Saarland University
Faculty of Natural Sciences and Technology I
Department of Computer Science

Master's Thesis

Hand Shape Recognition Using a ToF Camera: A Sign Language Application

submitted by
Martin Šimonovský
(mys007@seznam.cz)

submitted
March 29, 2011

Supervisor / Advisor
Prof. Dr. Christian Theobalt

Reviewers

Prof. Dr. Christian Theobalt
Priv.-Doz. Dr. Meinard Müller

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Statement under Oath

I confirm under oath that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis.

Einverständniserklärung

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.

Declaration of Consent

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Saarbrücken,

(Datum / Date) (Unterschrift / Signature)

Abstract

This master's thesis investigates the benefit of utilizing depth information acquired by a time-of-flight (ToF) camera for hand shape recognition in unrestricted viewpoints. Specifically, we assess the hypothesis that classical 3D content descriptors might be unnecessarily powerful for ToF data and extended 2D descriptors could perform well enough. Our system is based on the appearance-based retrieval paradigm, using a synthetic 3D hand model to generate its database. The system is able to run interactively. For increased robustness, no color, intensity, or time coherence information is used. A novel, domain-specific algorithm for segmenting the forearm from the upper body based on reprojecting the acquired geometry into the lateral view is introduced. Moreover, three kinds of descriptors exploiting depth data are proposed and the made design choices are experimentally supported. The whole system is then evaluated on an American sign language fingerspelling dataset. However, the improvement of using depth information is not consistent in our system and the retrieval accuracy is not ideal. Possible reasons are discussed.

Acknowledgements

I would like to thank Prof. Dr. Christian Theobalt for introducing me to the area of time-of-flight cameras, his guidance, and keeping the project on track.

I would also like to express my gratitude to Andreas Baak for his support, helpful suggestions, staying updated on the project, and for reviewing the text.

Also, I would like to thank Dr. Alexis Heloir for providing me with a 3D hand model.

Last but not least, I am indebted to Prof. Joachim Weickert and Dr. Andrés Bruhn for their enthusiasm, great lectures, and for provoking my interest in computer vision.

A special thanks goes to the great UdS campus and its people for hosting me here for a few years.

Contents

Chapter 1

Introduction

Hands play a crucial role in interaction with objects and people. Hand gestures are fast, intuitive, and often subconscious means of communication: we use hands to express and support our thoughts, feelings, or ideas. Sometimes, they are the only way of utterance, such as in noisy or noiseless environments, over long distances or, importantly, among deaf or mute people. Here, complex sign languages (SL) have evolved as complete substitutes for spoken languages. Computer-based recognition of gestures has thus received wide interest over the years.

Two main applications of automated gesture recognition exist. First, human-computer interaction (HCI) can allow equipment to be controlled by gestures without any need for physical contact. The second one is sign language recognition, which is important in many ways. Some deaf people, particularly those completely deaf from birth, are also not able to speak and usually have troubles in reading and writing a spoken language. An automated translation system would make it possible to directly communicate with deaf people both for untrained humans and for computers. However, the problem is still far from being solved due to the very high complexity of it.

This thesis concentrates on a small but very important part of the field, namely the recognition of hand shapes. A hand shape refers to a specific configuration of the fingers. In the context of sign language recognition, the hand shape is regarded as the most informative element after the location and motion [ten Holt et al. 2009a] and it was shown to improve the recognition performance when incorporated even in a very crude form [ten Holt et al. 2009b]. Although using simple shape features along with, e.g., motion is common in SL recognition, explicit detection of hand shapes is rare. The exception is the recognition of fingerspelling, a distinct part of a number of sign languages around the world. Here, the fingers are used to spell out more obscure words and proper nouns, letter by letter. In this case, the hand shape is typically the primary determinant of the letter.

The human hand is an articulated object with at least 20 internal degrees of freedom. Considering the additional 6 rotation and translation parameters and different hand proportions among the population, vision-based hand shape recognition is a very difficult problem in its generality [Erol et al. 2007]. However, different levels of accuracy are needed by different applications. In the case of sign languages, there is only a limited number of

semantically different hand shapes. In particular, there are 41 distinct hand shapes in the American sign language (ASL) [Tennant and Brown 2002]. In the domain of fingerspelling recognition, the restrictions are typically further tightened to a fixed viewpoint and about 25 hand shapes at maximum. For this thesis, we formulate its goal as the recognition of 14 ASL hand shapes[1] of an adult male's right hand in arbitrary viewpoints.

In real world applications it is important that the recognition works in a rather unconstrained environment. The majority of existing approaches use single or multiple RGB cameras for data acquisition. This brings about the problem of locating and segmenting the hand(s). Hence, diverse assumptions are usually made. These include using colored gloves, long sleeves, uniform background, slow motion, or fixed body poses, to name a few [Ong and Ranganath 2005].

To overcome some of these restrictions, we investigate the benefit of utilizing a time-of-flight (ToF) camera for this task. ToF cameras acquire depth information using an active illumination system. The positives lie in a reduced dependence on scene illumination (especially shadows), practically no restrictions on clothing, and no need to wear special gloves or have other additional objects attached. However, the current ToF cameras' main disadvantages are the low resolution, the high level of noise and other artifacts in the measured signal.

We use depth information exclusively and do not exploit any color data. Besides aiming to explore how far one can go with depth images only, not all ToF cameras are equipped with an RGB sensor of an acceptable quality. To allow for fast hand motions, we do a single frame estimation instead of assuming time coherence.

The underlying hypothesis of this thesis is that classical 3D content descriptors might be unnecessarily powerful for ToF data. Thus, we extend 2D descriptors specifically designed to operate on ToF depth images. The principal aim of this work is to test this hypothesis.

The main contributions of this work are threefold:

- We propose a novel, domain-specific algorithm for segmenting the forearm from the upper body based on reprojecting the acquired geometry into the lateral view where separating the two body parts is done by finding and cutting the gap between them.
- We design a new histogram-based descriptor for capturing the silhouette as well as the depth information of the segmented hand region. The made design choices are backed by an extensive evaluation.
- We extend two existing 2D contour matching algorithms to utilize 3D information and we explore the possibility of combining these with the histogram-based approach.

This thesis is organized as follows. In the rest of this chapter, related work on gesture recognition with ToF cameras as well as the overview of our approach is given. Chapter 2 gives background information on sign language, hand shape recognition and time-of-flight cameras. Additionally, details on our data acquisition process are given. Chapter 3 describes and evaluates several segmentation algorithms. In Chapter 4, histogram-based and

[1] The number 14 is a result of the time constraints given to finish the thesis and is not of a semantic nature or technological limitation.

contour-based descriptors are designed. Our synthetic hand shape database is introduced in Chapter 5. Here, we evaluate different design decisions from the previous chapter and perform an overall evaluation on test datasets together with a discussion about the results. Finally, Chapter 6 gives a conclusion of this work and perspectives for further research.

1.1 Related Work

Although the literature on hand gesture recognition is vast, we take into account solely the hand shape recognition from depth information here.

One of the first who worked with depth data for hand gesture recognition were Malassiotis et al. [Malassiotis and Strintzis 2008], following a structured light approach. They described the whole processing chain from the acquisition to classification and are able to recognize 20 hand shapes while allowing for a considerable degree of out-of-plain rotations. A hierarchical unsupervised clustering procedure was used to segment the forearm. Next, two Gaussians were iteratively fit to the forearm to separate the hand. We experiment with this approach in Section 3.2. Then, the depth image of the hand is rectified using the hand's eigenvectors. Finally, the classification is performed by k-nearest neighbor search in a space with the dimensionality reduced by principal component analysis (PCA). The authors explored generating this space with real training data and with synthetic data, the latter performing worse. A performance of 15 fps is achieved.

Several authors, e.g. [Kollorz et al. 2008, Soutschek et al. 2008], recognized a few static hand gestures using ToF cameras in a similar frame rate. However, their features were rather low-level and the viewpoint was fixed.

Recently, a model-based approach was tested on ToF data [Guomundsson et al. 2010] for handling out-of-plane rotations. The authors used an ellipsoids-based hand model and a particle filter tracker on a tracking space with the dimensionality reduced by PCA to the size of 4. The basis vectors were learned from 3 hand shapes and the transitions between them. The tracking is initialized manually and runs at about 0.5 fps.

A related area is that of full-body tracking using ToF cameras, which has received more interest than hand tracking in the research community up to now. For the current state of the art, see the recent works of [Ganapathi et al. 2010, Knoop et al. 2009] or the trackers used in commercial products like Microsoft's Kinect.

1.2 An Overview of Our Approach

We base our system on the paradigm of appearance-based recognition. This data-driven approach requires building a database of images of all hand shapes (classes) observed from different viewpoints in an offline phase. Our database is built by rendering a synthetic 3D hand model. In online recognition, the hand shape can be retrieved by searching for the best match between the input image and the database of stored images. The images are matched by means of descriptors extracted from them. The pipeline is visualized in detail in Figure 1.1.

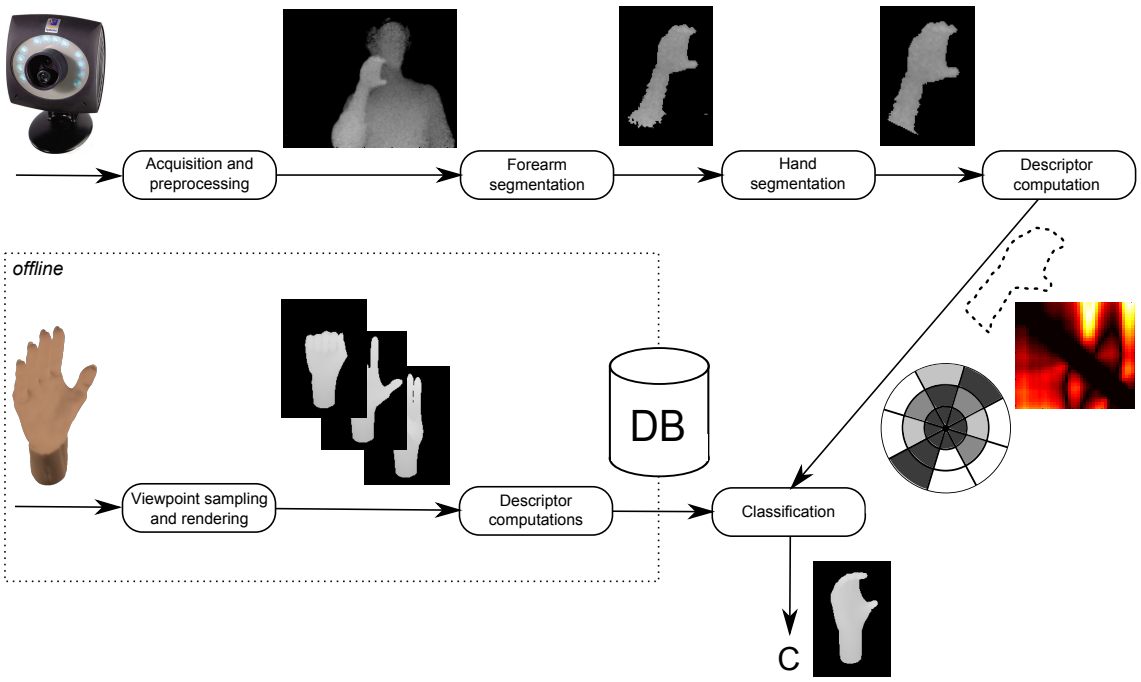


Figure 1.1 – An overview of our hand shape recognition system. Note that all three descriptors proposed in this thesis are visualized here but only one of them is computed at a moment.

First, we segment the input query depth image to extract the hand with just a small part of the forearm. Afterwards, we use the hand image to compute a single descriptor of the hand. We introduce two types of descriptors. The histogram-based approach summarizes the content of the whole hand region, whereas the contour-based approach works only on the extracted contour of the hand.

The final step is matching of the descriptor with every database image to find the nearest neighbor. The class of the database image is used to as the output of the system.

Chapter 2

Background

In this chapter, we introduce several topics related to this thesis. In Section 2.1, we present a very light introduction to (American) sign language. Section 2.2 reviews two approaches commonly used for hand shape recognition. Finally, Section 2.3 gives background on time-of-flight cameras. Additionally, details on our data acquisition process are given together with the necessary notation.

2.1 American Sign Language

A sign language is a communication system using gestural signs as the modality. Sign languages have developed in deaf communities in a natural way and independently from spoken languages. Hence, contrary to popular belief, there is no universal sign language. Instead, a different type of sign language has evolved in every deaf community on its own. Similarly to spoken languages, there are many national sign languages and their dialects. Yet, their relations are different than the ones of spoken languages. For example, American sign language (ASL) is much more similar to French signing language than to the British one. It should be emphasized that sign language is not a pantomime. Although there are some illustrative, iconic signs in sign languages as well, most of the gestures are abstract and not self-explanatory. Sign languages are not limited in its expressiveness to concrete subjects.

The research on sign languages is rather young. The first systematic linguistic studies of ASL were performed in the 1960's mainly by William Stokoe. Since then, it was shown that sign languages are actually real languages with a complete linguistic structure, i.e. phonology, morphology, semantics, and syntax. Afterwards, ASL gradually started to be taught in American schools for deaf people, replacing the until then forced education of lip reading (oralism). Later, similar development followed in other countries as well.

The smallest contrastive units in languages are called phonemes, that is, the smallest units that can distinguish morphemes (units of meaning) from another. In spoken languages, phonemes bear unique features identifying how and where a sound is created, e.g., whether the sound is consonantal or not. In contrast to the acoustic features, sign lan-

guage phonemes are defined by a set of visual features. We will review each of them in the following list:

Location: The signing space is the area in front of the signer in which the signs are performed, including the face and the upper part of the body. The area can be partitioned into lexically meaningful areas. Locations in front of the face are common since fluent signers usually look in the eyes of each other when communicating. The size of a sign can change depending on its 'loudness'. In ASL, 12 locations are distinguished.

Hand shape: The hand shape is defined by the positions of the fingers. Theoretically, it is possible to form a lot of hand shapes, but only a small subset of them are used in sign languages. Each language defines their own semantic hand shapes, although common ones exist. In ASL, 41 hand shapes are distinguished.

Motion: The motion refers to the trajectory of one or both hands in space. Some signs are performed with only one hand, the dominant hand. In signs performed with two hands the non-dominant hand performs a similar movement as the dominant hand or has a supporting role.

Orientation: The degree of orientation of the palm towards the signer.

Non-manual features: Non-manual components include facial expression, mouthing, eye gaze, and body posture. They are used for grammatical purposes and to modify the meaning of signs.

Speech has a linear structure: sentence is a sequence of words and a word consists of a sequence of sounds. Also in sign languages, signs are produced one after the other to formulate a sentence. However, morphemes can be expressed simultaneously. The use of the spatial dimension, in addition to the temporal one, gives additional possibilities to modify the meaning and inflection of words.

A distinct part of a number of sign languages around the world are fingerspelled alphabets. In many ways fingerspelling serves as a bridge between the sign language and the spoken language that surrounds it. It can be mainly used to spell out proper nouns and terms which do not have established signs, letter by letter. In the case of ASL, the letters are spelled with a single static hand located near the face, see Figure 2.1. Two letters, J and Z, include a motion, though. Other letters are characterized by their hand shape and orientation only.

Supplementary details on ASL can be found in [Klima and Bellugi 1979], for example. A recent survey of computer-based sign language recognition was published by [Ong and Ranganath 2005].

2.2 Hand Shape Recognition

The problem of estimating hand shape and orientation from images has been addressed by many researchers [Erol et al. 2007]. Their approaches can be roughly divided into the

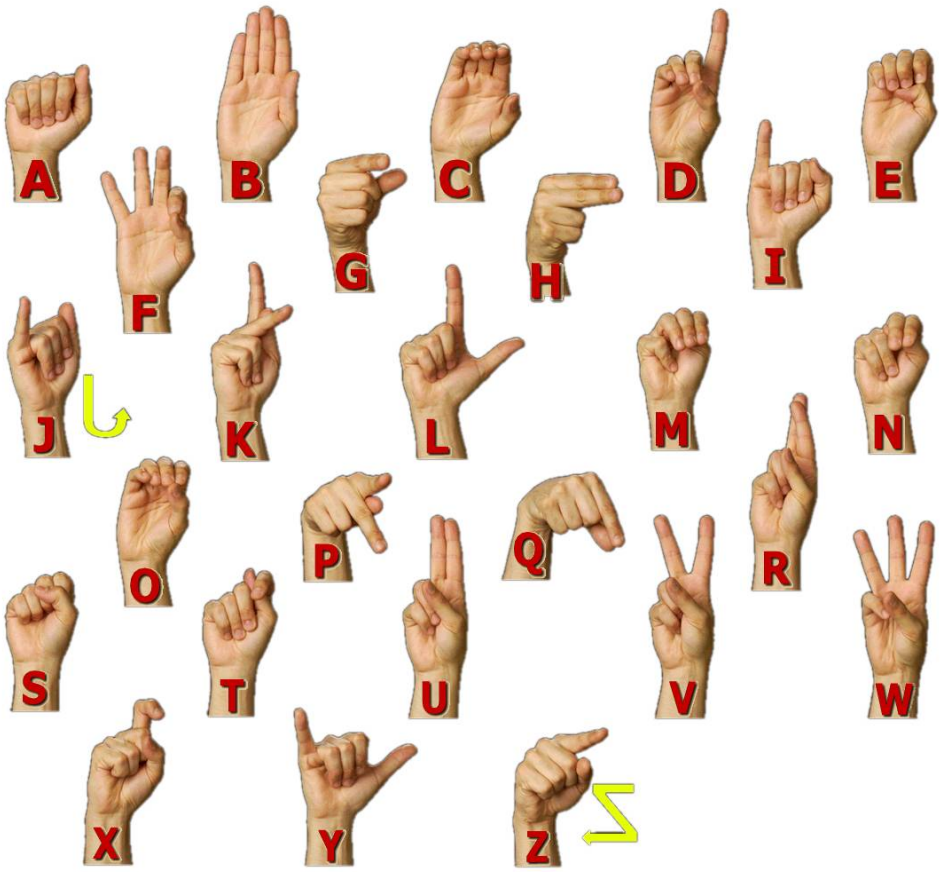


Figure 2.1 – American sign language fingerspelled alphabet. Letters J and Z include the indicated motion. Copyright © William Vicars [Vicars].

appearance-based or model-based categories.

Model-based approaches utilize an articulated 3D hand model for tracking. Usually, this is done in the analysis-by-synthesis manner by fitting projections of the hand to the input image. At each frame of the image sequence, a search in the configuration space is executed to find the best parameters that minimize a matching error, which measures the similarity between model features and features extracted from the input image. The search is started using configurations from the previous frames and a prediction of the dynamics. In the first frame, this information is not available and a separate initialization procedure is necessary. Typically, model-based approaches track multiple hypotheses for easier recovery from mistakes over longer image sequences. Many methods work in configuration spaces of lower dimensionality, constructed from training datasets. Although these systems can track general hand poses, estimate continuous parameters, and achieve viewpoint-independent recognition, they are often rather slow and require temporal context.

In appearance-based recognition, the task is usually formulated either as a classification problem or as a retrieval problem. In the former case, machine learning techniques are utilized to train a set of classifiers on real-world data ([Lockton and Fitzgibbon 2002] and

others). The latter approach uses a 3D hand model for building a database of images of all required hand shapes observed from different viewpoints in an offline phase. In online recognition, the hand shape can be retrieved by searching for the best match between the input image and the database of stored images ([Athitsos and Sclaroff 2002] and others). The configuration space of the hand is sampled discretely. As these techniques generally suffer from combinatorial explosion of the database, fast search methods are required. These include local sensitive hashing, learning mappings to lower dimensional spaces, or indexing techniques, to name a few. Appearance-based methods are typically applied to problems where the number of discriminated hand shapes is relatively limited.

The current state of art methods go beyond the appearance-based paradigm to allow for nearly continuous estimation of hand pose parameters. In [Wang and Popović 2009], the authors deal with the combinatorial explosion problem by using low-dispersion sampling to select a sparse database of samples from a dense collection of natural hand poses. Furthermore, each database sample is linked to a specific configuration and thus it is possible to compute the estimate by blending the configurations of k-nearest-neighbors of the input image. Additionally, inverse kinematics is used for extra accuracy.

This thesis is based on appearance-based recognition with a synthetic database. Since the retrieval method is not of the main concern in this study, we use simple exhaustive nearest neighbor search for retrieval, without any additional hand configuration refinements.

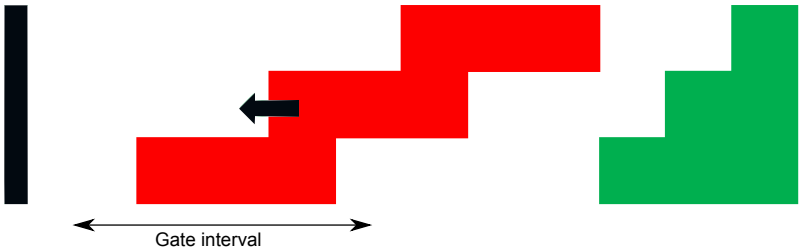
2.3 Time-of-Flight Cameras

Time-of-flight (ToF) cameras are devices used for range imaging, where a 2D image is produced showing the distance to points in a scene from the camera center. The cameras are relatively new devices that capture a whole scene at once and have no need for moving parts, thus offering a real-time stream of data. The measurement is based on the time-of-flight principle, which clocks the time needed for an actively emitted infrared light to reach the objects in the scene and becoming reflected back to the sensor of the camera.

