Neural Learning Methods for Human-Computer Interaction

Thomas Kopinski

NNT : 2016SACLY002

THESE DE DOCTORAT
DE
L'UNIVERSITE PARIS-SACLAY
PREPAREE A
L'ENSTA PARISTECH

ECOLE DOCTORALE N° 573
INTERFACES : Approches Interdisciplinaires : Fondements, Applications et Innovation

Spécialité : Informatique

Par

M. Thomas Kopinski

Neural Learning Methods for Human-Machine Interaction

Thèse présentée et soutenue à Palaiseau, le 1er février 2016 :

Composition du Jury :

M. Filliat, David	Professeur, ENSTA ParisTech	Président
M. Glasmachers, Tobias	Professeur, Ruhr-Universität Bochum	Rapporteur
M. Zöllner, Marius J.	Professeur, Karlsruher Institut für Technologie	Rapporteur
M. Handmann, Uwe	Professeur, Hochschule Ruhr West	Examinateur
M. Gepperth, Alexander	Maître de conférence, ENSTA ParisTech	Directeur de thèse
M. Geisler, Stefan	Professeur, Hochschule Ruhr West	Co-directeur de thèse

université PARIS-SACLAY

ÉCOLE DOCTORALE
INTERFACES
Approches Interdisciplinaires :
Fondements, Applications et Innovation

Titre: Méthods d'apprentissage automatiques pour l'interaction homme-machine

Mots clés: Gestes sans contact, Apprentissage automatique, Interactions homme-machine

Résumé: Des systèmes à commande tactile ont étés introduits dans le marché de masse avec l'advent des tablettes et des smartphones. L'interaction avec de tels dispositifs a ouvert tout un ensemble de nouvelles possibilités de développer des logiciels et des applications créatives. Il semble raisonnable d'en supposer de même pour les gestes de main, mais les contraintes imposés par la nécessité de traiter des données en temps réel sont beaucoup plus fortes. Cette thèse a pour but de mettre au point un système de reconnaissance de gestes pour l'interaction homme-machine en s'appuyant sur des capteurs du type 'time-of-flight' (ToF). Cette approche a l'avantage d'être applicable dans n'importe quel contexte, indépendamment des conditions d'éclairage ou du soleil (qui pose de graves problèmes aux capteur du genre Kinect). En plus, des capteurs ToF deviennent de plus en plus petits et s'approchent également à des capteurs normaux en termes de coût. Afin de combattre le bruit associé avec des capteurs ToF (qui peut être considérable), une combinaison de techniques efficaces a été développée dans cette thèse et implanté dans un système de démonstration qui est capable de reconnaître les gestes et qui tourne en temps réel. Plusieurs études ont été conduites pour s'assurer que ce système est bien accepté par des utilisateurs humains.

Le système développé dans cette thèse tourne en temps réel dans des conditions difficiles en s'appuyant sur des données qui proviennent d'un seul capteur ToF. Une précision jusqu'à 97% a été obtenue pour 4 gestes dynamiques et 20 gestes statiques en provenance de 20 personnes différentes.

université PARIS-SACLAY

ÉCOLE DOCTORALE
INTERFACES
Approches Interdisciplinaires :
Fondements, Applications et Innovation

Title: Neural Learning Methods for Human-Machine Interaction

Keywords: Freehand Gestures, Machine Learning, Human-Machine Interaction

Abstract: Touch-controlled systems have been introduced to our everyday lives with the launch of smartphones and tablet PCs on the mass-market. The seamless and intuitive way of interacting with these devices opened up a whole new range of possibilities for the development of software and creative applications. It seems reasonable to assume the same for in-air gestures, however the constraints posed by developing interfaces and reacting to human input in real-time in 3D are a different story. This thesis is devoted to developing an in-air hand gesture recognition system for Human-Machine Interaction based on data obtained from a time-of-flight camera. This has the advantage of being able to implement the system in any scenario, be it in- or outdoors since comparable approaches usually suffer from difficult illumination conditions. Moreover, these cameras are steadily becoming smaller in dimension and comparable to off-the-shelf sensors with respect to their price. In order to deal with potentially high noise coming from these devices, high data dimensionality as well as the intrinsic complexity of in-air gesture recognition, a combination of efficient feature computation methods and problem-adapted Machine Learning algorithms has been developed in this thesis and implemented in a real-time demonstration system. In order to not lose sight of one of the most important factors, namely the usability of the system, this works aims to evaluate the functionality of a hand gesture recognition system within a car environment by user studies.

