayesian Information Criterion
DOF	-	Degree Of Freedom
EM	-	Expectation Maximization
GBB	-	Gesture Building Block
GP	-	Gesture Part
HMM	-	Hidden Markov Model
LDA	-	Linear Discriminant Analyses
MDF	-	Most Discriminating Features
MEF	-	Most Expressive Features
SOFM	-	Self Organizing Feature Maps
SSG	-	Speech supporting gesture