There are two main technological implementations, depending on whether the illumination is modulated or not. The intensity modulation approach works as follows, see also the sketch in Figure 2.2a. The emitter sends out a light field with a radio-frequency modulated intensity. Demodulation of the reflected light is done by sampling. For each pixel, the phase shift between the emitted and detected wave is computed and based on it the distance is deduced, modulo the maximum range. To reduce the signal-to-noise rate the integration values are summed up over many periods. Some of the prominent manufactures are PMDTec and MESA Imaging, for instance.

This alternative approach is based on a fast shutter technique. The camera repeatedly emits a short light pulse which is reflected by the scene in a way which resembles the object's shapes (see Figure 2.2b). This information can be extracted by gating the returned signal with a rapid shutter in front of the sensor. The shutter opens for a short time interval which corresponds to the minimal and maximal distance of interest. The camera then measures pixel-wise the amount of light received when the gate was open. Additionally, all reflected light is captured to measure the reflectance of the objects. The ratio of the two

(a) The intensity modulation approach.



(b) The optical shutter approach.

Figure 2.2 – Illustration of ToF camera principles.

values is used to deduce the distance. The measured intensity values can be interpreted as distance values. The ZCam camera manufactured by 3DV Systems, used in this thesis, utilizes this technology.

We refer the reader to a relatively recent report of Kolb et al. [Kolb et al. 2009] for further details.

2.3.1 Sensor-related Artifacts

Besides the rather low resolution (current ToF cameras on the market offer spatial resolution of at most 320×240 pixels), there are other challenges to overcome.

Noise: The random noise level of the measurements is relatively high, typically in the order of centimeters. In the case of the ZCam device, we have observed that the noise gets stronger at more distant objects and also near corners, an effect likely to be caused by vignetting. However, the signer is supposed to sit relatively in the center of the view and not too far in our videos.

Flying pixels: This term refers to incorrectly measured distances along depth discontinuities. The object seems to melt with its background and precise measurements near the borders become difficult, see Figure 2.3. One reason lies in the limited resolution of the sensor chip: an affected pixel averages over the discontinuity as photons both from the background and the foreground are collected. Moreover, the incident angle reaches 90° near the object borders, resulting in a significant reduction of the number of reflected photons and increased noise. Since flying pixels appear in areas of high image gradient, we can remove them by thresholding the gradient at appropriate steps in our algorithms.

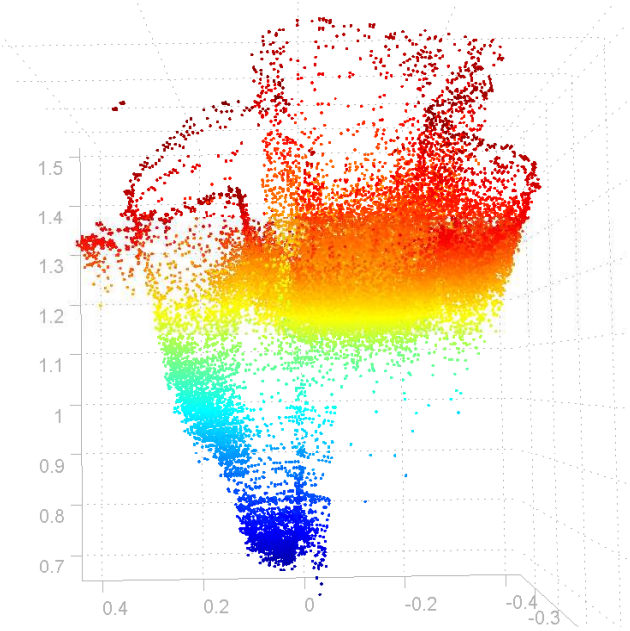


Figure 2.3 – Top view of a 3D point cloud of a signer (orange) with his hand in front (blue). Note the space between the hand and the body. While its left side represents the forearm, the right side consists purely of erroneous flying pixels.

Motion artifacts: In the case of a very fast motion, the boundary of the object may appear farther as less light gets reflected.

Systematic distance error: This error occurs only with some intensity modulation-based cameras due to imperfections in modulating the light.

Ray disturbance: Issues such as multiple reflections, specular reflections or direct sunlight lead to erroneous measurements.

2.3.2 Input Data Acquisition and Preprocessing

In this thesis, we are use a prototype of the product ZCam by 3DV Systems. The camera captures synchronized color and distance video streams of a spatial resolution of 320×240 pixels at 30 fps. The device requires its user to set an acquisition window, i.e., the minimal and maximal distance values to be acquired. The returned distance values are scaled linearly with 8 bit precision. A signer is expected to sit approximately 120 cm in front of the camera, which is sufficient to capture his upper body as well as his right hand signing space. To obtain the highest distance resolution possible, we set the window as narrow as allowed by the camera, which is 70-140 cm in out case. The theoretical distance resolution is then about 3 mm. However, an experiment done by analyzing 50 frames of a constant scene has shown that the standard deviation of the noise is slightly above 1 cm. A 3×3 median filter is used to reduce noise in the measurements.

The camera returns a distance map (also called a range image) $R(x, y) : \{1, \ldots, M = 320\} \times \{1, \ldots, N = 240\} \rightarrow [0, 1]$, where $R(1, 1)$ corresponds to the value of the pixel in

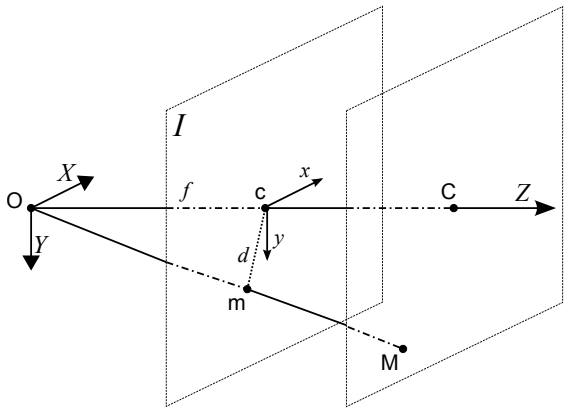


Figure 2.4 – The pinhole camera model with the virtual image plane I and principal point c is used to convert the distance value |OM|, as acquired by the camera, to the corresponding depth |OC| in the Cartesian coordinate system. The pixel $\mathrm{m}(x, y)$ is the projection of the point $\mathrm{M}(X, Y, Z)$.

the upper left corner. The gray level value of each pixel in the distance map represents the distance along the corresponding viewing ray to a point in the observed scene. Object farther than $\tau_{far} = 140\,\mathrm{cm}$ appear black while object closer than $\tau_{near} = 70\,\mathrm{cm}$ are white, see Figure 1.1 on page 4 for an example. Metric distance can be easily computed as

$$Z_{ray}(x, y) = \tau_{far} - R(x, y)(\tau_{far} - \tau_{near}) \tag{2.1}$$

As suggested above, each grayscale of the acquired data corresponds to a part of a spherical surface centered at the camera. Since this is inconvenient for our use further on, we remap the distance values so that each grayscale rather represents a plane in the Cartesian coordinate system. Let a pixel $\mathrm{m}(x, y)$ be the projection of the scene point $\mathrm{M}(X, Y, Z)$. Let O be the origin, $\mathrm{c}(c_x, c_y)$ the principal point, and C the intersection of the optical axis and a plane parallel to the image plane containing M as depicted in Figure 2.4. We define $Z_{ray} = |\mathrm{OM}|$ to be the distance of the point M acquired by the camera and $Z = |\mathrm{OC}|$ to be its depth in Cartesian coordinates. Furthermore, let f denote the focal length in pixel units and let $d(x, y) = \sqrt{(c_x - x)^2 + (c_y - y)^2}$ be the pixel-wise distance of m to the principal point. The mapping is performed using the pinhole camera equations as

$$Z(x, y) = \frac{f Z_{ray}(x, y)}{\sqrt{d(x, y)^2 + f^2}} \tag{2.2}$$

We can then define the depth image $I(x, y) : \{1, \ldots, M\} \times \{1, \ldots, N\} \to [0, 1]$ using the equation

$$Z(x, y) = \tau_{far} - I(x, y)(\tau_{far} - \tau_{near}) \tag{2.3}$$

Several algorithms mentioned in the following chapters need to work with 3D data. In this case, we can easily backproject the depth image into a metric 3D point cloud $\{(X, Y, Z)\}$, again using the pinhole camera model.

Computing a metric distance between two pixels in the depth image is sometimes required. Although this can be done by directly returning their distance in the 3D point cloud, such a distance is rather noisy. Moreover, if we wanted to mark all pixels within a certain metric radius, the region might consist of many isolated pixels, which could be difficult to handle. Hence, to be able to measure metric distances inside a certain region of interest, e.g. in a segmented hand, we approximate this region by a plane of constant depth and compute the metric distance in the 'flattened' point cloud of the region. We have experimentally determined the mean depth to be similar[1] to the median depth, therefore we use the mean depth value of the region as it is faster to compute.

[1] Estimating the depth of the segmented hand on the dataset introduced in Section 3.3, the difference between the two methods was 6 ± 4 mm.

Chapter 3

Segmentation

In the following chapters, we will examine descriptors which are region- or contour-based in their nature. Therefore, a proper hand segmentation is required as a preprocessing step, which we approach in this chapter. In the literature, the problem is usually simplified by making assumptions on the scene settings (e.g., concerning spatial constraints, background, camera position, or illumination) and the signer (motion restriction, body orientation limits, specific clothing, etc.). Especially for 2D images, this is a widely explored topic - see [Zabuli et al. 2009] for a survey. Common methods in the context of hand detection include skin color-based segmentation [Jones and Rehg 2002], motion cues [Habili et al. 2004, Holte et al. 2008], trivial thresholding [Ahn et al. 2009, Haubner et al. 2010], or clustering of 3D points [Malassiotis and Strintzis 2008]. Additionally, there is a class of object detectors which integrate segmentation with recognition, such as template matching [Triesch and von der Malsburg 2002] or algorithms for classification-based object detection [Ong and Bowden 2004].

As mentioned in the introduction, we restrict ourselves to work with single frames of depth data only. Using depth without any color information has its advantages and disadvantages. The benefit lies in reduced dependence on scene illumination and practically no restrictions on the signer's clothing. Moreover, unlike in color images [Smith et al. 2007], poses where a signer's hand occludes his face are much less problematic. On the other hand, poses where a signer (nearly) touches any part of his body present a challenge, especially without exploiting any temporal context. However, not relying on neighboring frames enables our method to handle fast and jerky motion, which is typical for sign languages. The results can thus be used for initialization and recoveries of tracking algorithms in possible future applications as well.

To simplify the problem and to make the segmentation more robust we make the following assumptions on the signer's position to the camera:

Assumption 1: The hand is the closest object to the camera.

Assumption 2: The hand does not touch any part of the body.

Assumption 3: The body is either roughly parallel to the camera plane or twisted with its active hand's shoulder in front.

Note that this still permits a signer to perform a variety of single-handed gestures as well as fingerspelling. Whereas Assumption 1 helps us to easily locate the hand, Assumptions 2 and 3 facilitate segmentation. The task of finding the hand in other positions could be solved by one of the recent fast full body trackers such as [Ganapathi et al. 2010, Knoop et al. 2009] or the trackers used in commercial products like Microsoft's Kinect. Correctly segmenting touching body parts would still be an issue, though.

Our segmentation procedure consists of two steps: first, we segment the forearm from the body (Section 3.1). Second, we cut only the hand out of it (Section 3.2). In Section 3.3 we will evaluate their performance on a ground-truth dataset. Based on this evaluation, we will fix the segmentation pipeline which will be used in the following chapters of this thesis. We will finish with a discussion on the idea of using multiple segmentation results at once.

3.1 Forearm Segmentation

Thanks to Assumption 1, hand localization is not a problem and we can rather concentrate on correctly delimiting it. The task in this section is to find a connected subset of a given depth image I corresponding to the signer's forearm. The hand and fingers are to be accurately delimited, while no exact cut-off position for the forearm has to be computed since this is will be tackled in Section 3.2. Segmenting also a portion of the arm or only a part of the forearm is hence acceptable.

In this section, we propose four techniques for segmenting the forearm from a depth image of the signer's upper body. The methods do not make use of any body or bone model. They are essentially operating on 2D depth images and all perform different variants of thresholding. However, note that we go beyond straightforward thresholding planes parallel to the camera plane: in Section 3.1.3 we design an algorithm for selecting the threshold for each row of the depth image individually.

3.1.1 Active Contours (AC2&4)

Active contours are a powerful and popular framework for segmentation in computer vision. The existing approaches can be categorized into two classes: edge-based, also called snakes, and region-based models.

Edge-based models [Kass et al. 1988] use local image features, typically the gradient, to stick to object boundaries. First, the contour is manually initialized roughly around the object of interest. Contour evolution is then guided by the imbalance of an external force (pulling the contour towards strong edges) and internal forces (controlling contour smoothness, area expansion/contraction and so on). The solution represents a local minimum of the energy functional. We were experimenting with an implementation of several types of snakes by D. Kroon [Kroon 2010]. However, we found it very difficult to set the initial contour as well as the multitude of parameters to get consistent results across a larger set of test frames. Therefore, we abandoned this class of models.

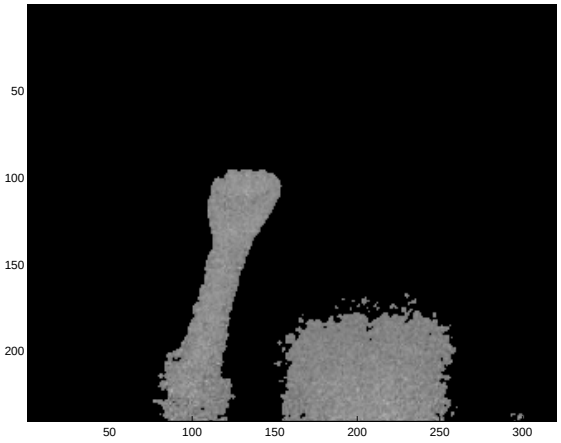
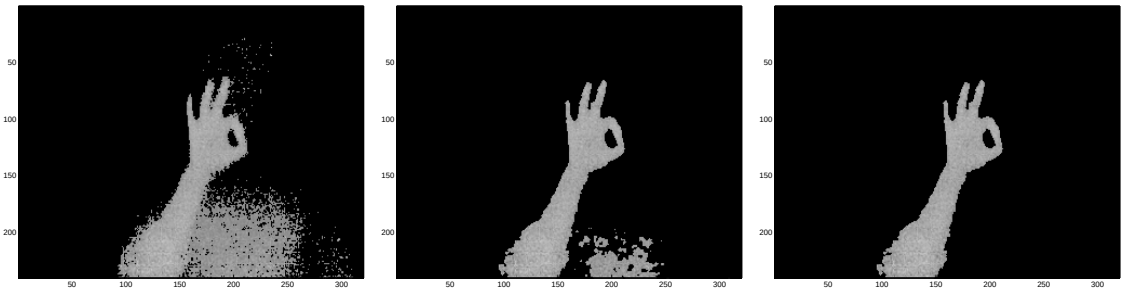


Figure 3.1 – Preference of elliptic components. Despite the abdomen being larger ($s_{abdomen} = 5765$, $s_{hand} = 4472$), its weighted size is smaller than the one of the hand (1778 and 2141 respectively), and thus it is discarded.

Region-based models [Mumford and Shah 1989] typically assume that the image is partitioned into several piecewise constant or smooth regions. Unlike common color images, depth images theoretically fulfill the piecewise smoothness assumption relatively well as they are not disturbed by textures and shadows (ignoring noise issues and other image acquisition artifacts). Moreover, the models have usually only a few parameters and often guarantee global solutions. Despite the recent efforts in speeding up the segmentation of piecewise smooth images [Bresson et al. 2007, Piovano et al. 2007], we are neither aware of any paper stating specific run times nor we were able to find any publicly available implementation which would perform reasonably fast. Therefore, we had to turn to models assuming piecewise constant images where the recent implementations are capable of running in the order of hundreds of milliseconds. In these models, the regions to be segmented are typically modeled as their mean depth values and the methods work well for images where the ideal segmentation is characterized by regions of quite different depth. Luckily, this should be more or less fulfilled under our three assumptions above, as the forearm and the upper body are supposed to be distinct in their depth.

We use a multiphase model of Ayed et al. [Ayed and Mitiche 2006; 2008, Ben Ayed et al. 2006] which partitions the image into a specific number of regions. It turned out that prescribing 4 regions appears to give reasonable results with 5 iterations necessary to produce a stable partitioning. Then, the region closest to the camera is picked. This region often consists of many connected components, though. We propose the following heuristic which prefers large elliptic components, hopefully being the forearm and not parts of the head or abdomen. Let λ_1 and λ_2 denote a component's larger and smaller Eigenvalues, $e = \sqrt{1 - \lambda_2/\lambda_1}$ a component's eccentricity and s its number of pixels. From the set of 5 largest components, the one which minimizes the weighted component size $e^2 s$ is chosen, see also Figure 3.1. We will refer to this method as AC4 from now on.

Additionally, we explore a fast implementation [Bresson 2009] of the widely used Chan-Vese model [Chan and Vese 2001] for binary segmentation. We rescale the input data to 80% and interpolate back the results to save running time. Note that more aggressive rescaling might already cause smaller features such as protruding fingers to be effectively ignored by



(a) Input image thresholded by taking the first 12 cm of the scene content.

(b) The image after running Chan-Vese. Note that the connection between the abdomen and the forearm has disappeared.

(c) The result after choosing the largest elliptic component.

Figure 3.2 – Active Contours 2

the segmentation[1]. Essentially, the Chan-Vese model separates the image into two regions of similar depth while keeping their boundary short. Without any assistance, the two regions always tend to be the background plane at τ_{far} and everything else. One remedy is to preprocess the input image: decide on a depth threshold τ and replace all depths larger than τ by the threshold itself, similarly to raising the water level in a tank, figuratively speaking. This removes the very isolated mode of maximal depth τ_{far} in the image and permits Chan-Vese to include some frontal parts of the body into the background region as the means of the both regions become less distinctly separated. In return, this fact slightly diminishes the possible consequences of an imprecisely selected τ because often any connections between the hand and the face/abdomen are destroyed (see Figure 3.2).

Nonetheless, it is very important to select the threshold τ correctly. We use a heuristic and set τ to cut off the nearest 12 cm of the scene content. It is an empirical compromise between the depth of an average hand (about 18 cm [Agnihotri et al. 2006]) when pointing directly at the camera and the permitted distance between the hand and the upper body, which we try to minimize. As in AC4, the last step of the algorithm is to select a large elliptic component from the closer region. This method will be referred to as AC2.

3.1.2 Volume Thresholding (VT)

The method presented in this subsection is a simple thresholding based on the following observation: if a plane parallel to the camera was placed between a signer's shoulder and elbow, the body volume in front of it would be more or less constant irrespective of the actual hand shape. Thus, if we cut off the fixed amount of body volume $V_{forearm}$ closest to the camera, we will segment the forearm. Since only the surface facing the camera is known, the problem is how to measure the volume given the depth images. We estimate the upper bound of this volume: given a depth threshold τ, we assume that all objects in front of τ are extended up to the threshold plane. The volume $V(x, y)$ of a pixel $I(x, y)$ is then defined as the volume of the frustum between $Z(x, y)$ and τ:

[1] This has proved to be the case with AC4 where downsampling by just 20% led to a loss of quality.

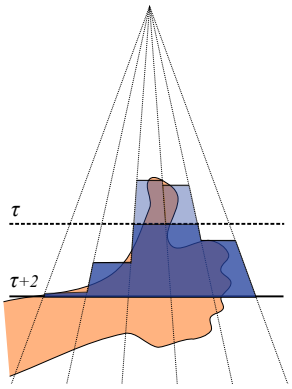


Figure 3.3 – Volume Thresholding (2D illustration, each ray represents a pixel)

$$V(x,y) = (\tau - Z(x,y))\frac{A_{Z(x,y)} + A_\tau + \sqrt{A_{Z(x,y)}A_\tau}}{3} \tag{3.1}$$

where A_z is the area of a pixel-sized square projected to depth z.

The algorithm starts by finding the biggest component within the nearest 5 cm of the scene content. The threshold τ is then set to this starting depth and iteratively increased by 2 cm. During each iteration, we grow the chosen component up to τ and compute its volume as the sum of $V(x,y)$ of the component's pixels. See Figure 3.3 for a general idea. The threshold producing a volume closest to $V_{forearm}$ is then the solution. Based on practical experiments, we set $V_{forearm} = 1500\,\mathrm{cm}^3$, which is unfortunately a quite signer-dependent quantity.

The worst case for this upper bound estimation is when the forearm is tilted by 45 degrees in which case one cuts off considerably less actual body volume but still, hopefully, at least the hand.

3.1.3 Scanline Thresholding (ST)

A common property of the methods presented above is that they seek an xy-plane parallel to the camera (w.l.o.g. upright) which can optimally separate the forearm and the upper-body apart. However, this may generally fail when the hand is close to the body. In that case, the face or the abdomen may have nearly the same distance to the camera as the hand, although the hand keeps a reasonable distance from both of them. This can be well seen by looking at the profile, i.e. the orthogonal reprojection of the data to the lateral yz-plane. The idea is to find a piecewise continuous curve (implying a cutting surface perpendicular to the yz-plane) in the lateral view separating the hand from the body well. See the red line in Figure 3.4b for an illustration. We find the curve by computing a depth threshold τ_i in each row $i = 1 \dots N$ (also called scanline) of the depth image independently and then enforcing some smoothness. The optimal location to set this threshold is the middle of the empty space separating the front surface of the hand and the front surface of the body. Mathematically, we try to detect whether each scanline's depth values come from a bimodal distribution and, if so, to optimally separate its modes. Note that as we

are considering the whole scanline for this, we usually have enough information about the depth of both of the surfaces (using Assumption 3). Scanlines not fulfilling this are to be dealt with by the smoothing.