The system developed in this thesis performs in real-time in an in-vehicle scenario under difficult, changing illumination by utilizing data coming from a singe depth camera. It achieves recognition rates of up to 97% for static and dynamic hand gestures on a complex gesture set of ten static and four dynamic hand poses tested on 20 different persons.

to Sanna -
my motivation to get stuff done

Acknowledgements

Too many contributions have been made for the completion of this thesis, it deems appropriate mentioning them here.
Alex, thanks so much for your enthusiasm, recursively iterating through my being in a seamless way, making everything look so easy. Your attitude, expertise and humor made these three years pass by nearly too quickly. Here's to hoping for more to come!

I would like to thank Stefan Geisler for his constant support and uplifting spirit, steadily having an open ear for any questions and showing an exemplary joy towards the subject.

Furthermore, thank you Uwe Handmann for opening all these opportunities without hesitating, having your hand over this project all this time - it's in your name, after all. Gan bei! 干杯

I am also thankful for all the nice colleagues at the HRW and ENSTA. I do not take this for granted.

I'd also like to thank my family and friends for their support, most of which I want to address to my wife. Thank you Sanna, for supporting every crazy idea I have.

Contents

List of Figures

Abbreviations

CWIM Continuous Wave Intensity Modulation

DL Deep Learning

ESF Ensemble of Shape Features

FPFH Fast Point Feature Histogram

FSM Finite State Machine

HGR Hand Gesture Recognition

HMI Human-Machine Interaction

HMM Hidden Markov Model

ML Machine Learning

MLP Multilayer Perceptron

MSE Mean Squared Error

NN Neural Network

NUI Natural User Interface

PC Point Cloud

PCA Principal Component Analysis

PFH Point Feature Histogram

ROI Region of Interest

RPFH Randomized Point Feature Histogram

SBI Suppression of Background Illumination

ToF Time-of-Flight

VFH Viewpoint Feature Histogram

VOI Volume of Interest

Introduction

This work is devoted to improve on the task of human hand gesture recognition[1] (HGR), i.e. interpreting the 'meaning' of a motion performed by humans with their hand. It is a mainly unsolved problem insofar as many approaches are limited by either the range of applicability, the number of interpretable hand gestures or simply a cost-efficient setup itself. Within the scope of this thesis, a contactless camera-based approach is presented showing the advantages of employing Machine Learning (ML) algorithms for this task, in order to be able to effectively integrate an HGR system into an automotive setting with respect to the relevant Human-Machine Interaction (HMI) questions. An HGR system would provide a complementary means of interaction to already present technologies in the car, assisting the driver in various tasks, optimally reducing the cognitive load and decreasing driver distraction.

One crucial factor, when developing such an application, is the desire to present a non-intrusive method, i.e. one that does not force the user to wear additional hardware as data gloves or even hold a device such as an accelerometer as, amongst many other reasons, the former would influence the movement range and the latter would not allow for full freedom of expression for the user's hand. This mainly reduces the possibilities to different sensor technologies, raising various problems as well as offering a lot of advantages. Furthermore, it is desirable to have a hand gesture recognition system detached from the formulation of a complex hand model, hence the underlying work presents a data-driven approach allowing to recognize human hand gestures from their appearance. The system developed in this work aims at detecting hand gestures in an in-vehicle scenario as depicted in Figure 1. A time-of-flight (ToF) camera is employed, recording a volume of interest (VOI) in the nearby user environment. Depth cameras in general, and ToF cameras in particular, allow for easy depth segmentation through simple thresholding. Thus, once the user enters the sensitive VOI, the system is presented a depth image, also termed point cloud, of the hand structure, containing the relevant x-y-z information. Since it cannot be assured that unnecessary information is contained in this point cloud, mostly irrelevant arm parts, a cropping module analyzes the principle components and cuts the point cloud up to the wrist.

ToF cameras have three main advantages: They are reasonably small, retrieve data at high frame rates and are able to work under challenging illumination influences.

[1] Within the scope of this thesis, the term *(hand) gestures* refers to gestures performed in three-dimensional space to avoid confusion with the generally known term of (touch) gestures. Other commonly used terms include in-air, mid-air or freehand gestures.