An essential preprocessing step is removing flying pixels since they seriously spoil reliable detection of the empty spaces between body parts. We do this by removing pixels at depth discontinuities which are detected by thresholding the gradient. After that we start the algorithm by computing a histogram over the depth values with 5 mm wide bins for each scanline. For each histogram, we classify its bins as either an object or a gap based on the number of pixels binned in a 3.5 cm wide bin neighborhood. The bin is labeled as a gap if this number is less than 10, which was set experimentally. In the histogram of each scanline i, we determine:

- the position o_i of an object-bin closest to the camera,
- the position g_i of the first gap-bin in the longest uninterrupted sequence of gaps enclosed between object bins (if there is no such sequence, the furthest one from the camera is considered),
- and the length ℓ_i of the continuous sequence of gap bins immediately behind g_i. The maximum length is set to 20 cm, i.e. 40 gap bins.

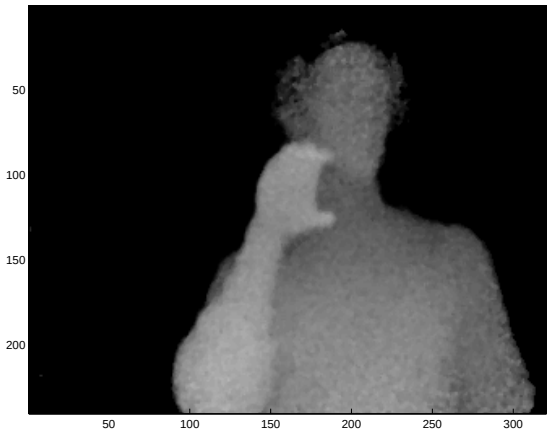
Thus, we have obtained one sequence of lengths $\boldsymbol{\ell}$ and two sequences of bin positions $\boldsymbol{o}$ and $\boldsymbol{g}$ (all of length N). The latter two are illustrated as curves in Figure 3.4b.

The next step is to smooth the sequence of gaps $\boldsymbol{g}$ and lengths $\boldsymbol{\ell}$ using a median kernel of size 15 scanlines. Median filtering preserves distinct discontinuities which are important in the following step.

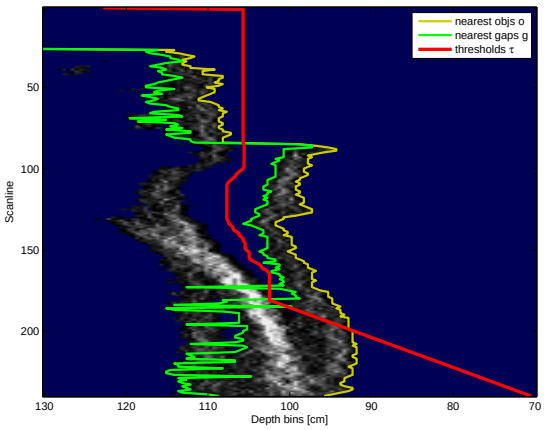
We now want to choose a subsequence $g_a, \ldots, g_b$ of some minimal length $L = b - a + 1$ which is as close as possible to the camera and doesn't contain any big discontinuities (see Figure 3.4c). Ideally, the range $a, \ldots, b$ corresponds exactly to the scanlines containing the empty space between the hand and the body which we are seeking to cut through.

We proceed in a loop until a suitable subsequence is found. Along the whole $\boldsymbol{g}$, we find the gap position g_s nearest to the camera and proceed to determine a and b, $a < s < b$, as follows. We search over all scanlines above s to find the lowest scanline a where the sequence gradient $\nabla\boldsymbol{g}$ is larger than a threshold: $(\nabla g)_a > T_{upper}$. Similarly, we find b below s such that it holds $(\nabla g)_b > T_{lower}$. If $L \leq 10$, we assume that the chosen g_s does not belong to the sequence of gaps separating the hand from the body and thus we restart the search using the nearest gap position outside the current subsequence. We use a sensitive $T_{upper} = 3$ as the hand and the face may be quite near, and a more robust $T_{lower} = 5$ for detecting jumps typically from the elbow to the back.

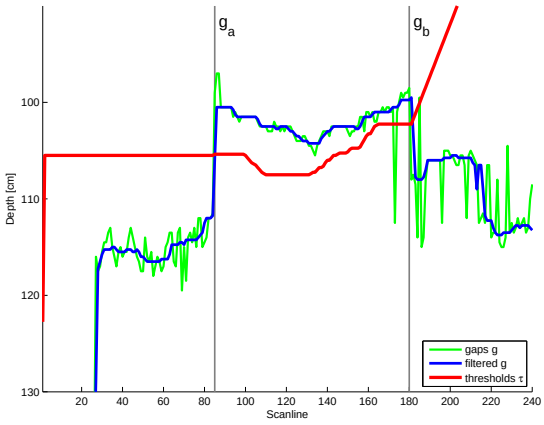
Once the subsequence $g_a..g_b$ has been chosen, the whole sequence of scanline thresholds τ can be computed. Let us explain the computation using Figure 3.4b as an example. The subsequence $g_a..g_b$ (scanlines 100-180) is shifted into the middle of the empty space separating the hand and the body. The thresholds for the scanlines above a (scanlines 1-100) are set to basically propagate g_a upwards so that the cut is carried out in front of the face. Staying in front of the face is ensured by adding half of the median depth



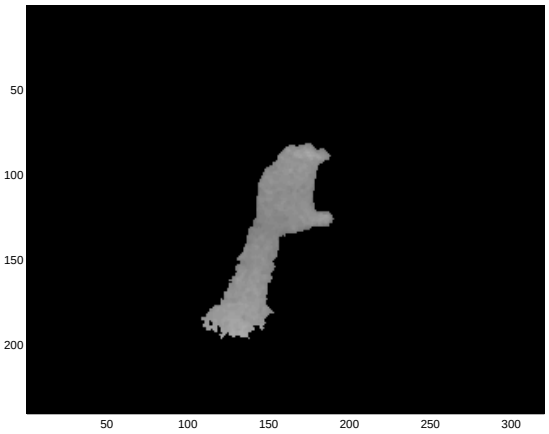
(a) Input depth image (xy-plane)



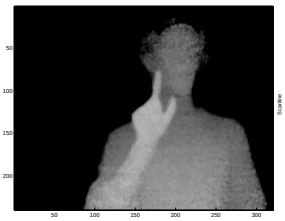
(b) Orthogonal reprojection to the side (yz-plane), each horizontal line represents a scanline's histogram over depth values. The whiter the bin the more pixels it contains. Bins classified as gaps are shown in blue. The meaning of the curves is explained in the text.

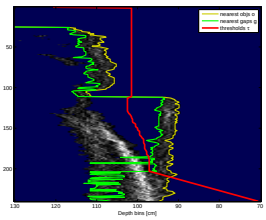


(c) The sequences from Figure 3.4b. After smoothing, the green subsequence $g_a \dots g_b$ likely corresponding to the hand is chosen.

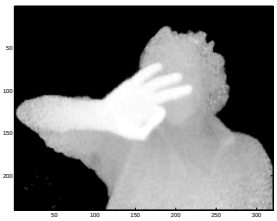


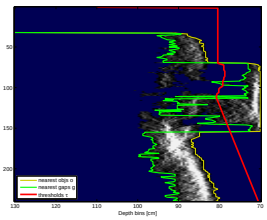
(d) The result after thresholding the image and picking the largest connected component.





(e) Example: despite the index finger being classified as gaps, the threshold is correct.





(f) Example: although a part of the hand is hidden by the forearm in the lateral view, the threshold is correct.

Figure 3.4 – Scanline Thresholding

between the upper end of the hand and the head. The thresholds for the scanlines below b (scanlines 180-240) are designed to linearly decrease towards the lower right corner, as it can be seen, e.g., in Figure 3.4f where this way of cutting ensures keeping the lower part of the hand despite it already being under the b scanline. Formally:

$$\tau_i = \begin{cases} g_a + \text{median}(\mathcal{H})/2 & i = 1, \ldots, a-1 \\ g_i + \ell_i/2 & i = a, \ldots, b \\ g_b - (i-b)\frac{\tau_{near} - g_b}{N-b} & i = b+1, \ldots, N \end{cases} \tag{3.2}$$

where the set $\mathcal{H}$ is given as $\mathcal{H} = \{o_i - g_a \,|\, i = 1, \ldots, a-1 \wedge o_i \geq g_a\}$.

Afterwards, the sequence $\tau_1 \ldots \tau_b$ is smoothed by a large median kernel. Finally, the image is thresholded and the biggest connected component selected as the result.

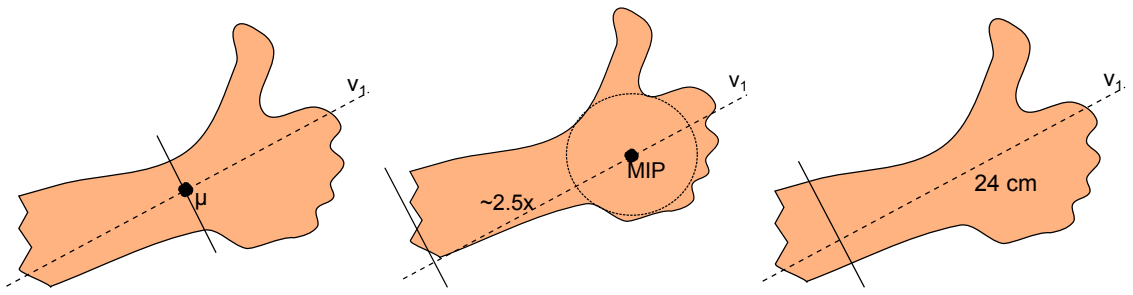
As an alternative, we tried using Otsu's algorithm [Otsu 1979] for bimodal histogram splitting. If the threshold τ_i proposed by Otsu hits a gap, it is accepted, otherwise it's set to infinity (we are most likely dealing with an unimodal histogram then). We filter the Otsu's τ the same way as $\boldsymbol{g}$ above to pick the nearest subsequence. The performance on our test dataset (see Section 3.3) is very similar to the original algorithm with Otsu being twice as slow, therefore this alternative isn't further considered.

3.2 Hand Segmentation

The forearm segmented by one of the algorithms from the previous section can be of a relatively arbitrary length, depending on the whole signer's pose. Yet, for further processing described in Chapter 4 it is convenient to have just the fingers, palm and a couple of centimeters of the wrist. We transform the shape into the basis of its eigenvectors using PCA and we assume that the major eigenvector $\boldsymbol{v}_1$ is more or less parallel along the hand's pointing direction. Note that due to nonzero forearm lengths this is typically fulfilled for both open and closed hand shapes. In the following we will discuss three possibilities of leading the cut, see also Figure 3.5.

Midpoint Cutting is cutting near the origin of the PCA basis, defined by the mean value $\boldsymbol{\mu}$ of the segmented forearm region. This was essentially proposed in [Malassiotis and Strintzis 2008]: they model the 3D point cloud as a mixture of two Gaussians $N(\boldsymbol{\mu}_\text{i}, \Sigma_i)$ and obtain their properties as the maximum likelihood estimate via expectation-maximization (EM). The result depends on the proper initialization. Both means $\boldsymbol{\mu}_\text{i}$ were selected along $\boldsymbol{v}_1$ in a specific distance from $\boldsymbol{\mu}$, which was 1/3 (respectively 2/3) of the maximal point cloud distance in each direction. The same numbers were also used to initialize the variances. We found out that the EM iterations don't change the result too much on our dataset, therefore classifying approximately half of the space as a hand and the other half as a forearm. This of course heavily depends on the stability of $\boldsymbol{\mu}$ with respect to different segmentations, which is very low in our case.

Proportional Cutting is meant to address this issue. The point having the maximum



(a) Midpoint Cutting. **(b)** Proportional Cutting **(c)** Top Cutting

Figure 3.5 – Hand Segmentation

distance to the closest boundary has been argued [Koike et al. 2001] to be more stable under changes in a silhouette shape than the mean (the center of gravity). This point, from now on the most inner point (MIP), can be easily found as the maximum r_{dt} of the distance transform of the silhouette. If there happen to be multiple maxima, their mean position is taken. As before, we model the shape as a mixture of two Gaussians: $\boldsymbol{\mu}_1$ is set to be the MIP and $\boldsymbol{\mu}_2$ to be $5r_{dt}$ away along $\boldsymbol{v}_1$. The variances are set roughly equal. We don't run any EM iterations and directly classify. The method works very well as long as the MIP roughly corresponds to the center of the palm. This is sadly not the case when either the hand shape is very thin (e.g., a flat hand perpendicular to the camera plane) or the whole elbow gets segmented, possibly with a piece of the body or the arm.

Top Cutting is as simple as taking the first 24 cm of the content (130% of the average hand length [Agnihotri et al. 2006]), measured along $\boldsymbol{v}_1$. The disadvantage is that different hand shapes, such as an open hand and a closed fist, yield different forearm lengths then. For example, there are problems with hands pointing towards the camera, see Figure 3.7c. However, as this approach performs consistently well through all hand shapes and poses of our dataset, we have decided to use this method. We run this algorithm on 2D data to save some running time with only a slight loss of cutting quality.

Note that we have implicitly assumed that $\boldsymbol{v}_1$ is oriented along the hand. However, the sign of $\boldsymbol{v}_1$ is not defined and is PCA implementation-dependent. We use a heuristic to solve this by looking at the gradient along the contour of the segmented region in the original depth image. The observation is that the hand-end is typically surrounded by depth discontinuities while the forearm-end has been cut more or less arbitrarily and a smooth surface is expected there. The implementation compares the median gradient of the contour within the first and last 5 cm of the shape along $\boldsymbol{v}_1$ and orients the major eigenvector to point towards the fingers. The method works as expected on the absolute majority of frames in our segmentation test dataset. In one frame the orientation was flipped, which of course affects any further processing.

(a) An RGB image from the test dataset.

(b) The corresponding depth image.

(c) ST segmentation result in blue, red being the ground truth from a).

Figure 3.6 – Empirical evaluation

3.3 Empirical Evaluation

In this section we evaluate the computational and qualitative performance of the four segmentation methods presented in Section 3.1 while using 'top cutting' as hand segmentation for all of them.

The implementation of the segmentation was done in Matlab 7.4.

We have acquired both color and depth video of a person performing 11 fingerspelled letters with his right hand in three different locations: next to the body, near the face and near the camera. The test dataset contains 383 equidistantly sampled frames. The signer wears a green surgical glove, based on which the ground truth silhouette (GTS) is generated. The glove was automatically segmented by a learned 'skin'/background Gaussian model and manually corrected for imperfections afterwards. The ground truth is however not perfect due to the very poor image quality of the RGB camera. Moreover, the RGB camera has a longer exposure than the Z camera and the images appear not to be synchronized in time in the case of fast hand movements. Additionally, the RGB and Z cameras are not properly calibrated, the fact also observed by [Schuon et al. 2008], which we compensate for by manual tweaking. Nevertheless, a comparison to the GTS gives a rough estimate of the segmentation quality.

We consider seven benchmark criteria:

Run time per frame is the average run time in milliseconds for the whole segmentation pipeline measured by Matlab's tic-toc on an Intel Core2Duo P8400 laptop with 2 GB RAM.

Misses is the relative number of the frames when the GTS and the segmentation result have no intersection (Figure 3.7a).

The following metrics are averaged over the frames which are *not* classified as misses.

Pixel-wise precision measures the fraction of true positive pixels in the silhouette produced by the segmentation.

	AC4	AC2	VT	ST
Run time per frame	336 ms	216 ms	93 ms	124 ms
Misses	30%	36%	35%	26%
Pixel-wise precision	60%	88%	84%	91%
Pixel-wise recall	86%	81%	87%	84%
Pixel-wise F-measure	0.70	0.84	0.85	0.88
Pixel-wise overlap	55%	75%	74%	77%
Shape flaws	52%	6%	11%	9%

Table 3.1 – Evaluation results.

Pixel-wise recall measures the fraction of the GTS segmented.

Pixel-wise F-measure is the harmonic mean of precision and recall and constitutes a summary score for the algorithm.

Pixel-wise overlap is the ratio $|\,\text{GTS} \cap \text{result}\,| \,/\, |\,\text{GTS} \cup \text{result}\,|$.

Shape flaws is the relative number of frames where the important part of the hand is either incomplete or connected to a part of the head. This is a subjective measure evaluated by the authors.

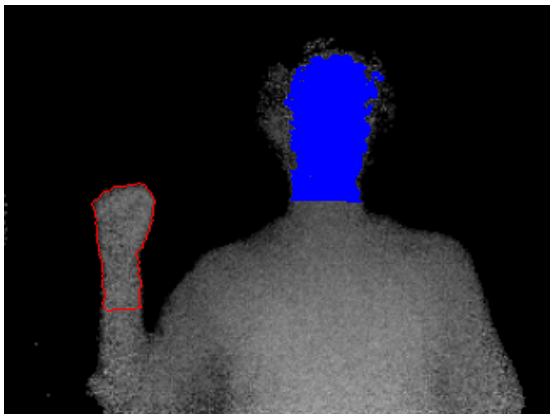
The results can be seen in Table 3.1. Our implementations unfortunately won't allow rates faster than 10 fps for the whole system. About 40 ms of each run time per frame is the time taken by the 'top cutting' algorithm. Besides the external AC solvers, the biggest drags in performance are convolutions, gradient computations and possible morphological operations. We believe the run time would considerably decrease by using a fast C implementation of these.

For all methods, something different than the hand is segmented in roughly one third of the frames. These are the poses where the hand is besides the body, which violates Assumption 1 about the hand being the nearest object.

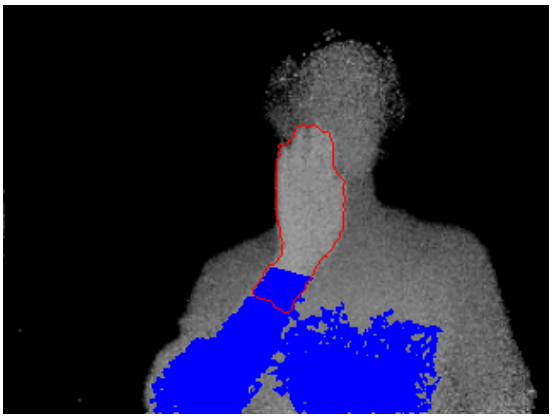
AC4 has shown to have the distinctly worst performance both in time and quality. The forearm is usually segmented together with a large part of the body, which explains the high recall and low precision. This suggest that applying otherwise powerful methods without any prior knowledge is insufficient.

AC2 gives one of the best results. The credit goes to a relatively conservative threshold that ignores the hand unless it is quite in the front, which results in the largest percentage of misses. As noticed in Section 3.1.1, pure depth thresholding can create a lot of connections (also called bridges, see Figure 3.7b) between the hand and the upper-body when the hand is close to them. While the active contour method can typically manage to break those leading to the face, many of those linking the forearm with the abdomen are preserved.

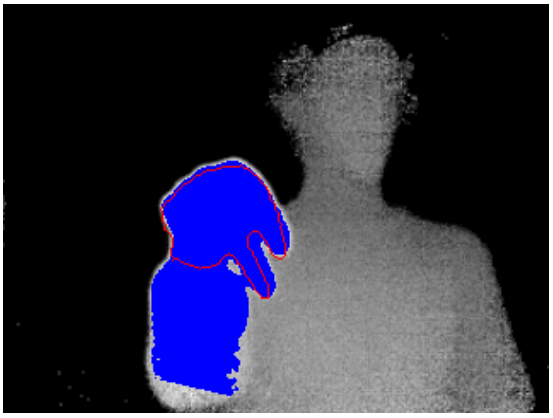
VT is the fastest algorithm and its threshold appears to be conservative too, with a similar number of misses as AC2. The quality is slightly lower, though. This is due to the fact that



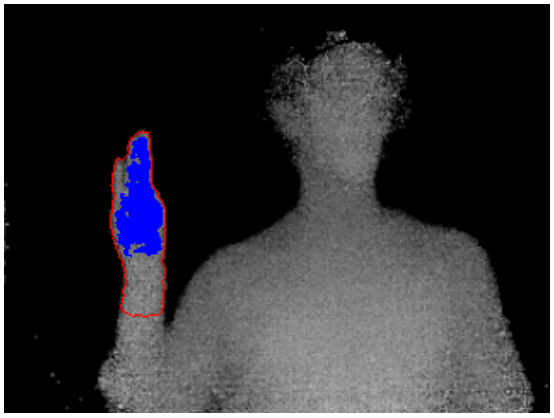
(a) A miss: the head is preferred as it is closer to the camera (ST).



(b) A bridge: the forearm is connected to a part of the body, confusing the successive PCA analysis (VT).



(c) A common problem with tilted hands: fingers are not distinguished from the forearm behind them (ST).



(d) An incomplete segmentation when the hand is only marginally closer than the face (ST).

Figure 3.7 – Segmentation failure examples. Proposed segmentation is in blue, ground truth in red. For a correct segmentation, see Figure 3.6c.

when the region of interest is accidentally grown into a part of the body via some noisy bridges, its volume is expanded beyond the threshold. This leaves the hand's more distant parts (e.g., thin fingers which sometimes appear further than they really are with the 3DV camera we use) being not fully segmented. Unlike AC2, no active contour computation is used to break even the thinnest bridges, and thus it's precision is the second lowest in our test.

ST is the winner in terms of quality. While its shape flaw percentage is only the second best to AC2, ST has the lowest amounts of misses. We have determined that if ST was to process only the frames not missed by AC2, its shape flaw percentage would be superior. Moreover, as the run time is acceptable among the compared methods, ST segmentation has been chosen and used in the following chapters.