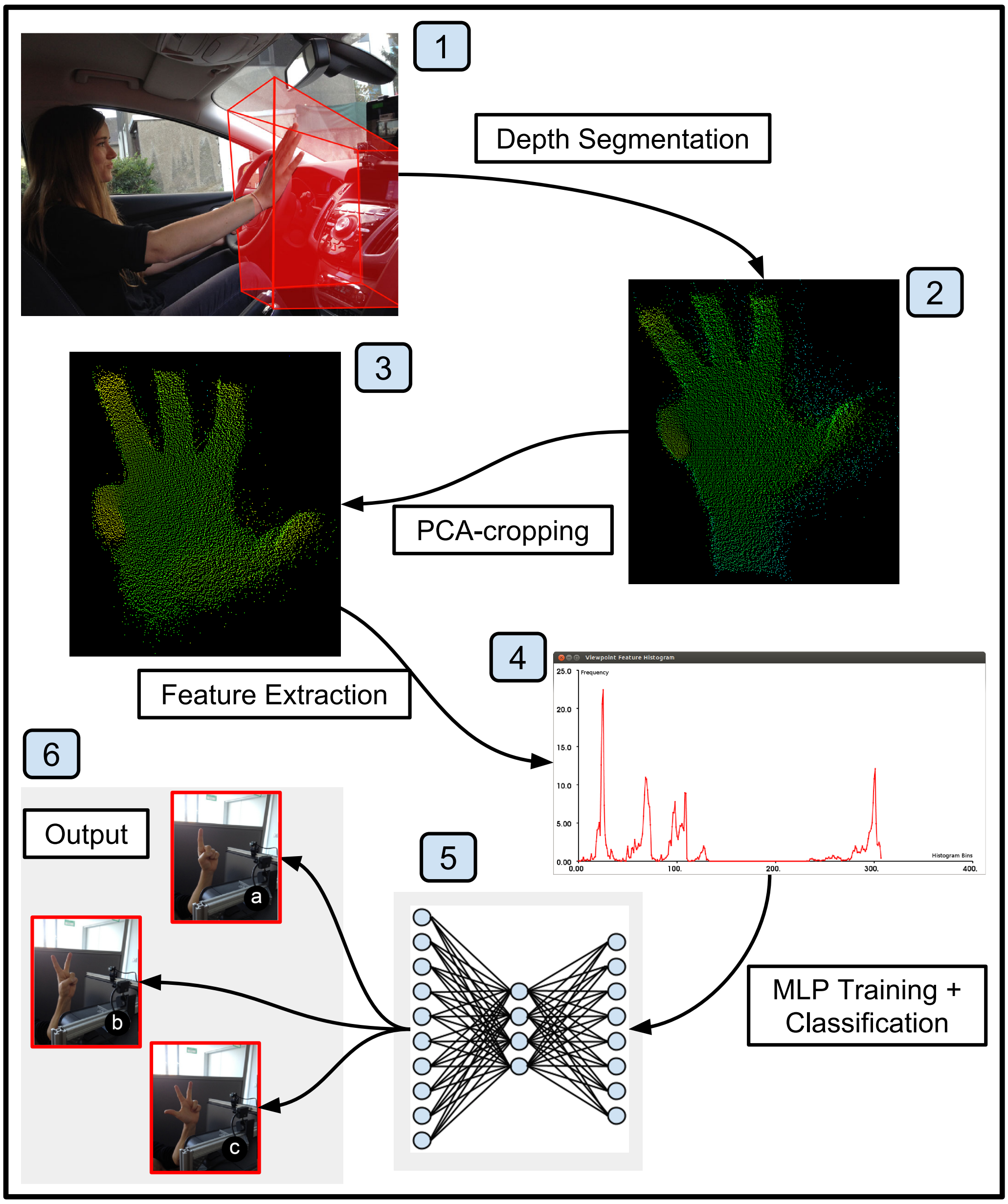


Figure 1: The schematic setup of the HGR system developed in this thesis. The volume of interest (marked red) outlines the region sensitive to user input (1). Data recorded in this area is presented as a depth image (2) containing x-y-z information (here the depth is color-coded, green being more distant to the point of view). This data is cropped to contain only the relevant information (3) by principal component analysis (PCA). Extracting features from the data is realized by creating a *descriptor*, here in the form of a histogram (4). Classification is implemented by training and utilizing a multilayer perceptron (5), its output determines the detected class (6).

However, the signal to noise ratio for depth measurements coming from these cameras depends, amongst others, on the color of the target object, therefore on the amount of light reflected (the reflection coefficient of the medium). To make up for potentially high noise, a robust feature extraction process is developed. These features, termed descriptors, comprise the geometry of an object, reducing the dimensionality of the data while being robust to interferences and remaining computable in real-time. These descriptors form the input for the training and classification steps of the system. The recognition part is handled by neural networks (NNs), more specifically by multilayer perceptrons (MLPs), with sophisticated Machine Learning algorithms at its core, allowing to discriminate between a variety of different hand gestures. MLPs facilitate efficient, robust and fast ML techniques and are therefore the means of choice out of the many available and established classifiers. They moreover comprise a set of distinct features especially suitable for the task of multiclass classification and will be motivated to that effect in detail later on.

This work can be placed at an intersection of the research fields of Computer Vision, Machine Learning and Human-Machine Interaction as the goal is to have real-time applicable software which is easily transferable into different scenarios, remaining robust to various kinds of interferences. Hence, the system was embedded into the automotive environment, being an HMI context where software tools have to meet demanding criteria in order to demonstrate the aforementioned goals.

This thesis is divided into two parts: Part I presents an introduction into the most important topics relevant to the three aforementioned research fields. Building on these prerequisites, Part II describes how each of the set goals was achieved, putting each of the chosen means into the overall context.

The following chapter motivates the chosen 3D sensor technology, followed by a short introduction to the topic of gestures. Afterwards, the requirements of creating such a system are presented from the viewpoint of HMI in Section 1.3, followed by a more specific description within the automotive framework. Then, a broad overview of the related work along with its applications puts the underlying contribution into context of the vast amount of related work dedicated to this topic so far, presenting the pitfalls as well as the possibilities of the chosen approach. In order to be able to make sense from 3D data, Section 2 explains the background of the 3D algorithms employed in this thesis along with the advantages as well as the challenges. Subsequently, an introduction to the topic of Machine Learning in Section 3, with a focus on multiclass classification, motivates the chosen algorithms for the recognition of hand gestures.

Part I

Foundations

Chapter 1

Problem description

Dealing with the task of hand gesture recognition entails a number of difficulties. From the technological perspective, data retrieved from any type of sensor is susceptible to miscellaneous error sources, e.g. from illumination interferences, sensor resolution or stemming from the nature of the topic itself. From the user's perspective, all kinds of expectations have to be met, mostly at the interface between human and machine, in order to develop a well-functioning software. This section aims at addressing these difficulties, explaining the various available approaches and alternatives as well as motivating the means of choice, beginning at the opted sensor technology.

1.1 3D sensors

The task of hand gesture recognition creates a number of challenges, some of which cannot be dealt with by the approaches utilizing RGB cameras. 3D depth sensor technology is the means of choice for a number of reasons. Depth data coming from these sensors evidently has the advantage of adding another dimension, opposed to optical approaches. Relying solely on RGB data makes it difficult to e.g. distinguish between multiple adjacent blobs of skin-colored regions belonging to different parts of the body, different hands or even different persons overall. Such tasks become easier to handle with the help of 3D sensors through simple depth segmentation.

It is moreover possible to capture fine-grained structures of the surrounding environment, or even the hand itself, by using depth sensors, information which is not obtainable from optical approaches or which can sometimes only be extracted via complicated algorithms, resulting in increased computation time. Last but not least, depending on the sensor technology utilized, 3D depth cameras allow for setting up frameworks capable of functioning under difficult lighting conditions, as shall be explained throughout this chapter. A distinction of sensor technology is sometimes made between active and passive sensor technologies, the former being called active due to the fact that a specific signal is actively emitted into the scene and its travel time is measured from the reflected objects in the scene. Passive sensors do not emit a signal of their own but rather gather any kind of signal reflected or emitted from objects in the surrounding environment.

This section presents the most important and commonly employed 3D sensing technology for works in research and industrial applications along with the respective advantages and disadvantages and contrasts it with the already mentioned ToF technology.