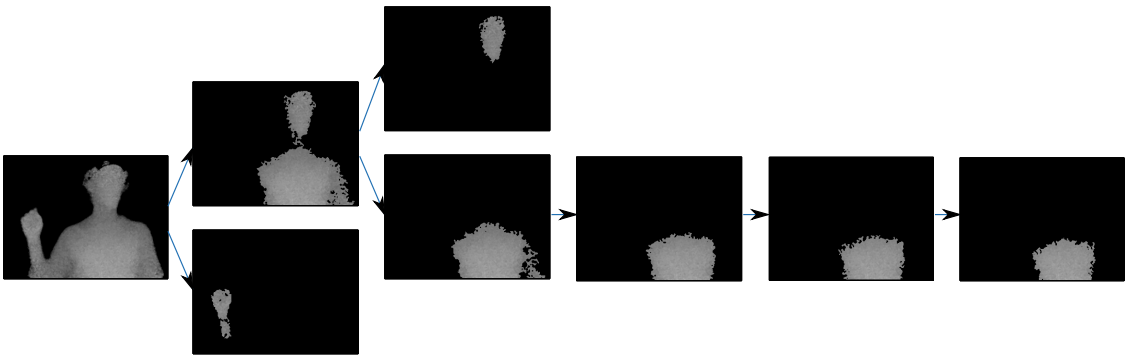


Figure 3.8 – An example of MSER regions ordered in a component graph tree. Note the component showing the hand only.

3.4 Segmenting Multiple Hypotheses

Our system's processing pipeline is designed in a manner where the recognition strongly depends on the quality of the segmentation, which is done without any high-level understanding of the segmented region. Ideally, we would like to segment the hand in a way which would provide the best matching to a certain database element. In this section we contemplate the possibility of keeping multiple hypotheses and postponing the decision on the correct segmentation to the recognition phase. To accomplish this, a relatively low number of segmentation candidates is chosen and a descriptor of choice is computed for each of them. The set of descriptors is then compared to a database and the most similar match is taken. Note that this is different to trying to find every database pose/class in the input image and choosing the best score, as proposed in [Triesch and von der Malsburg 2002].

We choose a set of candidates using a thresholding strategy again. Instead of sampling thresholds every 2 cm as with the VT algorithm, we used an implementation [Vedaldi and Fulkerson 2008] of maximally stable extremal regions (MSERs) [Matas et al. 2004]. Each MSER is a connected component with an area stable over a range of depth thresholds. The extremal regions can be ordered in a component tree by the inclusion relationship. The root includes all pixels of the input image and and each of its children was extracted using a lower depth threshold, see Figure 3.8 for an example. This provides us with a relatively low amount of hypotheses, usually under 15 with our selection of parameters. We have observed that at least one region typically corresponds to the forearm in our settings. More specifically, the best MSER hypothesis was manually chosen for each frame from the test dataset utilized in the previous section and the result benchmarked to obtain a pixel-wise F-measure of 0.85 with miss rate of only 18%. This shows that when being able to select the correct region, the overall quality of the result might outperform the methods above.

To obtain an automated selection of the correct MSER, we have conducted two experiments using the descriptors introduced in Chapter 4. First, we compared the descriptor of every MSER to our database and selected the one with the minimum distance. Second, we extracted MSERs from our database consisting of rendered hand images (Chapter 5) as

well and then tried to match the sequence of database MSERs[2] against each branch of the component tree generated by the query. The distance was then aggregated for every branch (e.g., a mean over the matches) and the best branch was eventually chosen.

However, the performance in terms of classification accuracy of the whole recognition process, as evaluated in Chapter 5, was worse than with a single segmentation hypothesis consistently over all descriptors. The run time was obviously longer as well. One possible explanation is that due to the broad variety of hand shapes under different viewpoints it is relatively easy for a segment of an unrelated body part to spuriously match a certain database element well enough to win over any semantic match. Moreover, the MSERs extracted from the database images often appeared to be visually different from those obtained from real depth images, an issue likely to be caused by the perfect quality of the rendering. For example, fingertips pointing towards the camera were often an MSER in database images, which has been never been observed in real data. One of the possible improvements is to add a hand/no-hand classifier similar to [Ong and Bowden 2004], which would reduce the number of hypotheses leading to a lower run time and hopefully a better accuracy. Investigating this is left to future work, though.

[2]It was assured that the database component tree is actually linear.

Chapter 4

Hand Descriptors

The research on content-based retrieval using regions of interest descriptors is vast. Technically, both 2D and 3D approaches are applicable to depth images as it is possible to easily convert between the two representations.

3D content can be described either globally or locally [Tangelder and Veltkamp 2004], generally speaking. Global methods aggregate information across the whole content, often in the form of statistical moments or Fourier transform coefficients, whereas local feature-based methods describe the 3D shape around a number of surface points. Global 3D descriptors do not seem to be appropriate since complete 3D models are required and this is rarely the case with 2.5D depth data. However, new global, object's centroid-based descriptors designed for ToF depth images have been published recently in the context of body tracking [Schwarz et al. 2010]. Local descriptors are extracted on some distinct key points and utilized for complete or partial surface matching. Surface properties, such as curvatures and normals, are typically used to detect key points and certain features of the neighborhood are summarized in a histogram. Examples in the context of range images include [Bayramoglu and Alatan 2010, wai Rachel Lo and Siebert 2008, Taati et al. 2007]. However, to the best of our knowledge, all papers use high quality (typically LIDAR) data for evaluation and it is unclear how the descriptors would perform on ToF depth images, which have more challenging noise characteristics.

Although most of the latest papers utilizing ToF cameras concentrate on exploring the 3D nature of the data, we want to investigate the opposite approach in this thesis. The hypothesis is that 3D descriptors might be unnecessarily powerful for 2.5D and extended 2D descriptors specifically designed to operate on ToF depth images might perform well enough. We set our thesis in the context of appearance-based methods where the 3D object (i.e., the hand) is represented by multiple projections, as described in Chapter 2. Here, the goal is to match an input depth image to one of the database images using 2D or 2.5D features.

There have been numerous 2D image features proposed for various computer vision applications. To gain an overview over the state-of-the-art we refer to the survey papers [Roth and Winter 2008, Yilmaz et al. 2006]. In the following, we describe the presegmented hand shape by its contour and by depth histograms of its interior. Note that as in Chapter 3, we

are using depth images only without any color clues. However, unlike with segmentation where the color might be a source of confusion, the additional color information would be of a benefit here, especially for the description of the hand's interior. The reason is that the human hand is relatively textureless by itself and any edges might provide additional hints on the finger position, particularly for closed hand shapes. Nevertheless, the goal here is to investigate how far one can go with depth information only and, moreover, not all ToF cameras are equipped with an RGB sensor of an acceptable quality. Evaluating the benefit of color data is a part of our future work.

The design objectives for a descriptor appropriate for an appearance-based hand shape recognition are as follows:

- Exploit depth information.
- Gain invariance to translation, scale, and in-plane rotation. Invariance to reflection or distinct out-of-plane rotations is not demanded.
- Be sensitive to local features, such as finger positions and angles, while being robust to sensor noise, minor segmentation artifacts, inexact wrist delimitation, and subtle changes in viewpoint or hand pose.
- Keep computational time as low as possible, both for extraction and matching.

We assume that the hand was segmented accurately, and thus its contour is supposed to be correct, while we still allow for a certain variation in the hand/forearm cut. Moreover, we do not expect any occlusions.

The chapter is structured as follows. In Section 4.1 several descriptors capturing the 2D interior of a hand are presented, while in Section 4.2 descriptors based on contours are introduced. In the last section, we discuss methods that aggregate both approaches.

4.1 Distribution-based Region Descriptors

In this section, several descriptors capturing the 2D interior of a hand are proposed. In contrast to silhouette shape descriptors, the actual depth variations of the hand area is also expressed. The method introduced in this section places a single histogram-based descriptor at the hand's central point. We present several variations in the design and content of the histogram. We do not treat any parts of the hand special in terms of, e.g., explicitly detecting fingers [Al-Hamadi et al. 2010] as we believe this might decrease the overall robustness of the method in case of misdetections.

4.1.1 Estimation of the Center, Radius and Orientation

Our experience shows that the robust localization of a centralized descriptor is one of the crucial factors determining the overall retrieval accuracy. Although the literature nearly unanimously proposes the centroid (the center of gravity or, equivalently, the component-wise mean) as the point of interest, other options exist.

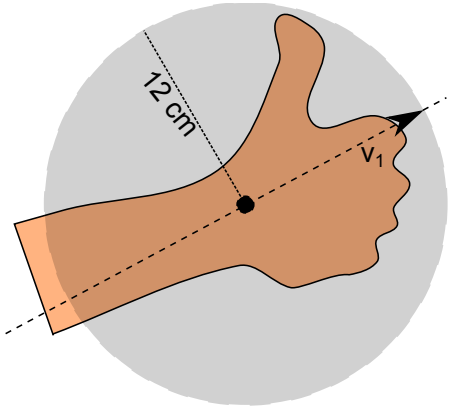


Figure 4.1 – A general idea of a centralized descriptor design: median-shift algorithm used to find the location, 12 cm radius and orientation by the major eigenvector $\boldsymbol{v}_1$.

For example, the geometric median, which generalizes the 1D median as the point minimizing the sum of distances to the sample points, is one of them. It can be computed using Weiszfeld's algorithm [Weiszfeld 1937], a form of iteratively re-weighted least squares. Note that simply taking component-wise median of pixels coordinates is not rotation invariant.

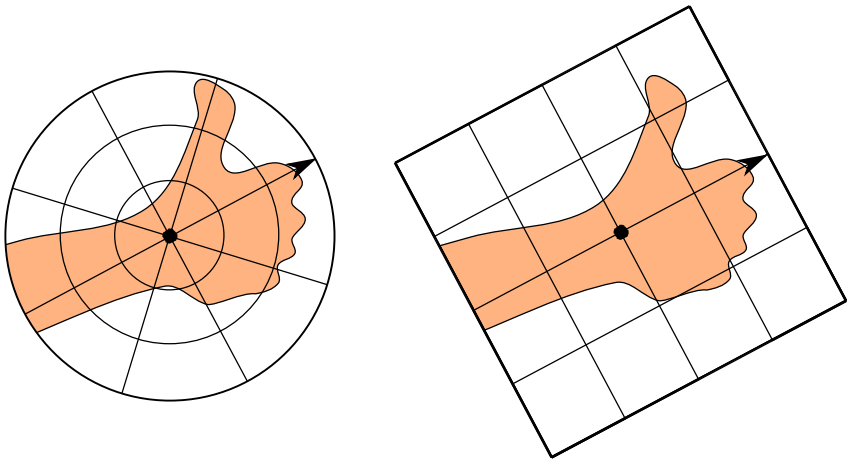
The point having the maximum distance to the closest boundary (MIP) has been argued [Koike et al. 2001] to be more appropriate for defining the center of a hand. We refer the reader to Section 3.2 for more details.

Wang and Popović [Wang and Popović 2009] use a mean-shift algorithm with a uniform kernel and variable bandwidth to crop the hand out of an image. We follow their implementation: we start at the centroid of the hand with a bandwidth that spans the entire depth image. After each iteration, the bandwidth is adjusted as a multiple of the standard deviation σ of the distances of pixels within the current bandwidth to their mean. In our case, 4σ is used. This procedure is iterated until convergence of the mean.

Additionally, we explore a variant with the bandwidth set to 12 cm. Its corresponding pixel-wise distance is recomputed at each iteration based on the mean depth of the pixels within the current bandwidth. Finally, we also use the geometric median instead of the arithmetic mean to produce a median-shift algorithm.

We have evaluated all six proposed methods in a comprehensive benchmark in Section 5.3. Based on the evidence, median-shift algorithm with a uniform kernel and 12 cm bandwidth has been chosen. The bandwidth is used as the radius of the descriptor as well. Since the average hand length is about 18 cm [Agnihotri et al. 2006] and we assume that the geometric median isn't too far from the center of the palm, the diameter of 24 cm should be able to accommodate the whole hand in any pose and viewpoint. Note that the fact that the radius is defined in centimeters and not in pixels facilitates scale invariance. Unlike radii based on the mean distance of pixels to the center of the descriptor, a fixed radius is more robust to confusing open and closed hand shapes.

Additionally, we make the descriptors rotationally invariant by orienting them along the major eigenvector $\boldsymbol{v}_1$ of the segmented hand. The general idea can be seen in Figure 4.1.



(a) Circular configuration **(b)** Grid configuration

Figure 4.2 – Example descriptor cell configurations

4.1.2 Descriptor Representation

The location $\mathbf{c} = (c_x, c_y, c_z)$, radius r and orientation impose a local 2D coordinate system in which we describe the image region, and therefore facilitate invariance to these parameters. The next step is to compute a descriptor that is distinctive and robust to remaining unimportant variations.

We address the descriptor's design from two rather orthogonal perspectives: the binning of image features into spatial cells and the actual representation of the features in each cell. The features are extracted for every pixel of the hand (x, y) and are either the pixel's depth $I(x, y)$ or its gradient magnitude $|\nabla I(x, y)|$. The gradient is computed using centralized differences on a segmented hand's image presmoothed by Gaussian kernel with $\sigma = 4\,\text{px}$, where the influence of pixels outside the hand is ignored.

There are two main trends in the layout of cell configurations of a descriptor, namely the circular and grid ones, see Figure 4.2. A circular layout (e.g., [Belongie et al. 2002]) has often been used as a local descriptor in a log-polar fashion, which aims to cope with the increasing spatial uncertainty of distant pixels due to possible image deformations around the point of interest. In our settings, however, the informative parts of the hand, i.e. the fingers, are often located further away at the boundary. Therefore, we use a circular layout with regular rings. Note that despite this, the outer cells are larger than the inner ones due to the longer circumference of the outer rings.

A regular grid layout (e.g., [Lowe 2004]) is the other option. We use a square layout with its side length equal to $2r$. The cells are of the same area everywhere in the grid.

We believe that the circular layout might be more invariant to slight rotations of the hand about its center, whereas the grid layout could be more robust to small translations. Regarding the implementation, no anti-aliasing technique for reducing sampling artifacts is used. Still, due to the relatively large area of the cells and using aggregating functions as histogram or mean of the cell content (described below), the descriptor should not change

too abruptly when some pixels shift from one cell to another. Any part of the depth image outside the descriptor's radius is ignored.

Four alternatives of cell content representations are considered:

Point count (PC) is simply the number of pixels in a cell. The vector of point counts for each cells is then normalized to unit length to account for different scalings. This representation utilizes the silhouette only and is of a similar concept to occupancy maps, where the normalization is performed cell-wise by the area of each cell. We have not found consistent differences in accuracy between the two normalizations in our experiments.

Mean depth difference (MDD) is the difference of the mean depth value within the cell to the mean depth c_z of the whole histogram. Only pixels not marked as flying pixels are considered, which is accomplished by thresholding the gradient of the whole depth image.

Depth histogram (DH) is a normalized histogram of depth values inside the cell. The motivation is that simple mean depth is prone to canceling of differences with opposite signs. For example, a hill and a valley within a cell could then be similar to a flat area. Using a histogram avoids this problem. We use 5 bins centered around c_z with bin diameter set to 2 cm of depth, the double of standard noise deviation. Votes are interpolated linearly between the neighboring bin centers to reduce aliasing. Again, flying pixels are discarded.

Mean gradient magnitude (MGD) is the mean gradient magnitude within the cell.

Each of these representations produces a vector of measurements which is then to be compared with others. We were experimenting with a range of distance metrics, namely the L^1 and L^2 distances, cosine distance, and measures appropriate for histograms (χ^2 and histogram intersections). Surprisingly, the differences in accuracy during evaluations against the training set (Section 5.2) were more or less negligible or spoke for the L^p distances. For that reason, we work with L^2 distances in any further evaluation.

Additionally, we need to compensate for different cell significance in all representations but the point count. The reason is that these cells do not reflect any reliability information: a mean computed from 1 pixel is represented the same way as a mean of 1000 pixels. Therefore, we weigh the distance contributions of each cell by their respective relative point count. Formally, let vectors $\mathbf{p}$ and $\mathbf{q}$ be two descriptors of D cells, p_i,q_i ; $i = 1, \ldots, D$ their cells, and n_{p_i}, n_{q_i} relative counts of points in each cell (similar to the normalized point count vector but accounted for flying pixels). Then the distance between $\mathbf{p}$ and $\mathbf{q}$ is computed as $||(\mathbf{p} - \mathbf{q})\mathbf{w}^{\mathrm{T}}||$ where $\mathbf{w}$ is the weight vector with

$$w_i = \frac{2}{\pi} \arctan(500 \frac{n_{p_i} + n_{q_i}}{2})$$

The function, the plot of which can be seen in Figure 4.3, is designed so that it reduces the influence of cells having less than approximately 0.5 % share of the points. The constant was set experimentally. This weighting consistently increases the accuracy of the

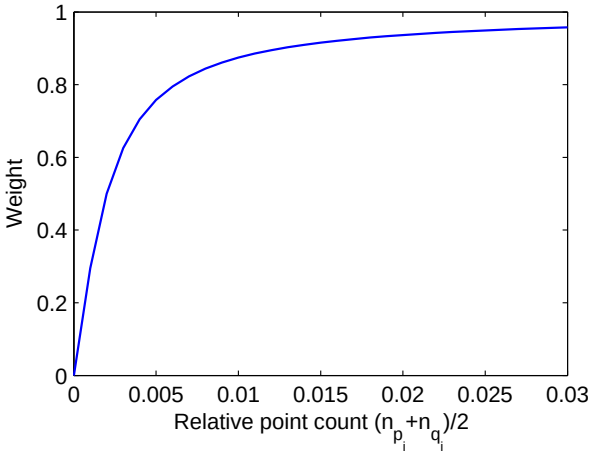


Figure 4.3 – Weighting function used to reduce the influence of cells with low relative point count when computing distances between two descriptors.

three representations considered. Another option is to use absolute point counts and a different constant in the weighting function. However, this has been shown not to perform consistently worse or better.

The evidence in Section 5.4 shows that the circular configuration of size 4×28 cells has the best performance. Moreover, silhouette-only point count representation is distinctly better than other representations. However, in a strive for utilizing any depth information, we further combine it with the mean depth difference representation in Section 5.4.1 to gain a slight improvement in performance. Our final distribution-based region descriptor thus consists of both representations.

4.2 Contour-based Descriptors

Contour-based descriptors capture only the points sampled along the boundary of a shape. As an alternative to histogram-based region descriptors, we propose two algorithms based on partial contour matching in this section. First, we explain the reasons suggesting the use of partial matching. Afterwards, our method of extracting sequences of 3D contour points is presented in Section 4.2.1. Finally, we describe both algorithms in Sections 4.2.2 and 4.2.3.

Most of the current research tries to achieve excellent retrieval of relatively complicated shape classes with high intra-class variability. However, the classes are typical quite distinct, as in, e.g., MPEG-7 silhouette database, which is currently one of the popular databases for shape matching evaluation. On the contrary to this, our task is to discriminate among hand silhouettes of different hand shapes captured from different perspectives, i.e., to distinguish a lot of very similar classes of nearly no intra-class variability. This raises two issues which are discussed in the following.

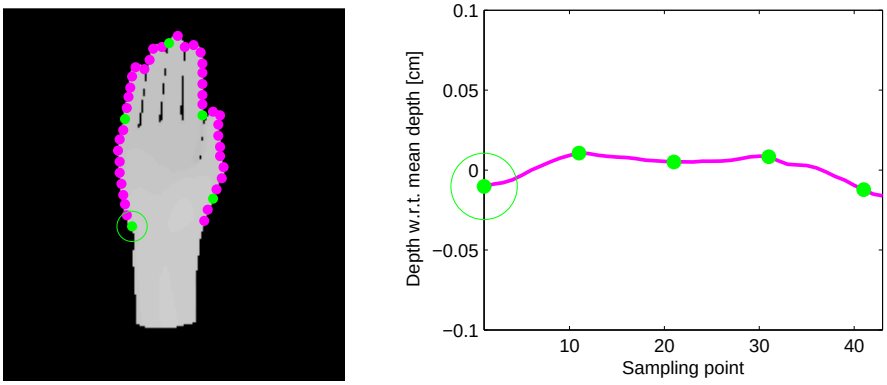
First, contour has to be represented quite accurately in order not to easily match a too broad set of shapes. This is, however, not the case for all methods. For example, we

were experimenting with shape contexts [Belongie et al. 2002] in the initial phase of this thesis. Their approach works by sampling points along a contour and extracting a log-polar histogram, so called shape context, of the relative positions of the other points at every sampling point. To reduce the complexity of finding pairwise correspondences, we use the vector quantization technique [Mori et al. 2005] which first finds a set of canonical shape contexts, called shapemes, by clustering and then labels each shape context of every shape by its nearest shapeme. The whole shape is thus represented only by a histogram of labels, which is a representation much faster to compare. Even though this is reported to perform well on general datasets, we have determined that the loss of exact spatial and ordering information allows for too much generalization in our settings. Moreover, the results were quite dependent on the particular set of shapemes.

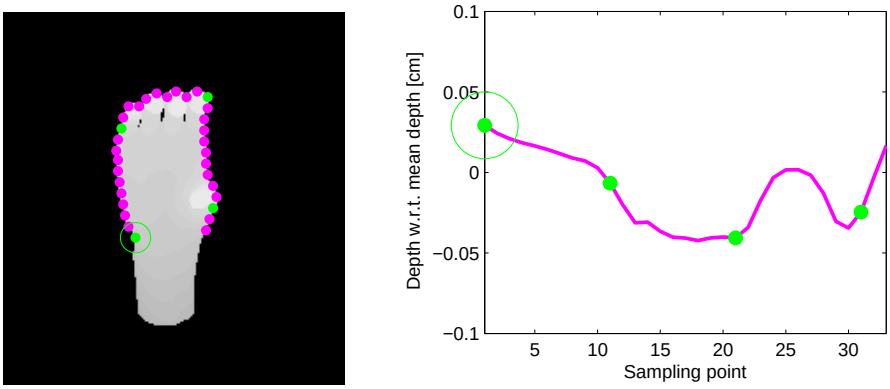
Second, restricting the generalization power inherently leads to sensitivity to artifacts in the contour, especially to changes in the topological structure of the shape. For example, unlike with histogram-based descriptors, merging or splitting two closely adjacent fingers poses a challenge as the contour, i.e., a linear sequence of points, suddenly gets much shorter or longer. Note that this is not only a noise-related issue but also the result of a high instability of the contour with respect to marginal changes in viewpoint or small finger movements. We deal with this unfortunate behavior in three ways. First, by having a reliable segmentation step. Second, by having a dense viewpoint sampling of our database hand shapes, which increases the probability that such a topologically perturbed contour actually matches to a contour of the same hand shape, just from a different viewpoint. Last, we do not match the whole closed contours but rather use algorithms for partial matching, which should prevent the distorted parts of the contour from ruining the matching process

The aim of partial contour matching is to identify parts of two shapes that are similar to each other. In our case, these are the reference hand contour from the database and the contour of the segmented query hand. We considered the IS-Match algorithm [Donoser et al. 2009], the benefit of which lies in local matching and in an accurate representation of the contour geometry during the matching process. IS-Match represents the contour as an ordered sequence of sampled points and the descriptor is based on angles which describe the relative spatial arrangement of these points. Further details are given in Section 4.2.2 where we extend a follow-up paper [Riemenschneider et al. 2010] to exploit 3D information. The IS-Match algorithm evaluates all possible matches for all possible lengths and returns a set of subsequence correspondences between the two contours. However, this set has to be translated into a single distance metric for our task of shape retrieval. The authors propose to compute two quantities for each match: its partiality and dissimilarity. Afterwards, the match minimizing the sum of both metrics is found and the sum is declared to be the distance. Unfortunately, the authors do not provide enough details about the exact computation of the quantities. The minimizing subsequences were either too long or too short in our experiments, depending on whether we normalized the quantities or not, both resulting in a suboptimal retrieval performance. In particular, shorter subsequences have shown to be prone to wrong matching of semantically different parts of the hand's contour.

This experience made us to limit the freedom of the matching as follows. We constrain the general matching problem to a template matching problem of finding a specific sequence of points in a given contour. Specifically, for every image in our database, a single rigid continuous sequence of points $\mathbf{C}_{DB}$ is extracted from the contour of the hand. The sequence



(a) The contour of hand shape B (left, zoom-in) and the depth profile (right).



(b) The contour of hand shape C (left, zoom-in) and the depth profile (right).

Figure 4.4 – An example where a different depth profile of the contour can help to distinguish otherwise relatively similar silhouettes. Each magenta point represents a sample point. The green circle indicates the starting point of the sequence for clarity. Depth images are synthetic.

corresponds to the semantically important part of the hand shape, which is the region of the palm and the fingers, see Figure 4.5 for an example. In the query image, this sequence is then searched for in the closed contour of the segmented hand $\mathbf{C}_Q$. We proceed by explaining the details.

4.2.1 Contour Extraction

First, we represent the shapes by a sequence of points sampled equidistantly from the contour. The sampling step is set to 1 cm. By using a metric distance, our method becomes scale invariant as more or less the same number of points is sampled regardless of the distance of the hand to the camera (for the same hand pose). To find the distance in pixels corresponding to 1 cm, we approximate the segmented hand as a plane of constant depth - see Section 2.3.2 for details. The alternative, measuring the pixel-wise distance locally for each sample point based on its neighborhood patch, was experimentally shown to hamper the accuracy of both algorithms presented in the next two sections.

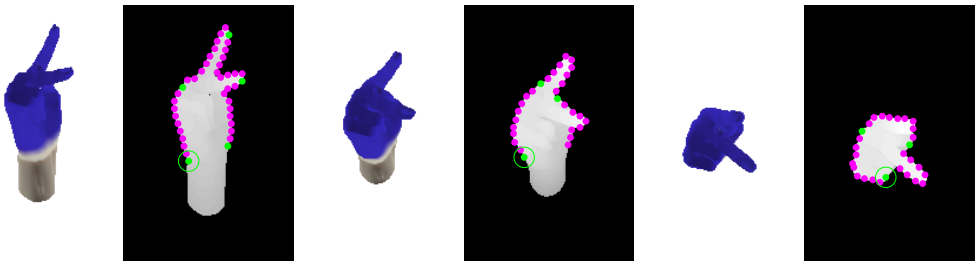


Figure 4.5 – Automatic choice of the contour for generating database sequences. The hand/wrist transition is left out.

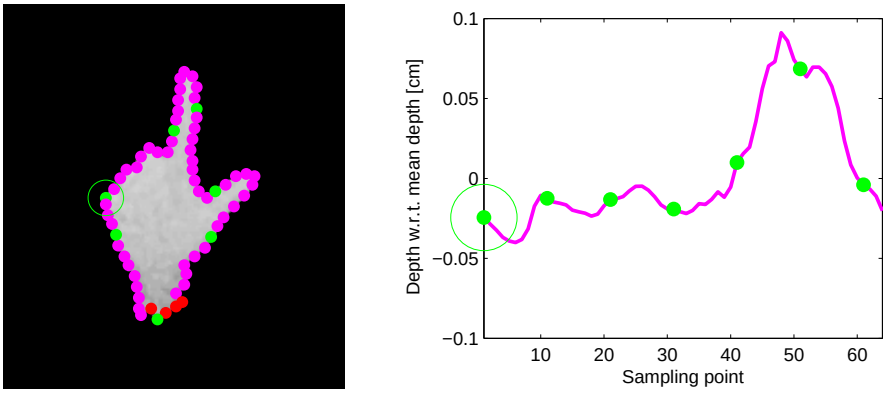


Figure 4.6 – An example of a real image captured by the camera (zoom-in). The red points indicate the likely forearm transition and are to be penalized in the subsequent matching.

The depth of each sampling point is determined as the Gaussian average ($\sigma = 1\,\mathrm{cm}$) of its neighborhood renormalized to ignore any pixels outside the hand. Flying pixels, detected by thresholding the gradient of the whole depth image, are discarded beforehand. See Figure 4.4 for an illustration of contours' depth profiles. Finally, the points are back-projected to their 3D metric coordinates c_i to gain a sequence $\mathbf{C} = (c_i)$.

As already mentioned, the sequences $\mathbf{C}_{DB}$ in our database corresponds to the outer contour of the hand only, see Figure 4.5. We detect the outer contour automatically. As described in Section 5.1, the 3D model used to generate all synthetic images in our database consists of both the hand and a part of the forearm. However, the hand is textured in blue color, which enables us to find the contour of the blue hand and mark its points which border on the skin-colored part of the model. The longest subsequence of the bordering points is then removed. In addition, to successfully handle the rare cases when fingers are in front of the forearm, we do not mark the points if the border is actually a depth edge (specifically, of 7 cm or more).

To further reduce the risk of spurious matches, especially those between the hand and the forearm, we adopt a similar strategy also for the query contour $\mathbf{C}_Q$. We find a subsequence of 5 'unstable' points $\mathbf{F} \subset \mathbf{C}_Q$ (ignoring possible circular shift for the sake of clarity) which is likely to correspond to the transition between the hand and the forearm, see Figure 4.6. Matching with these unstable five points will be penalized (but not impossible) when aligning the contours later. The actual selection is done by computing the gradient at each

point and choosing the subsequence of 5 points minimizing the sum of the gradients as this should be the only part not lying on a depth edge. A similar idea was introduced for hand segmentation in Section 3.2.

4.2.2 Alignment by IS-Match

The task is to align a template sequence of points $\mathbf{C}_{DB} = a_1, \ldots, a_K$ with some subsequence of a query contour $\mathbf{C}_Q = b_1, \ldots, b_L$ and to return their distance d. We assume that $K \leq L$ otherwise we return no match, i.e. infinite distance d. Since the query contour is closed and we do not want to miss matches going over b_L to b_1, we unwind it into a non-closed sequence $\tilde{\mathbf{C}}_Q = b_1, \ldots, b_L, b_1, \ldots, b_L$ of length $2L$.

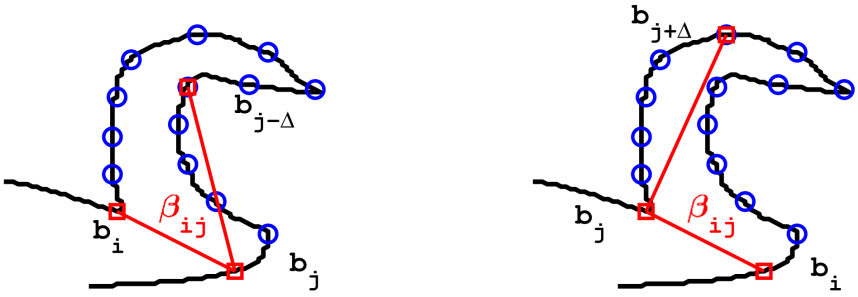
In this section, we utilize the IS-match algorithm for non-closed contours [Riemenschneider et al. 2010] adapted to exploit 3D coordinates of the sampled points. Riemenschneider et al. use a matrix of angles which encode the geometry of the sampled points. Since angles are preserved by a similarity transformation, this descriptor is invariant to translation, rotation and scale. They calculate angles β_{ij} between a line connecting the points b_i and b_j and a line to a third point relative to the position of the previous two points. The angle is defined as

$$\beta_{ij} = \begin{cases} \measuredangle(\overline{b_i b_j}, \overline{b_j b_{j-\Delta}}) & i < j \\ \measuredangle(\overline{b_i b_j}, \overline{b_j b_{j+\Delta}}) & i > j \\ 0 & \text{abs}(i-j) \leq \Delta \end{cases} \tag{4.1}$$

$fi = 5$ is an offset parameter of the descriptor, see Figure 4.7 for an illustration. The third point is always chosen to lie between b_i and b_j. The angles are calculated for every pair of points along the contour $\tilde{\mathbf{C}}_Q$, thus building a matrix $\mathrm{B} = (\beta_{ij})$ of size $2L \times 2L$, see Figure 4.8b. Similarly, we compute the matrix $\mathrm{A} = (\alpha_{ij})$ for $\mathbf{C}_{DB}$.

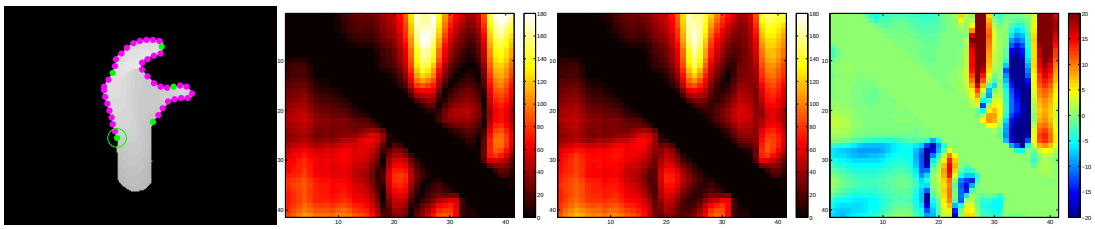
However, while the original paper [Riemenschneider et al. 2010] computes angles in the image plane, we measure the angles in the 3D plane defined by each triplet of selected contour points. The differences in the angles are subtle, usually within 10% of the range, see Figure 4.8d. We also experimented with other extensions, such as statistically summarizing the depth content of each triangle $b_i b_j b_{j \pm \triangle}$ or explicitly representing angles in planes perpendicular to the image plane in addition to the original angles. The variations caused a distinct decrease in the method's accuracy, though.

Having matrices A and B, the optimal alignment can be computed. Due to the selection of the third point to lie between b_i and b_j, any square block in B starting at its main diagonal describes the corresponding subsequence in a self-contained way. Therefore, we extract L square blocks of size $K \times K$ from B starting at B_{ii} for $i = 1, \ldots, L$ and compare each of them to A by mean squared error. In addition, we penalize each comparison by the number of unstable points $\mathbf{F}$ (Section 4.2.1) in the considered subsequence. The minimal error is set as the distance d.



(a) $i < j$ (b) $i > j$

Figure 4.7 – An angle is measured between any two sampled points b_i and b_j. The illustrations are taken from [Riemenschneider et al. 2010] with a slight modification.



(a) The analyzed image with its contour (zoom-in).

(b) The matrix of angles A measured in the image plane.

(c) The matrix of angles A measured in 3D planes.

(d) The difference (b) - (c).

Figure 4.8 – The original and adapted IS-Match algorithm.

4.2.3 Alignment by Procrustes Analysis

As an alternative to IS-Match, we exploit Procrustes analysis to solve the same problem, i.e. to align a template sequence of points $\mathbf{C}_{DB} = a_1, \ldots, a_K$ with some subsequence of query contour $\mathbf{C}_Q = b_1, \ldots, b_L$, $K \leq L$, and to return their distance d. Let us restate that we are working with 3D points.

The constrained orthogonal Procrustes analysis is a method which can be used to determine the optimal translation, uniform scaling and rotation to align two point clouds with S paired points. In the following explanation, let us denote $\mathrm{X}^{3\times\mathrm{S}}$ and $\mathrm{Y}^{3\times\mathrm{S}}$ the two point clouds in their matrix form. First, both point clouds are normalized in translation and scale. The translational component is removed by subtracting the mean of each point cloud, whereas the uniform scaling factor is found as $\sqrt{\mathrm{tr}(\mathrm{X}^T\mathrm{X})}$, equivalently for Y. The rotation matrix can be found as a closed-form solution utilizing the SVD of $\mathrm{X}^T\mathrm{Y} \in \mathbb{R}^{3\times3}$, which can be computed efficiently. The error metric, being the sum of squared distances (SSD), is then computed on the rotated result.

Originally, we were experimenting with the Kabsch algorithm [Kabsch 1978], which solves the same problem but without any normalization in scale. The reasoning was that no scaling is necessary since the contours are located in a metric space, thus of a proper size, and any non-semantic scaling would increase the rate of false matches. Hence, we were

surprised at the fact that normalizing the scale actually increased the retrieval accuracy, especially due to a better separation of open and closed hand shapes.

The algorithm works as follows. We extract L subsequences of length K from $\mathbf{C}_Q$ starting at b_i for $i = 1, \ldots, L$ (circularly shifting the sequence if necessary) and align each of them to $\mathbf{C}_{DB}$ independently. Similarly to IS-Match, we penalize each comparison by the number of unstable points $\mathbf{F}$ (Section 4.2.1) in the considered subsequence. The minimal error is set as the distance d.

Note that as we are basically matching all database template sequences to all L starting points on the query contour, it is worth optimizing the flow. First, we normalize database sequences in scale and location offline. Additionally, we preprocess the query contour sequence by extracting all subsequences of a certain set of lengths, normalizing and storing them in a cache. This way the normalization is computed only once per a specific length of database sequences, regardless the number of such template sequences. We precompute the subsequences for every template length in the database. Furthermore, we discard subsequences having their end points farther than 10 cm apart during the preprocessing. This is valid as all templates are relatively closed in this sense (their end points are only separated by the synthetic forearm) and thus a good match to a line-like subsequence is excluded anyway.

4.3 Ranking Aggregation

In the previous sections, we have presented two different approaches to extract discriminative information from a depth image of a segmented hand, namely the region descriptors and contour-based descriptors. They concentrate on different parts of the image and are sensitive to different artifacts in the image. Therefore, each approach has its own benefits and disadvantages. For instance, the computation of the central point, radius and orientation is required for region descriptors. On the other hand, region descriptors can be fast and easily compared using the L^2 distance, whereas the presented contour-based methods require computationally intensive alignment to try all possible superpositions. While contour-based description is very sensitive to imperfections in the contour, histogram-based descriptions are more sensitive to misplacement of the descriptor due to any suboptimal hand/forearm segmentation.

All this suggests that it might be beneficial to use two different descriptors in parallel to increase the recognition accuracy. We approach this problem by aggregating the results of a pair of descriptors, which are treated as black boxes in this section. However, direct aggregation of distance metrics (for each pair of the query and database images) is problematic since each descriptor generates its distances in a different range, which even might not be linearly scaled. Therefore, we propose to combine the rankings produced by each descriptor after comparing a query to all database images, which are independent of the actual scaling of the distances. Formally, the ranking $\mathbf{R} \in S_n$ is a permutation of n elements, n is the number of images in the database, where R_i is a natural number indicating the rank of database image i, the best match having the first rank. We present two possibilities of aggregation of descriptors A and B:

Average rank is the average of two ranks $\mathbf{R}^{\mathrm{A}}$ and $\mathbf{R}^{\mathrm{B}}$, i.e. $(\mathbf{R}^{\mathrm{A}} + \mathbf{R}^{\mathrm{B}})/2$.

Post-filtered rank is finding the best k matches in $\mathbf{R}^{\mathrm{A}}$, indexed as $\mathcal{K}$, and afterwards comparing the query only to the few database images $\mathcal{K}$ using B (the post-filter) to produce the final ranking. In our experiments, we use $k = 5$. This method is suitable in particular for slow and/or precise descriptors to be used as a post-filter.

The database image ranked as first is then the aggregated solution.

We present the results of ranking aggregation in Section 5.5. The most beneficial combination seems to be the IS Match post-filtered with the circular region descriptor where the recognition quality is improved by 4 percent points. Although this is quite little, the improvement comes computationally nearly for free as the the circular region descriptor shares the segmentation phase with IS Match and its run time for making 5 comparisons is negligible. Moreover, as shown in Section 5.6, its creation time is very fast.

Chapter 5

Experiments

We evaluate the performance of both types of descriptors proposed in Chapter 4 using real recorded videos as queries against a synthetic database. The database and the evaluation methodology is introduced in Sections 5.1 and 5.2. Sections 5.3 to 5.5 contain benchmarks used to support several decisions made during the design of the descriptors. Finally, Section 5.6 contains the general evaluation on our testing datasets.

5.1 A Database for Hand Shape Retrieval

We use a synthetic hand shape database for nearest-neighbor classification of query images. An articulated right hand 3D model was kindly provided by A. Heloir [Heloir and Kipp 2010]. The model is of adult proportions and includes a part of the forearm as well. The forearm is practically ignored in the computation of contour-based descriptors, whereas it is used for creating distribution-based region descriptors.

We have manually modeled 14 out of the 41 frequently used ASL hand shapes [Tennant and Brown 2002]. These are A-I and K-O (note that there is no 'J' hand shape), see Figure 5.1 for frontal renderings. Note that especially the closed-fist hand shapes are rather similar to each other. The technique of introducing small distortions to the joints' angles and positions was not used: while it might decrease the sensitivity to variations in hand shapes, it also makes the database very big.

The hand is located at the origin of the 3D coordinate system and a virtual perspective camera is used to render each hand shape from different viewpoints. The virtual camera orbits the hand with a fixed distance to it and is always pointed towards the center. We define the position of the virtual camera using the spherical coordinates as in Figure 5.2. The viewpoints are sampled equidistantly in the inclination θ ranging between 0 and 90 degrees and in the azimuth φ between 0 and 360 degrees. We use sampling steps of 15 and 30 degrees, depending on the particular experiment. In the following, we use the notation $s(\varphi)$ and $s(\theta)$ to denote the chosen sampling step. With the maximum sampling density, there are 168 viewpoints per pose, totaling 2352 rendered frames for the whole database. Generating multiple rotations of the same image is not necessary as all our descriptors are

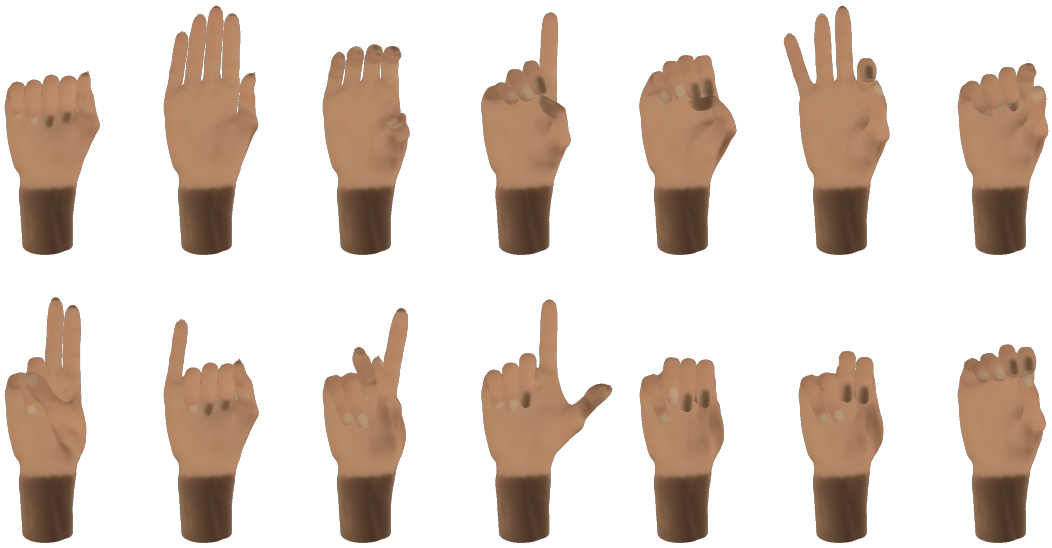


Figure 5.1 – Rendered images of 14 hand shapes contained in the database. Upper row: shapes A-G, lower row: shapes H-I, K-O.