1.1.1 Depth by triangulation

Depth measurements can be retrieved by methods called *Active Triangulation* and *Passive Triangulation*. Depth by *Passive Triangulation* is an optical method, simultaneously utilizing two cameras aligned on a common baseline. Depending on the distance to be observed, this baseline is extended to be able to measure depth in further distances. To determine depth, the *correspondence problem* has to be solved - i.e. matching a pixel in one image to a pixel in another image. This method is highly susceptible to noise such as intensity and color variation in an image. Moreover it becomes increasingly complex when trying to determine corresponding pixels from homogeneous regions. Therefore, it is computationally intensive insofar as reliable features have to be calculated first while it is unclear, how confident these features can be, depending on the problem. A benefit of this approach is that any low-cost camera can easily be utilized.

Depth by *Active Triangulation* combines only one camera with an active structured light emitter, projecting a single line or a complete pattern set onto the scene. There are many disadvantages here such as the need of a strong power source or low frame rates due to extra scanning iterations of the pattern, however the biggest problem - standing in contrast to the ToF cameras - is that this technology requires very controlled lighting environment making it unsuitable for the underlying purposes (cf. Foix et al. [26]).

1.1.2 Depth measurement by Kinect

The Kinect sensor released by Microsoft in November 2010 has been the basis for many applications and research interests in the field of Computer Vision and HMI. One of the benefits is that it is able to deliver depth and RGB data simultaneously

allowing it to tackle difficult problems such as segmentation and occlusion of body parts in object recognition tasks more easily. It has proven to be applicable to numerous other tasks as shown later on in Section 1.4.
The system, as described by the inventors in Freedman et al. [28], consists of an infrared laser emitter, an infrared camera and an RGB camera. Depth data is now calculated by a so-called triangulation process. A single laser beam emits near-infrared (NIR) light which is subsequently split into multiple beams creating a *speckle pattern* (see Figure 1.1). This pattern is projected onto the scene and collected by the Kinect's infrared camera which can then be correlated against a reference pattern. The reference pattern is obtained by capturing a plane at a known distance from the sensor, and is stored in the memory of the sensor. Any 'mispositioning' of a speckle results in a shift along the baseline of the sensor and thus depth can be easily measured by the disparity of the collected pattern towards the reference pattern.

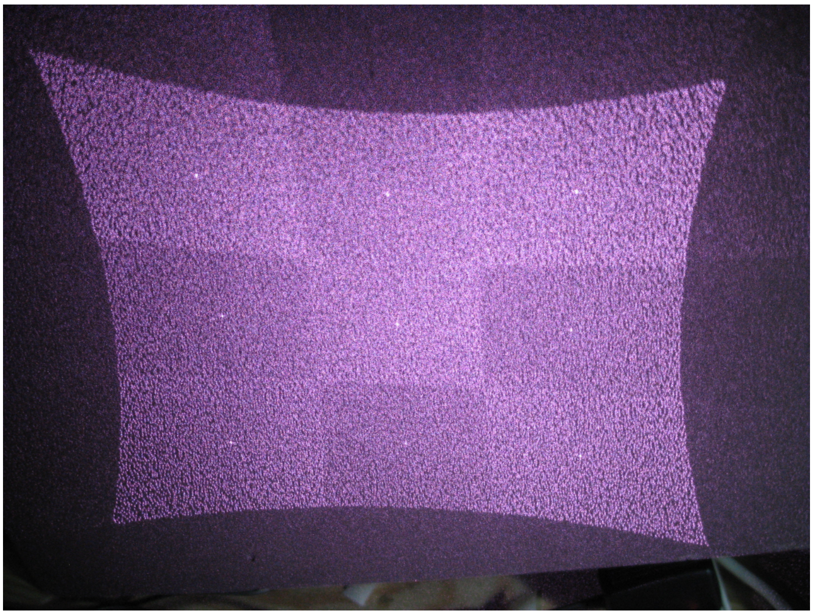


Figure 1.1: The Kinect reference pattern made visible. Any object interfering with the pattern would result in distorted points, making the shift measurable.

The Kinect operates at frequencies of up to 30Hz depending on the resolution, has an operative range of 1.2m-3.5m and a resolution of 640x480 pixels, thus the data collected at a single time frame contains a coloured point cloud of over 300,000 points. The main disadvantage is its susceptibility towards other infrared light sources making it unsuitable for outdoor scenarios. In comparison to ToF cameras its frame rate is low, however it is also low-cost and provides additional RGB data of the scene.

1.1.3 ToF sensors

Name	Resolution	Framerate	FoV	Size	Illumination
Camboard Nano	160 x 120	<90 fps	90 x 68	37x30x25	850nm
Camcube	200 x 200	<40 fps	40 x 40	-	870nm
Argos 3D - P100	160 x 120	<160 fps	90	75x56x27	850nm
SwissRanger 4000	176 x 144	10-30fps	69x55	65x65x76	850nm

Table 1.1: List of the most common ToF sensors employed for research and industrial applications.