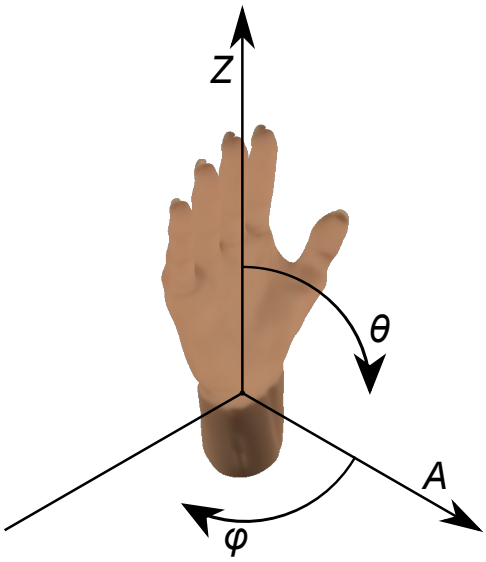


Figure 5.2 – A spherical coordinate system with zenith direction Z and azimuth axis A. The hand in the picture is being observed from inclination $\theta = 45^\circ$ and azimuth $\varphi = 45^\circ$.

rotation invariant. Note that the viewpoint is changed and not the hand pose, i.e. the wrist always stays straight.

We use Blender to render the database. The resolution is the same as of the 3DV camera, i.e. 320×240 pixels. The output is a depth image and a color image with the hand part in blue, see Figure 4.5 on page 35 for an example. The color image is utilized solely in Section 4.2.1, the blue color being an overlaid texture of the model. The values in the depth image are then rescaled so that the synthetic hand measures 18.3 cm, in accord

Figure 5.3 – Several frames of the letter C from Test dataset 1.

with the average human hand length [Agnihotri et al. 2006]. Finally, the descriptors are precomputed for each rendered depth image. The content of the database are the descriptors and not the images themselves.

5.2 Evaluation Methodology

Before we report any evaluation of the descriptors presented in Chapter 4, let us first introduce the general testing procedure, datasets, and metrics.

The input of our system is a depth image query, which is either a frame of a dataset video or, in interactive mode, comes directly from the camera. The hand is segmented and its descriptor extracted. This descriptor is then compared to all database descriptors in a nearest-neighbor fashion[1] and a ranking is produced. Optionally, this ranking may be altered as described in Section 4.3. Finally, the hand shape class of the best match is chosen as the output. The actual best matching viewpoint, i.e., the orientation of the hand, is not used in the evaluation. We stress that the classes correspond to hand shapes and not fingerspelled letters. Specifically, the letters H and U, G and Q, or K and P have the same hand shape but different hand orientation towards the camera, see Figure 2.1 on page 7.

Although there are many elaborate sign language datasets captured using conventional cameras, we are not aware of any dataset created by a ToF camera for this purpose. Therefore, we had to revert to capture our own videos using the 3DV camera. As the author is not a trained signer himself, the recordings show a large variance in hand shapes. All videos capture the same person. All dataset frames are manually annotated by their ground-truth hand shape class.

We have acquired one training dataset and two testing datasets. The training dataset was used to design the descriptors and tune their parameters, while the testing datasets are purely used for assessing the retrieval performance. Note that the classical three-way splitting is superfluous in our context since we do not utilize any machine learning techniques. We now describe the datasets in more detail:

Training dataset is a video of a person performing the fingerspelled letters A-I, K-Q, and U with his right hand in several different locations around the head and with azimuth within ±90 degrees and inclination of 45-90 degrees. The hand distance

[1] We have also experimented with k-nearest neighbors but no consistent increase in recognition performance was observed.

from the body is relatively big not to raise any major segmentation issues. The actual dataset contains 20 equidistantly sampled frames for each letter.

Test dataset 1 is a dataset semantically similar to the training one, albeit coming from a different shooting session and with a small distance between the body and the hand. Figure 5.3 contains 4 frames for illustration. The dataset contains 20 equidistantly sampled frames for each letter as well.

Test dataset 2 is a video of a person performing fingerspelled the letters A-I, K-Q, and U with his right hand near the signer's face and facing the camera, which is a realistic settings for fingerspelling. The actual dataset contains 10 equidistantly sampled frames for each letter.

Typically, we use only the letters A-I and K-O from the datasets for evaluation as they correspond to distinct hand shape classes. The letters P, Q and U are evaluated in Section 5.6.

Our evaluation metric is the retrieval accuracy, measured as the fraction of the correctly classified queries in the set of all queries from a given dataset.

5.3 Descriptor Localization Evaluation

In this section, we evaluate six methods of localizing a descriptor, as presented in Section 4.1.1. To recapitulate, these are: mean, geometric median, MIP, mean-shift with a standard deviation-based bandwidth, mean-shift with a 12 cm bandwidth, and median-shift with a 12 cm bandwidth. Additionally, MIP with its input morphologically closed by a circular structure element of radius of 5 pixels is evaluated.

We argue that the most important asset of a good algorithm for detecting the center point is its robustness to semantically unimportant changes of the hand shape rather than its absolute closeness to one specific semantic point like, e.g., the palm's center. Therefore, we investigate the point's stability with respect to several alternations of the silhouette:

Small out-of-plane rotations in azimuth and inclination. Four views with differences of ± 3 degrees and four with ± 6 degrees were considered. Invariance to these changes is important as the viewpoints in our database are only sparsely sampled.

Morphological opening and closing by a circular structure element of radius of 1 to 4 pixels. This is mainly to simulate joining and separating of close fingers as a result of imperfect segmentation and low resolution of the depth camera.

Changes in the length of a forearm are also very important to be invariant to. We extend or shorten the forearm by 2 and 4 centimeters during a simulated hand segmentation.

Little movements of fingers is significant as one cannot expect a signer to hold a certain hand pose without any shivering going on.

While the first three criteria are evaluated using synthetic images of our hand model (hand shapes A-O without J; azimuth angles sampled every 30 degrees; inclination fixed to 30, 60, and 90 degrees), we use a tailored real video for testing the fourth criterion. In the video, the signer performs C and F fingerspelling signs with a stable hand and just very slight movements of the fingers.

The results can be seen in Figure 5.4. We are interested in the fluctuations of estimated center points in the modified images with respect to the reference image. For the first three criteria, each figure shows the average distance to the position in the reference image. The statistics are aggregated over all poses and viewpoints. As far as the fourth criterion is considered, the distances are shown with respect to the mean position over the whole video and the statistics are per-frame.

Morphological closing reveals increased sensitivity of MIP to this operation while other interest points are nearly not affected. This is due to the fact that the distance transform is very sensitive to any cracks in the silhouette. We can see that this might be improved by closing the shape beforehand, as MIP with closing performs very well. However, this apparently helps only on the synthetic dataset as both versions of MIP perform very poorly in the real video, especially in the case of the letter C (Figure 5.4e). Note that morphological opening does not exhibit any differences, which is mainly due to the fact that the contours are rendered perfectly and the structuring element radius is small enough not to separate any touching fingers.

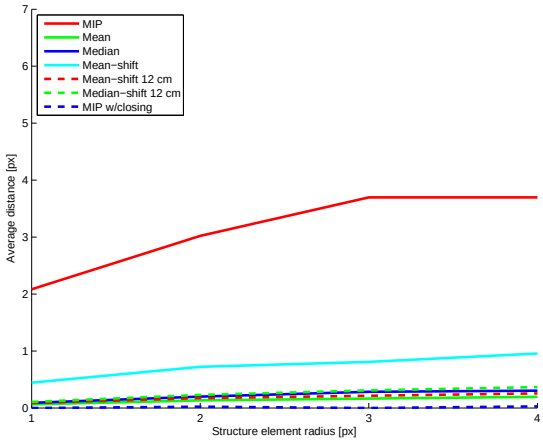
Out-of-plane rotations appear to be problematic for MIP and slightly for the mean-shift based on standard deviation, which has to do with the changing contours of the projections. On the other hand, the MIP excels in different forearm lengths, while the mean and median perform the worst.

We can conclude that if the length of the segmented hand was stable, the mean would be the best choice due to its quality and speed. However, as this might not be the case in our settings, we sacrifice some computational time to compute the median-shift with a 12 cm bandwidth. The algorithm takes about 10 ms on an input depth image of a hand subsampled by a factor of 1.5.

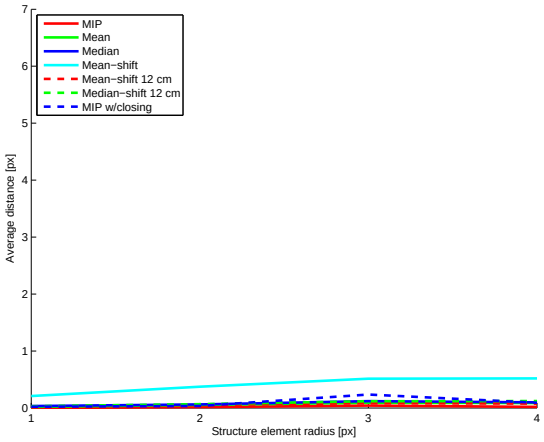
5.4 Cell Configuration and Representation Evaluation

In Section 4.1.2 two layouts of cell configurations of a descriptor were presented: the circular and the grid one. The circular descriptor has two parameters that can be used to vary the complexity of the descriptor: the number of sectors within a ring (angular resolution) and the number of rings (radial resolution). The square grid descriptor has only one parameter, the number of cells. As the complexity of the descriptor grows, it will be able to discriminate better, but it will also be more sensitive to shape distortions. In addition to that, four cell representations were proposed: point count (PC), mean depth difference (MDD), depth histogram (DH) and mean gradient magnitude (MGM).

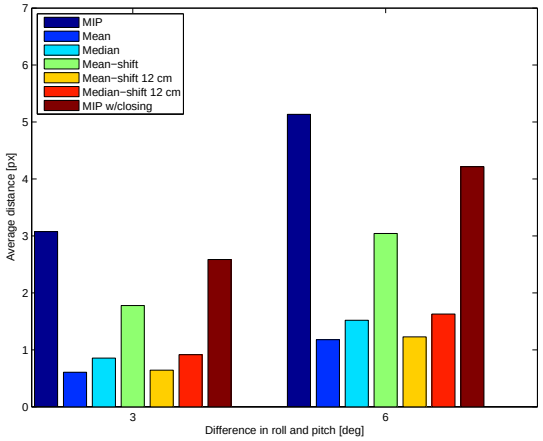
We evaluate the retrieval accuracy of the system using each cell representation in a number of configurations. In the benchmark, the letters A-I and K-O from the training dataset are matched to the equivalent range of hand shapes in the database (azimuth angles sampled



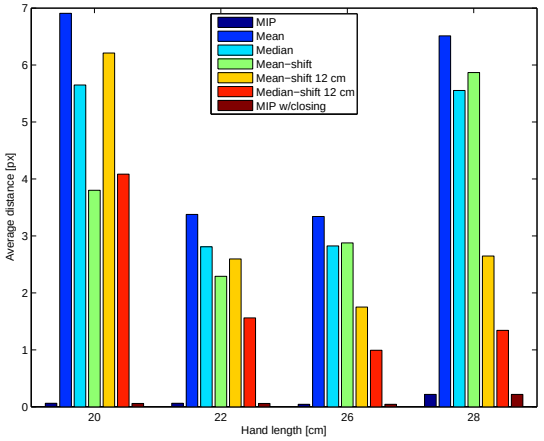
(a) Morphological closing.



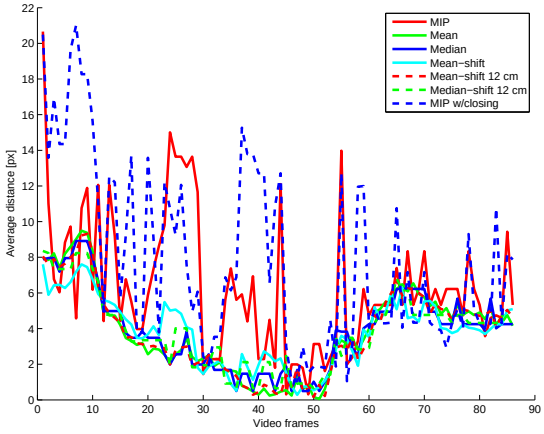
(b) Morphological opening.



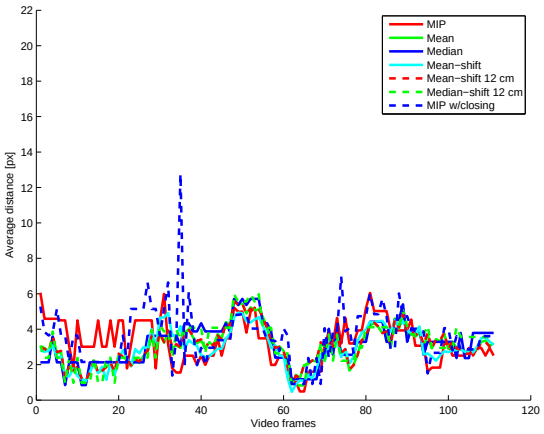
(c) Out-of-plane rotations.



(d) Changes in the length of a forearm.

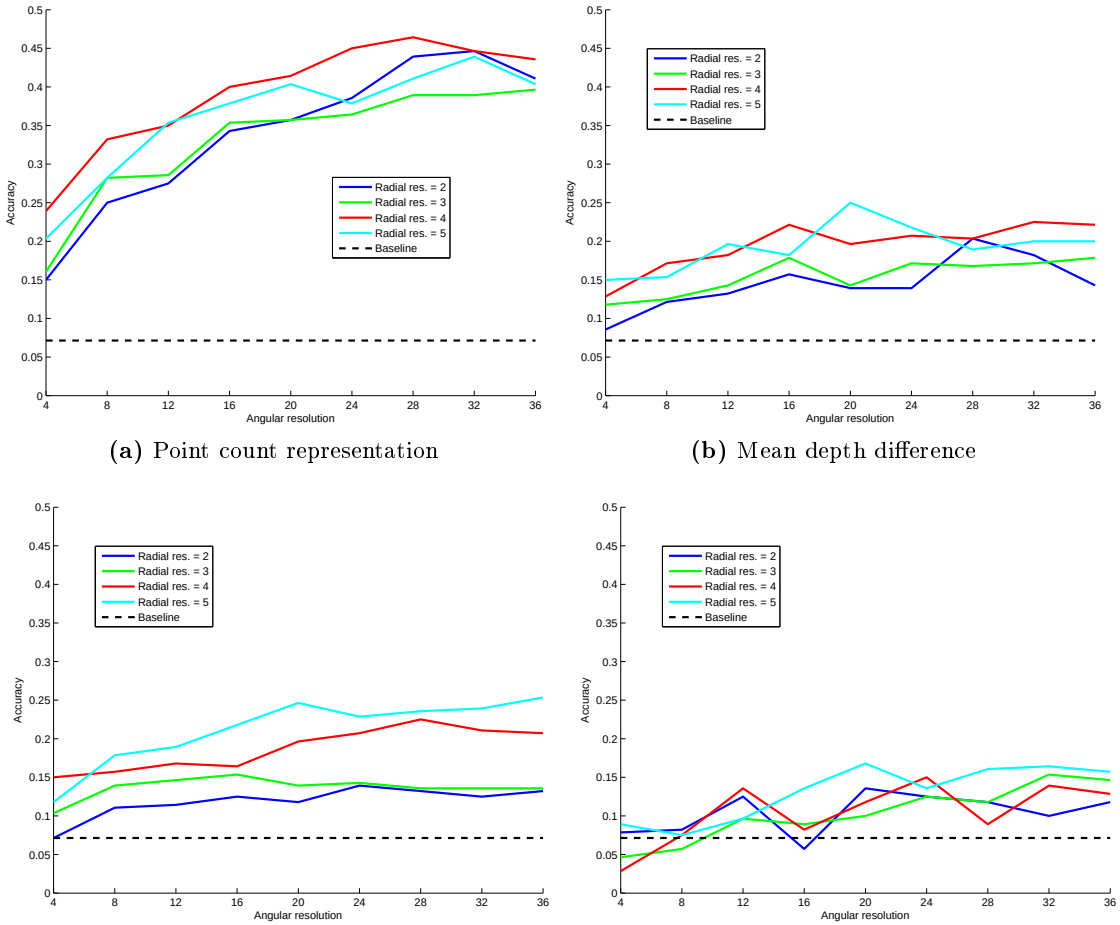


(e) Little movements of fingers, the letter C.



(f) Little movements of fingers, the letter F.

Figure 5.4 – Evaluation of the robustness of different localizations of descriptors.



(a) Point count representation

(b) Mean depth difference

(c) Depth histogram

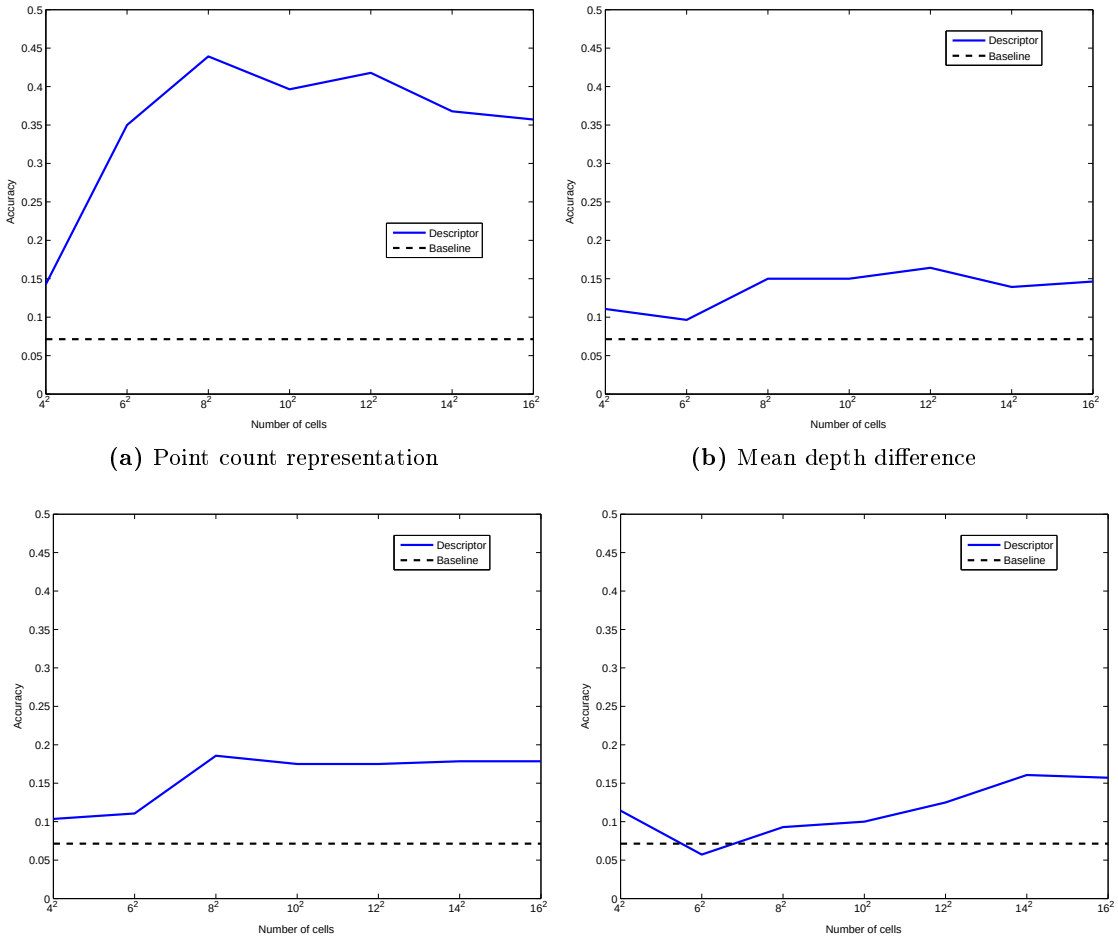
(d) Mean gradient magnitude

Figure 5.5 – Circular design. The baseline is defined as random classification.

every 30 degrees; inclination fixed to 60 and 90 degrees).

The results can be seen in Figures 5.5 and 5.6 and are rather discouraging. Generally, it can be said that using silhouettes is distinctly better than using any depth information alone and that grid configuration does not have an edge over circular one in any test. While the PC representation scores around 40% of accuracy, the depth representations are in the area of 20%. The MGM is the worst representation, likely due to its high noisiness. The DH appear not to be worth the fivefold increase in descriptor size as their performance is only marginally better. In the following, we pick a configuration which performs well both in PC and MDD.

First, let us analyze the performance of circular descriptors. There does not seem to be any distinct peak which would indicate a sweat spot between the discriminative power and sensitivity since the curves become relatively flat after the initial growth in angular resolution. Configurations with 2 or 3 rings seem to perform worse than those with 4 or 5. However, there is an indication that 5 rings might be already too much as this configuration is not the best as far as PC and MDD are considered. Although there is



(a) Point count representation

(b) Mean depth difference

(c) Depth histogram

(d) Mean gradient magnitude

Figure 5.6 – Grid design. The baseline is defined as random classification.

no apparently optimal configuration, we choose a descriptor of size 4×28 = 112 cells. Interestingly enough, the performance is very similar to the 2×28 cell version. However, we believe that its quality in MDD is an outlier by judging the whole curve.

Grid design peaks performance-wise at 64 cells when using PC representation while it stays flat in the other ones. We can conclude that 64 cells is thus the optimal size. Its performance is worse than in the case of the optimal circular one, though.

5.4.1 Joint Representation

We carry out an experiment to determine whether the two best performing representations, namely the point count (PC) and the mean depth difference (MDD), of the 4×28 circular descriptor could be combined to gain retrieval accuracy. We evaluate the aggregation of distance metrics d_{PC} and d_{MDD} using the following functions d_i with a free parameter k:

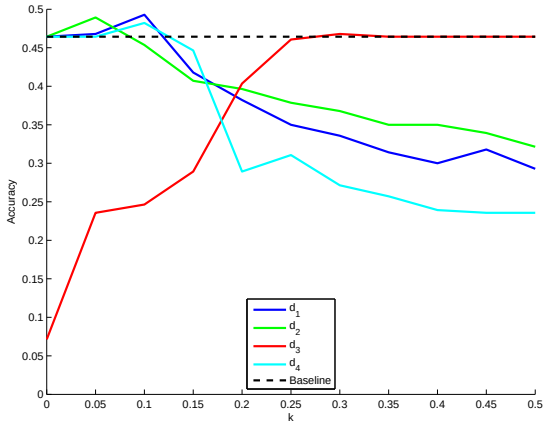


Figure 5.7 – Aggregating distance metrics of the PC and the MDD descriptor representations. The baseline is defined as the accuracy of the PC representation alone.

$$d_1: \quad ||((1-k)d_{PC}, kd_{MDD})|| \tag{5.1}$$
$$d_2: \quad (1-k)d_{PC} + kd_{MDD} \tag{5.2}$$
$$d_3: \quad \min((1-k)d_{PC}, kd_{MDD}) \tag{5.3}$$
$$d_4: \quad \max((1-k)d_{PC}, kd_{MDD}) \tag{5.4}$$

The results are plotted in Figure 5.7 for a certain range of the parameter k. One can see that it is possible to get a marginal improvement this way, the best performance being reached by the weighted L^2 norm d_1 with $k = 0.1$.

5.5 Ranking Aggregation Evaluation

In this section, we explore whether results of different descriptors introduced in Chapter 4 can be aggregated to increase the overall retrieval accuracy, as motivated in Section 4.3. We experiment with 6 variants listed in Section 5.6, using therein introduced abbreviations for clarity. In the benchmark, the letters A-I and K-O from the training dataset are matched to the equivalent range of hand shapes in the database (azimuth angles sampled every 30 degrees; inclination fixed to 60 and 90 degrees).

The results are shown in Table 5.1. Blue values mark combinations which pay off in terms of accuracy. Unfortunately, the data suggest that no distinct improvement can be made. For average rank, combing 2D and 3D versions of the same descriptor seems beneficial, although the highest improvement is only of 3 percentage points. In the case of post-filtered rank, the best improvement of 4 percentage points is attained when post-filtering ISMatch/ISMatch2D with CHist. Additionally, some combinations of 2D and 3D versions of the same descriptor yield a marginal increase as well.

	CHist	CHist2D	ISMatch	ISMatch2D	Procr	Procr2D
CHist	**48 %**	50 %	61 %	58 %	59 %	58 %
CHist2D		**46 %**	57 %	55 %	58 %	52 %
ISMatch			**63 %**	65 %	65 %	63 %
ISMatch2D				**62 %**	68 %	64 %
Procr					**66 %**	69 %
Procr2D						**64 %**

(a) Average rank. The matrix is symmetric.

	CHist	CHist2D	ISMatch	ISMatch2D	Procr	Procr2D
CHist	**48 %**	51 %	58 %	53 %	55 %	53 %
CHist2D	47 %	**46 %**	54 %	51 %	54 %	52 %
ISMatch	67 %	63 %	**63 %**	63 %	66 %	62 %
ISMatch2D	66 %	60 %	64 %	**62 %**	66 %	64 %
Procr	64 %	63 %	65 %	67 %	**66 %**	68 %
Procr2D	61 %	58 %	63 %	61 %	65 %	**64 %**

(b) Post-filtered rank. Rows are original descriptors, columns are post-filters.

Table 5.1 – A comparison of retrieval accuracy of descriptors aggregated using average rank and post-filtered rank. The diagonal (in bold) shows the performance of each method alone, whereas off-diagonal quantities represent aggregations. Blue values indicate the given aggregation to outperform any of the two methods it combines.

5.6 Evaluation on Test Datasets

Here, we utilize both test datasets to benchmark the following 7 descriptors introduced throughout Chapter 4:

CHist The histogram-based region descriptor with circular configuration of size 4×28 cells. The representation is of joint point count and mean depth difference.

CHist2D The same descriptor but without mean depth difference. Hence, no depth information is used.

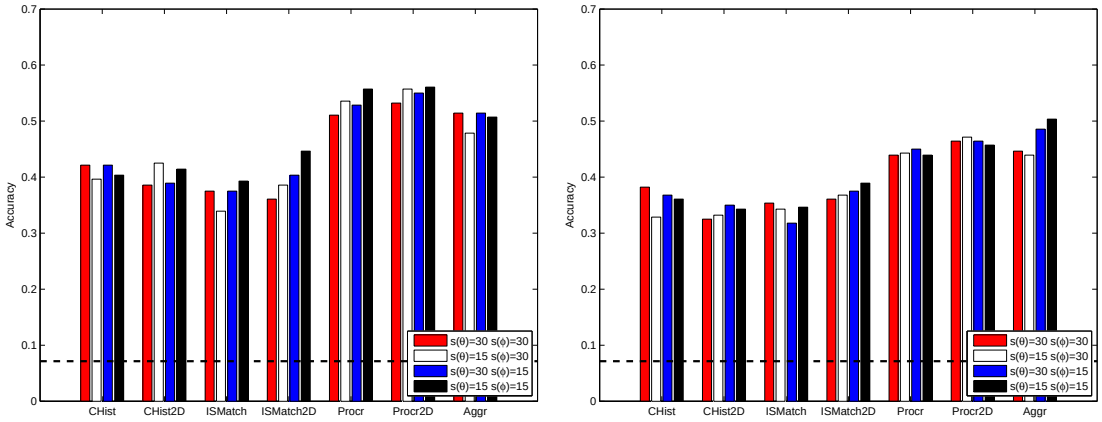
ISMatch The contour-based descriptor using IS-match algorithm for alignment of 3D contours.

ISMatch2D The same descriptor but with constant Z coordinates of the contour points. This represents the original version of [Riemenschneider et al. 2010].

Procr The contour-based descriptor using the Procrustes analysis for alignment of 3D contours.

Procr2D The same descriptor but with constant Z coordinates of the contour points.

Aggr The ISMatch descriptor post-filtered with CHist.



(a) Full range ($\theta \in [0; 90]$, $\varphi \in [0; 360]$) **(b)** Frontal range ($\theta \in [30; 90]$, $\varphi \in [-60; 60]$)

Figure 5.8 – The retrieval accuracy measured on Test dataset 1, the letters A-I, K-O. The plots relate accuracy to different sampling densities of the database, with the function s indicating the sampling step. The horizontal dash line shows the baseline of random classification.

We use two different subsets of the database: a full range of camera angles (inclination $\theta \in [0; 90]$ and azimuth $\varphi \in [0; 360]$) and a frontal range of angles (inclination $\theta \in [30; 90]$ and azimuth $\varphi \in [-60; 60]$), where the palm more or less faces the camera. As the frontal range is sufficient[2] to recognize fingerspelling, we expect Test dataset 2 to perform similarly on both subsets. On the contrary, Test dataset 1 also contains some frames of azimuth larger than 60° and a marginal drop in performance is thus anticipated. Unfortunately, no dataset was acquired to test the recognition in orientations with the palm facing the signer, i.e. azimuths $\varphi \in [90; 270]$. This is a part of the future work.

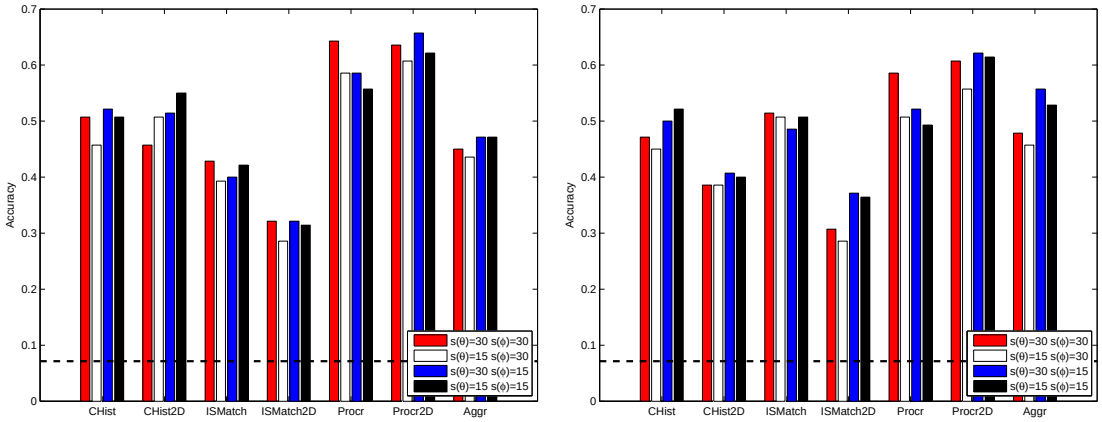
Four different viewpoint sampling densities are tried out in each subset, with the steps of 15° or 30° for both angles independently. Most of the testing is done using the fingerspelling letters A-I and K-O. In the last part, we also investigate the recognition of the 'tilted' letters P, Q, and U.

5.6.1 Retrieval Accuracy

The results are plotted in Figures 5.8 and 5.9. Generally, the best cases score about 60% in accuracy. The Procrustes descriptor performs the best in nearly every test, followed by the circular histogram-based descriptor. Retrieval performance on Test dataset 1 is worse than on Dataset 2.

What is striking is that the sampling density seems to have nearly no influence on the accuracy and it cannot be concluded that higher resolution helps. Despite this, the surge in the number of DB images has an effect on the matching. We have studied the changes in the best matching DB image for each frame of Test dataset 1 and 2 under the full range of angles when changing the step from 30° to 15°. While most of the query frames do not

[2] To be correct, the letter H is fingerspelled with $\varphi = 180°$ but we neglect the fact here and hope that its distinctive shape will save the recognition. The letter G should be recognizable within the constraints.



(a) Full range ($\theta \in [0; 90]$, $\varphi \in [0; 360]$) **(b)** Frontal range ($\theta \in [30; 90]$, $\varphi \in [-60; 60]$)

Figure 5.9 – The retrieval accuracy measured on Test dataset 2, the letters A-I, K-O. The plots relate accuracy to different sampling densities of the database, with the function s indicating the sampling step. The horizontal dash line shows the baseline of random classification.

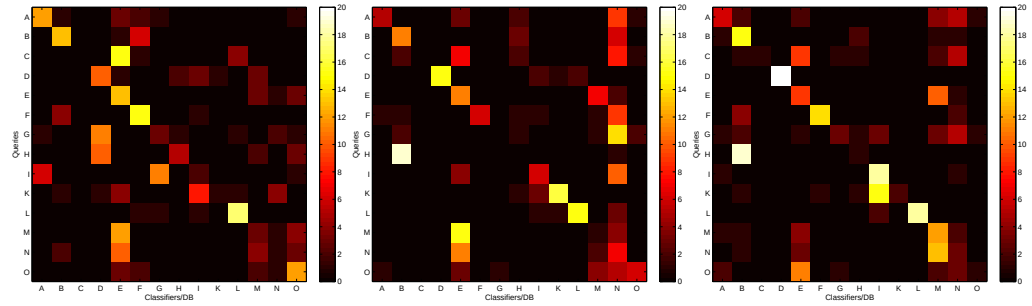
change their classification correctness either way (about 91 %), roughly 23 % change their assigned label. About 50 % of frames change their best matching viewpoint (approximately 25 % in more than 15°). The percentages for the frontal range of angles are slightly lower.

Another aspect is the contribution of depth information. By evaluating each descriptor both with and without depth information, we can conclude that, unfortunately, no variant utilizing the depth performs consistently better than its plain 2D version. Specifically, CHist works better than CHist2D only on the frontal subset, ISMatch is better than ISMatch2D only in Test 2 (distinctly, however) and Procr is marginally worse than Procr2D in every test. This of course neglects the fact that the depth information is used also by the 2D variants in selecting the correct radius (histogram-based descriptors) or the contour sampling step (contour-based descriptors).

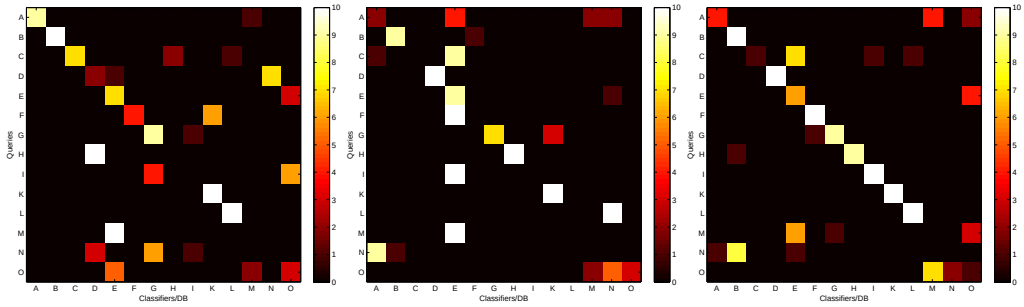
Comparing the performance on full and frontal subsets, we see a small decrease of performance in Test 1, whereas the results in Test 2 depend on the particular descriptor.

The Aggr descriptor is better than its both parts only in Test1. For Test 2, the aggregated descriptor is not worthwhile as CHist performs equivalently or better.

Let us now present the performance of chosen descriptors in more detail. We utilize confusion matrices, where each column represents the instances of the predicted classes and each row of the ground truth classes. In Figure 5.10, confusion matrices of the CHist, ISMatch and Procr descriptors are presented for the full range DB subset in Test 1 and for the frontal range in Test 2, both with 30° sampling. Generally, hand shapes A, E, M, N, and O are all being confused among each other, with slight dominance of one or another depending on the particular case. This suggests that the image quality and resolution of the 3DV camera might not be sufficient to distinguish the shapes. Shape C is nearly always misclassified as E. Interesting is the case of classifying H as B in Figure 5.10a, which is due to the fact that the letter H looks similar to a B-shaped hand from azimuth of 90 degrees, see Figure 5.1 for renderings of the hand shapes.



(a) Test dataset 1, full range ($\theta \in [0; 90]$, $\varphi \in [0; 360]$)



(b) Test dataset 2, frontal range ($\theta \in [30; 90]$, $\varphi \in [-60; 60]$)

Figure 5.10 – Confusion matrices of CHist (left), ISMatch and Procr (right) descriptors.

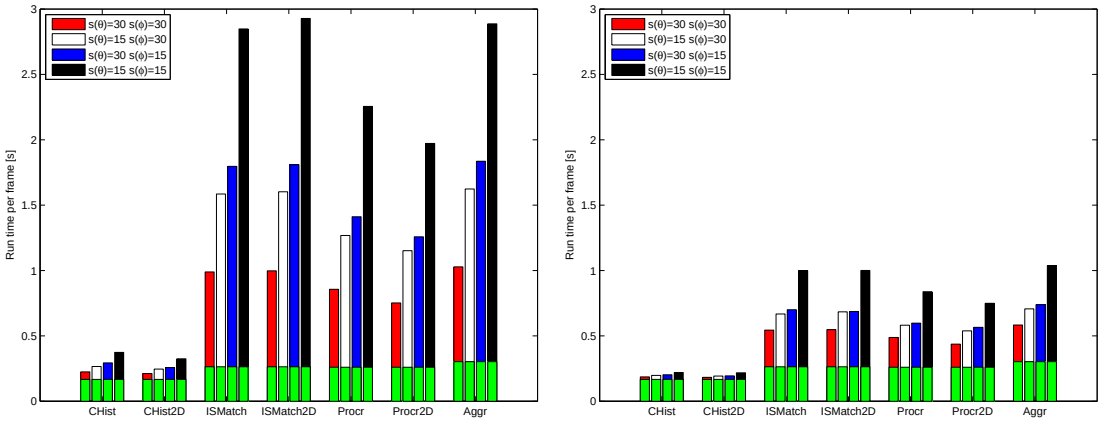
5.6.2 Run time Performance

The implementation was done in Matlab 7.4. Sole exception is the preprocessing stage of Procr/Procr2D, which is implemented as a C wrapper. Figure 5.11 shows the average run time in seconds for the whole retrieval pipeline measured by Matlab's tic-toc on an Intel Core2Duo P8400 laptop with 2 GB RAM. The time required for the creation of a descriptor, including the segmentation, is fixed (roughly 170 ms for CHist and 250 ms for the remaining), whereas the time used for nearest neighbor searching linearly depends on the number of DB elements.

As the benefit of using either the full range subset or dense sampling is not evident, we consider the frontal range angles with the sampling step of 30 degrees, which totals to 36 database images per hand shape. Under this settings, the system is able to run at 5.3 fps with CHist, at 2 fps with Procr, and at 1.7 fps with the ISMatch descriptor. This makes the system interactive. We believe that with a C implementation of both the CHist and the segmentation the system would be able to run in real time.

5.6.3 The Letters P, Q, and U

In all previous evaluations, the letters P, Q, and U were never considered in order to prevent confusion since they classify under the same hand shape as K, G, and H, respectively. Here, we evaluate the accuracy on these three letters exclusively against the whole database of 14 hand shapes. Note that we are using the full range of angles to be able to recognize



(a) Full range ($\theta \in [0; 90]$, $\varphi \in [0; 360]$) **(b)** Frontal range ($\theta \in [30; 90]$, $\varphi \in [-60; 60]$)

Figure 5.11 – The run time per frame, measured on Test dataset 1, the letters A-I, K-O. The plots relate run time to different sampling densities of the database, with the function s indicating the sampling step. The green bars indicate the time spent on segmentation and descriptor computation.

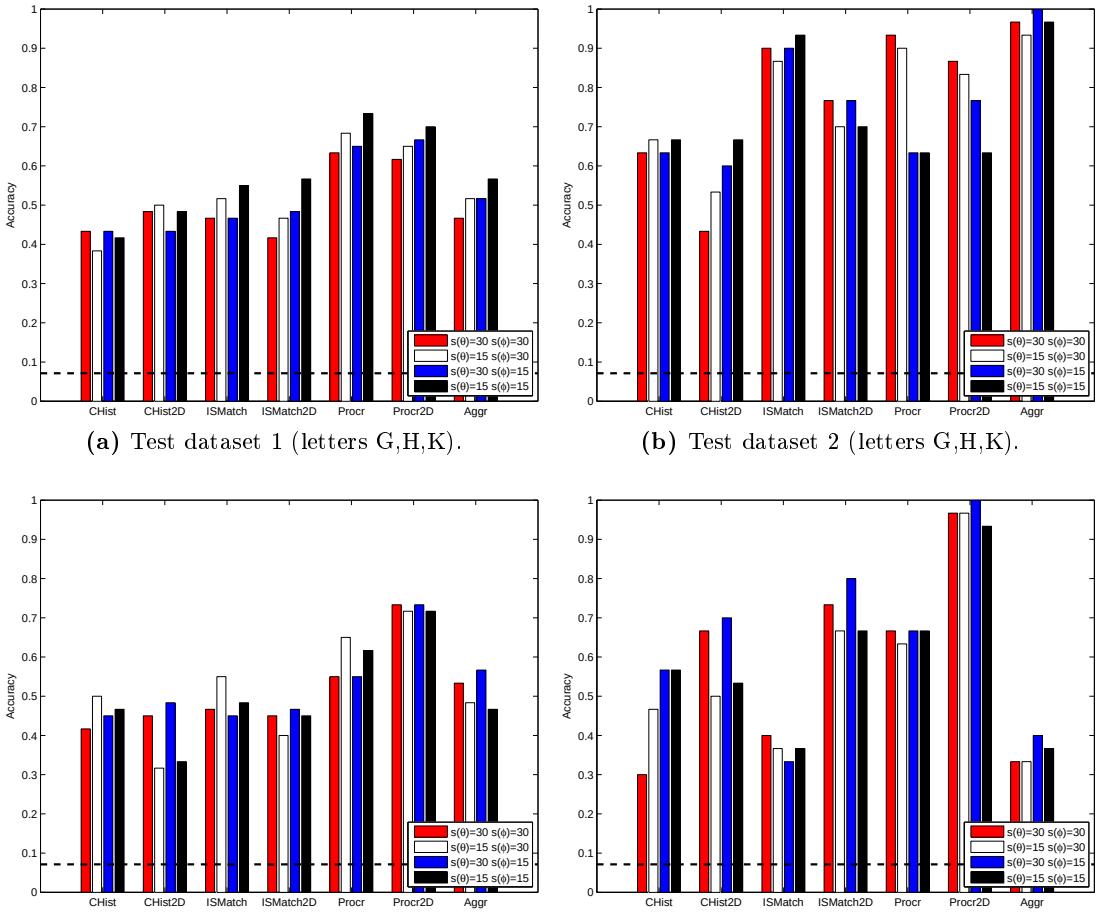
the hand in its minimal inclination. The hand shapes in the database stay the same. The results are plotted in Figures 5.12c and 5.12d in comparison to the evaluation of the 'base' letters G, H, and K. While the results are similar in Test 1, there are some big drops in accuracy in the second dataset. Nevertheless, we conclude that the letters P, Q, and U can be retrieved reasonably well.

5.6.4 Analysis of the Recognition Performance

Let us analyze the discovered inconclusive benefit of dense database sampling first. We argue that this might be partially due to the mostly wrong detection of orientations of the hand. Note that this is possible, since we define the accuracy in terms of hand shape classes rather than in terms of the hand orientation or combination of both (see Section 5.2). Let us assume that the matching process classifies query images independently of their actual hand orientation. Then making the sampling of orientations more dense equivalently for all classes leads to no change in results. We test this conjecture as follows. We make an experiment to see what is the quality of the assigned hand orientations. Using the full range of camera angles, we examined the correct best matches and computed the fraction of frontal angles among them. This should be near 100 % for Test dataset 2 and slightly less for Test 1. As Figure 5.13 shows, the percentage is lower but still far above the baseline of randomness, which suggests that the hypothesis is neither completely wrong nor correct.

The recognition accuracy peaks at 64.3 %, respectively 55.7 % for the two test datasets and thus is far from being excellent. We can see at least two problems which make the recognition perform this way.

First, the closed hand shapes A, E, M, N, O are very hard to distinguish, especially from their frontal views. Their contour is quite similar and the interior of the hand has a very low signal-to-noise ratio when acquired using the ToF camera. Even by directly looking



(a) Test dataset 1 (letters G,H,K). **(b)** Test dataset 2 (letters G,H,K).

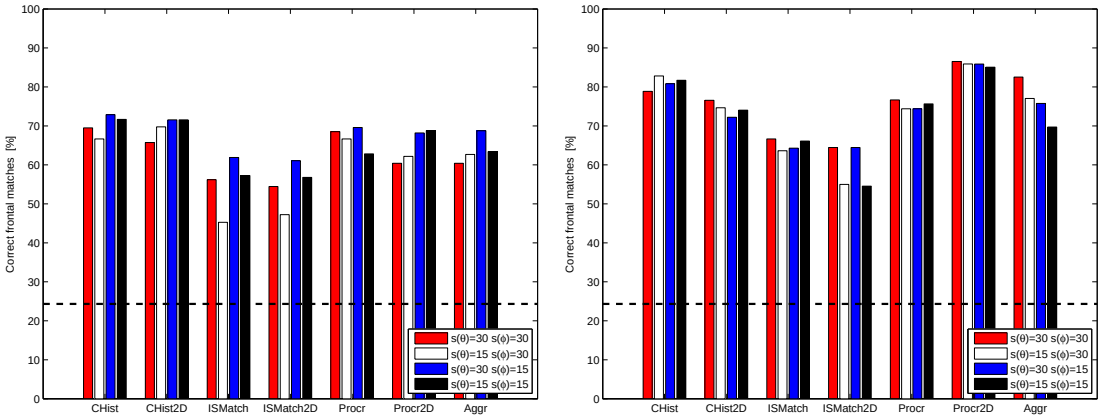
(c) Test dataset 1 (letters P,Q,U). **(d)** Test dataset 2 (letters P,Q,U).

Figure 5.12 – The retrieval accuracy using the full range of angles, i.e. inclination $\theta \in [0; 90]$ and azimuth $\varphi \in [0; 360]$, and a subset of the fingerspelling letters. The plots relate accuracy to different sampling densities of the database, with the function s indicating the sampling step. The horizontal dash line shows the baseline of random classification.

at the depth image with bare eyes, it is often very challenging to guess the position of the fingers in the palm from a single frame.

To verify this assertion, we acquired five videos of a real hand forming the mentioned hand shapes. Each video had 40 frames. The hand was fixed on a box and did not move during or between the acquisitions. For each video k, an averaged depth image $\bar{I}_k$ was computed to represent the clean signal. Afterwards, we extracted the hand interior region common to all five poses using a manually created mask, see Figure 5.14. Each region was normalized to have zero mean depth. Then, for each video k, we considered all 200 frames I as noisy versions of the signal $\bar{I}_k$ and computed their signal-to-noise ratio (SNR):

$$\mathrm{SNR} = 10 \log_{10}\left(\frac{\sum_{mask} \bar{I}_k(x,y)^2}{\sum_{mask} (\bar{I}_k(x,y) - I(x,y))^2}\right) \tag{5.5}$$



(a) Test dataset 1 (letters A-I, K-O) **(b)** Test dataset 2 (letters A-I, K-O)

Figure 5.13 – The fraction of the correct matches being frontal views ($\theta \in [30; 90]$, $\varphi \in [-60; 60]$) in a full range evaluation. The plots relate accuracy to different sampling densities of the database, with the function s indicating the sampling step. The horizontal dash line shows the baseline of random view assignment.

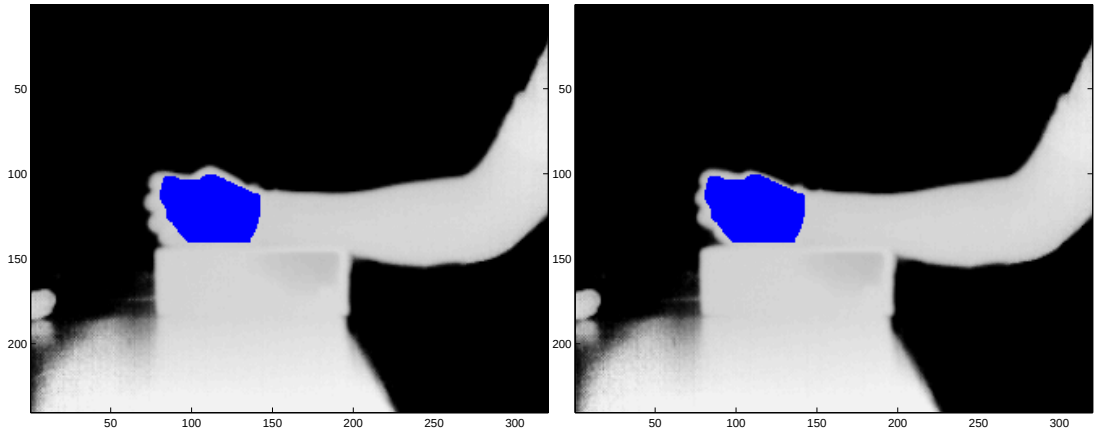


Figure 5.14 – Averaged depth images $\bar{I}_E$ and $\bar{I}_M$ with the highlighted region common to all five poses.

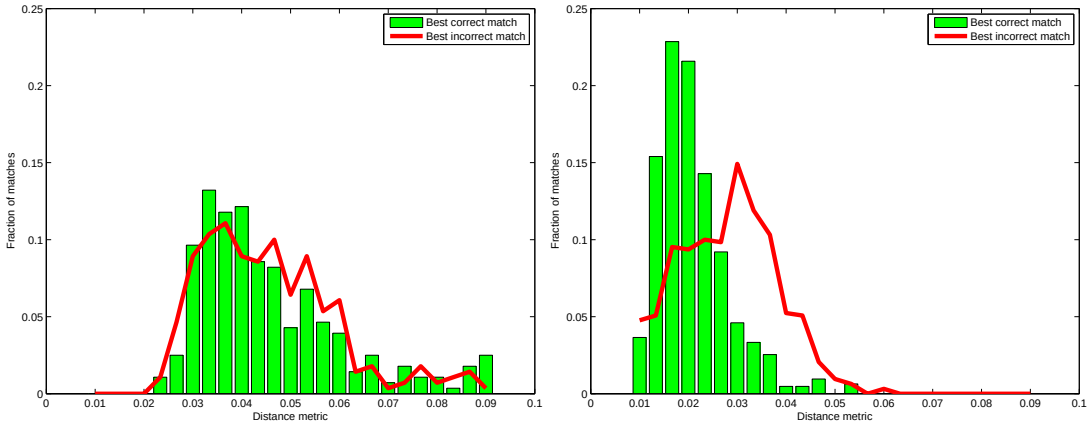
The results are shown in Table 5.2. The diagonal has the highest SNR in each row, which indicates that the hand shapes can be in theory distinguished. However, except for the shape O, all SNRs are negative. This implies that the considered part of the interior of the hand is too flat to stand out of the noise.

The second problem is that intra-class variabilities seem to be comparable or higher than inter-class differences. The are two factors contributing to this, besides the particular design of the descriptor, of course. First, the viewpoint invariance which results in generating many different images for one class. Second, the quality of the acquired test sets is low due to the signer's tendency to form the same gesture each time in slightly different shapes.

We base this belief on the fact that the good and bad matches are very often not distinctly separated in terms of the distance metric and thus it is very easy for bad matches to

[dB]	A	E	M	N	O
$\bar{I}_A$	**-5.25**	-6.66	-6.97	-6.10	-8.59
$\bar{I}_E$	-4.56	**-3.10**	-5.12	-4.25	-6.00
$\bar{I}_M$	-3.97	-4.23	**-2.72**	-3.72	-6.54
$\bar{I}_N$	-5.22	-5.47	-5.86	**-4.39**	-7.09
$\bar{I}_O$	-1.98	-1.49	-2.85	-1.35	**1.39**

Table 5.2 – Signal-to-noise ratios split up by the hand shape. Rows represent five signals introduced in the text. Values in each column are mean SNRs aggregated over 40 frames of the corresponding video.



(a) Test dataset 1 evaluated on the full range database with $s(\theta) = s(\varphi) = 15°$.

(b) The frontal range database evaluated on the full range database, both with $s(\theta) = s(\varphi) = 15°$. Trivial zero-distance matches are discarded.

Figure 5.15 – The histograms of the distances to the nearest correct and incorrect database image for the CHist descriptor.

become the nearest neighbors. We illustrate the behavior on the example of the CHist descriptor, see Figure 5.15. For each query image, the distances to the nearest correct and incorrect database image were found and plotted as a histogram. In Figure 5.15a one can see that the modes of both distributions are relatively equal. An interesting comparison is made in Figure 5.15b where the query was chosen to be synthetic and the bad matches are typically more distant. This suggests that the intra-class variabilities are considerably higher in real-world datasets, both due to camera noise and variabilities in imitating the 'canonical' database hand shapes by the signer.

Chapter 6

Conclusion

We have presented a hand shape recognition system processing depth images from a ToF camera, addressing all steps of the pipeline. The input is classified into one of 14 classes based on a single image of the hand, observed from an arbitrary viewpoint, using appearance-based matching with a database of synthetic views. We have proposed and evaluated three types of descriptors. The evaluation is done on a set of videos of a single person performing fingerspelling in a given range of viewpoints. The system is able to run interactively and its recognition accuracy peaks at about 60 % for the two test datasets.

Different types of features and their representations were investigated, each having its assets and drawbacks. For instance, the computation of the central point, radius and orientation is required for region descriptors. On the other hand, region descriptors can be fast and easily compared, whereas the presented contour-based methods require computationally intensive alignment. While contour-based description is very sensitive to imperfections in the contour, histogram-based descriptions are more sensitive to misplacement of the descriptor. The best recognition results were achieved by a contour-based descriptor where Procrustes analysis is used to search for database templates in a given query contour. Aggregating the ranking of contour- and histogram-based descriptors was shown not to guarantee a stable accuracy improvement.

The retrieval performance is far from being excellent, though. We believe this is partly due to the signer's tendency to form the same gesture each time in slightly different shapes. Additionally, we have shown that several closed hand shapes, differing only by the position of the thumb, are very hard to distinguish using our ToF camera. Also, note that the actual spatial resolution of the hand is rather small: as it is allowed to move freely through the signing space, it covers typically only about 15% of the area of the depth images.

On the other hand, our novel segmentation algorithm produces good results both on the test datasets and in the interactive mode of the system. The system can deal with realistic poses where the hand is quite close to the body. We believe that the idea of using the lateral projection (possibly jointly with the ground projection) for segmentation of 2.5D data might be applicable to other context. Moreover, a C implementation should be able achieve very low run times.

Several insights and observations were brought up during the design as well as during the

evaluation. The biggest is the fact that no consistent improvement was made by using depth information as a discriminative factor in the way presented. This suggests that the chosen depth representation might not be appropriate for this task or ToF depth data in general. However, note that similar behavior is reported by [Kollorz et al. 2008] where integrating depth features raised their reported accuracy by only 1.47 percentage points.

Last, we hope that our evaluation of descriptor localization might be useful for future research.

Future Work

There are multiple ways of extending this thesis. First, it would be very helpful to acquire an annotated database of depth videos of experienced signers and make it publicly available. This would make our experiments more conclusive and would give birth to a standardized benchmark, which is currently not available to our best knowledge.

Second, the recognition could be made signer-independent. One of the easiest ways is to introduce a bootstrapping hand configuration, based on measuring which the proper scaling coefficients between the real hand and the database would be estimated.

Third, investigating a joint recognition of hand orientation and hand shape might be beneficial. It could both prune the search and increase the retrieval accuracy. In the current system, retrieving orientation is rather inexact.

Fourth, the benefit of fusing depth data with additional color information would be helpful to explore, especially for the descriptors.

Furthermore, different depth cameras should be tried out in this context. Especially experiments with Microsoft Kinect could be promising. However, our descriptors are probably not directly transferable to this data due to the different noise model and image acquisition artifacts.

Additionally, one might concentrate their effort on removing the need to test against all database images for every query. This might be done by pruning the search or using techniques such as hashing.

Finally, utilizing machine learning and dimensionality reduction techniques should be investigated. Again, this would benefit from having a large dataset beforehand.

Bibliography

A. K. Agnihotri, B. Purwar, N. Jeebun, and S. Agnihotri. Determination of sex by hand dimensions. *The Internet Journal of Forensic Science*, 1(2), 2006.

Y.-K. Ahn, Y.-C. Park, K.-S. Choi, W.-C. Park, H.-M. Seo, and K.-M. Jung. 3D spatial touch system based on time-of-flight camera. *WSEAS Transactions on Information Science and Applications*, 6:1433–1442, September 2009.

A. Al-Hamadi, O. Rashid, and B. Michaelis. Posture recognition using combined statistical and geometrical feature vectors based on SVM. *International Journal of Information and Mathematical Sciences*, 6(1):7–14, 2010.

V. Athitsos and S. Sclaroff. An appearance-based framework for 3D hand shape classification and camera viewpoint estimation. In *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, FGR '02, pages 45–, 2002.

I. B. Ayed and A. Mitiche. A partition constrained minimization scheme for efficient multiphase level set image segmentation. In *IEEE International Conference on Image Processing*, pages 1641–1644, 2006.

I. B. Ayed and A. Mitiche. A region merging prior for variational level set image segmentation. In *IEEE Transactions on Image Processing*, volume 17, pages 2301–2313, 2008.

N. Bayramoglu and A. Alatan. Shape index SIFT: Range image recognition using local features. In *Proceedings of the 2010 20th International Conference on Pattern Recognition*, ICPR '10, pages 352–355, 2010.

S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:509–522, April 2002.

I. Ben Ayed, A. Mitiche, and Z. Belhadj. Polarimetric image segmentation via maximum-likelihood approximation and efficient multiphase level-sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1493–1500, September 2006.

X. Bresson. A short guide on a fast global minimization algorithm for active contour models. Energy, 2009.

X. Bresson, S. Esedoglu, P. Vandergheynst, J.-P. Thiran, and S. Osher. Fast global minimization of the active contour/snake model. *Journal of Mathematical Imaging and Vision*, 28:151–167, June 2007.

T. F. Chan and L. A. Vese. Active contours without edges. *IEEE Transactions on In Image Processing*, 10(2):266–277, 2001.

M. Donoser, H. Riemenschneider, and H. Bischof. Efficient partial shape matching of outer contours. In *Proceedings of Asian Conference on Computer Vision*, pages 281–292, 2009.

A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly. Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding*, 108:52–73, October 2007.

V. Ganapathi, C. Plagemann, S. Thrun, and D. Koller. Real time motion capture using a single time-of-flight camera. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2010.

S. A. Guomundsson, J. R. Sveinsson, M. Pardas, H. Aanaes, and R. Larsen. Model-based hand gesture tracking in ToF image sequences. In *Proceedings of the 6th international conference on Articulated motion and deformable objects*, pages 118–127, 2010.

N. Habili, C. Lim, and A. Moini. Segmentation of the face and hands in sign language video sequences using color and motion cues. *IEEE Transactions on Circuits and Systems for Video Technology*, 14:1086–1097, 2004.

N. Haubner, U. Schwanecke, and R. Dörner. Recognition of dynamic hand gestures with time-of-flight cameras. Technical report, ITG/GI Workshop on Self-Integrating Systems for Better Living Environments, 2010.

A. Heloir and M. Kipp. Realtime animation of interactive agents: Specification and realization. *Applied Artificial Intelligence*, 24(6):510–529, 2010.

M. B. Holte, T. B. Moeslund, and P. Fihl. View invariant gesture recognition using the CSEM SwissRanger SR-2 camera. *International Journal of Intelligent Systems Technologies and Applications*, 5:295–303, November 2008.

M. J. Jones and J. M. Rehg. Statistical color models with application to skin detection. *International Journal of Computer Vision*, 46:81–96, January 2002. ISSN 0920-5691.

W. Kabsch. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 34(5):827–828, Sep 1978.

M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

E. Klima and U. Bellugi. *The Signs of Language*. Harvard University Press, 1979.

S. Knoop, S. Vacek, and R. Dillmann. Fusion of 2D and 3D sensor data for articulated body tracking. *Robotics and Autonomous Systems*, 57(3):321–329, 2009.

H. Koike, Y. Sato, and Y. Kobayashi. Integrating paper and digital information on EnhancedDesk: a method for realtime finger tracking on an augmented desk system. *ACM Transactions on Computer-Human Interaction*, 8:307–322, December 2001.

A. Kolb, E. Barth, R. Koch, and R. Larsen. Time-of-flight sensors in computer graphics. In *Proceedings of Eurographics 2009 - State of the Art Reports*, 2009.

E. Kollorz, J. Penne, J. Hornegger, and A. Barke. Gesture recognition with a time-of-flight camera. *International Journal of Intelligent Systems Technologies and Applications*, 5:334–343, November 2008.

D.-J. Kroon. Snake: Active contour. `http://www.mathworks.com/matlabcentral/fileexchange/28149-snake-active-contour`, 2010.

R. Lockton and A. W. Fitzgibbon. Real-time gesture recognition using deterministic boosting. In *Proceedings of British Machine Vision Conference*, 2002.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, November 2004.

S. Malassiotis and M. Strintzis. Real-time hand posture recognition using range data. *Image and Vision Computing*, 26(7):1027–1037, 2008.

J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767, 2004.

G. Mori, S. Belongie, and J. Malik. Efficient shape matching using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1832–1837, 2005.

D. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 42(5):577–685, 1989.

E.-J. Ong and R. Bowden. A boosted classifier tree for hand shape detection. In *Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, FGR' 04, pages 889–894, Washington, DC, USA, 2004. IEEE Computer Society.

S. C. W. Ong and S. Ranganath. Automatic sign language analysis: A survey and the future beyond lexical meaning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27: 873–891, June 2005.

N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

J. Piovano, M. Rousson, and T. Papadopoulo. Efficient segmentation of piecewise smooth images. In *Scale Space and Variational Methods in Computer Vision*, pages 709–720. Springer-Verlag, 2007.

H. Riemenschneider, M. Donoser, and H. Bischof. Using partial edge contour matches for efficient object category localization. In *Proceedings of the 11th European Conference on Computer vision: Part V*, ECCV'10, pages 29–42, 2010.

P. M. Roth and M. Winter. Survey of appearance-based methods for object recognition. Technical report, Institute for Computer Graphics and Vision, Graz University of Technology, Austria, 2008.

S. Schuon, C. Theobalt, J. Davis, and S. Thrun. High-quality scanning using time-of-flight depth superresolution. In *CVPR Workshop on Time-of-Flight Computer Vision 2008*, pages 1–7, 2008.

L. Schwarz, D. Mateus, V. Castaneda, and N. Navab. Manifold learning for ToF-based human body tracking and activity recognition. In *Proceedings of the British Machine Vision Conference*, pages 80.1–11, 2010.

P. Smith, N. da Vitoria Lobo, and M. Shah. Resolving hand over face occlusion. *Image and Vision Computing*, 25:1432–1448, September 2007.

S. Soutschek, J. Penne, J. Hornegger, and J. Kornhuber. 3-D gesture-based scene navigation in medical imaging applications using time-of-flight cameras. In *Computer Vision and Pattern Recognition Workshop*, pages 1–6, 2008.

B. Taati, M. Bondy, P. Jasiobedzki, and M. Greenspan. Variable dimensional local shape descriptors for object recognition in range data. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007.

J. W. H. Tangelder and R. C. Veltkamp. A survey of content based 3D shape retrieval methods. In *Proceedings of the Shape Modeling International 2004*, pages 145–156. IEEE Computer Society, 2004.

G. A. ten Holt, J. Arendsen, H. de Ridder, A. J. Koenderink-van Doorn, M. J. T. Reinders, and E. A. Hendriks. Sign language perception research for improving automatic sign language recognition. In *Human Vision and Electronic Imaging XIV*, 2009a.

G. A. ten Holt, M. J. T. Reinders, E. A. Hendriks, H. de Ridder, and A. J. van Doorn. Influence of handshape information on automatic sign language recognition. In *8th International Workshop on Gesture in Human-Computer Interaction and Simulation*, pages 301–312, 2009b.

R. A. Tennant and M. G. Brown. *The American Sign Language handshape starter: A beginner's guide*. Gallaudet University Press, 2002.

J. Triesch and C. von der Malsburg. Classification of hand postures against complex backgrounds using elastic graph matching. *Image and Vision Computing*, 20(13-14):937 – 943, 2002.

A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`, 2008.

W. Vicars. ASL university. `http://lifeprint.com`.

T. wai Rachel Lo and J. P. Siebert. SIFT keypoint descriptors for range image analysis. In *Annals of the British Machine Vision Association*, volume 3, 2008.

R. Y. Wang and J. Popović. Real-time hand-tracking with a color glove. *ACM Transactions on Graphics*, 28:63:1–63:8, July 2009.

E. Weiszfeld. Sur le point pour lequel la somme des distances de n points donnes est minimum. *Tohoku Mathematical Journal*, 43:355–386, 1937.

A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys*, 38, December 2006.

X. Zabuli, H. Baltzakis, and A. Argyros. Vision-based hand gesture recognition for human-computer interaction. In *The Universal Access Handbook*, pages 1–30. Lawrence Erlbaum Associates, Inc., 2009.