ToF-Sensors have emerged in the recent past as an attractive alternative for retrieving 3D information about the nearby environment. They appear with different specifications (cf. Table 1.1) with the main advantages being high frame rates, robustness versus other light sources and comparatively low price. The CMOS chips coming to use in these sensor allow for calculation of depth and greyscale information simultaneously.

A ToF sensor typically consists of an illumination unit, a lens, an image sensor, driver electronics and an interface. Usually an LED or a laser diode is used to illuminate the scene in near-infrared light operating at a wavelength of ~850nm - just outside the visible electromagnetic spectrum (cf. Figure 1.2) - as on the one hand speed is a crucial factor for the calculation of depth and on the other hand the surrounding environment should not be disturbed. The lens is responsible for collecting the reflected light, typically paired with a chip for suppression of background illumination (SBI). The sensor is responsible for measuring the time the light traveled forth and back and calculates the depth per pixel. The setup is complicated thus a pixel sometimes reaches $100\mu m^2$ in size (pixels on modern cameras in mobile phones can be about $63\mu m^2$ in size). The driver electronics operates at a precision within the scope of picoseconds in order to be able to measure light pulses for the desired frame rates. The calculation of distances is also usually realized directly on the sensor, data is transmitted via USB or Ethernet.

1.1.3.1 ToF sensors with lightpulse emission

In order to measure the distance to an object via pulse modulation the surrounding scene is illuminated by the light source and the amount of reflected light is collected for every pixel. This happens simultaneously and is realized by photo-detectors operating in the scope of picoseconds. During a fixed time window (Δt) the amount of reflected light (cf. Figure 1.3) is sampled by the two out-of-phase windows C_1 and

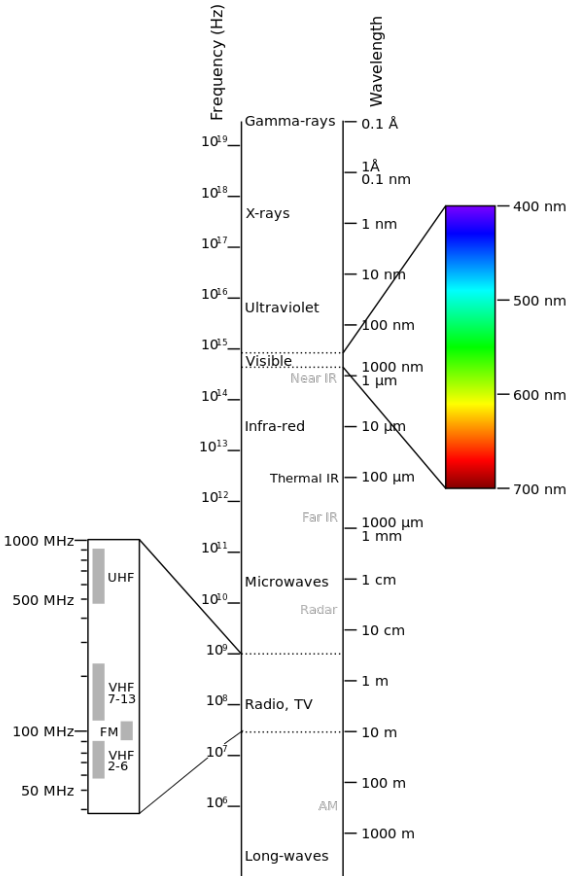


Figure 1.2: The whole electromagnetic spectrum. The visible amount of light ranges from 390nm to 700nm. ToF cameras operate just out of this range at near infrared(NIR) - at around 850nm. Image: Wikipedia.

C_2. This results in two electric charges measured, Q_1 and Q_2 respectively, and hence the distance d can be calculated via

$$d = \frac{1}{2} c \Delta t \frac{Q_1}{Q_1 + Q_2} \tag{1.1}$$

Here, c is the speed of light.

The *integration time* of a ToF camera is the duration of data acquisition - the time from light emission until measuring the reflection of the emission. Longer integrations times allow for measurement of objects in further distance while shorter integration times are more suitable for near-range interaction. Setting the integration time too high can result in oversaturated pixels for objects close by.

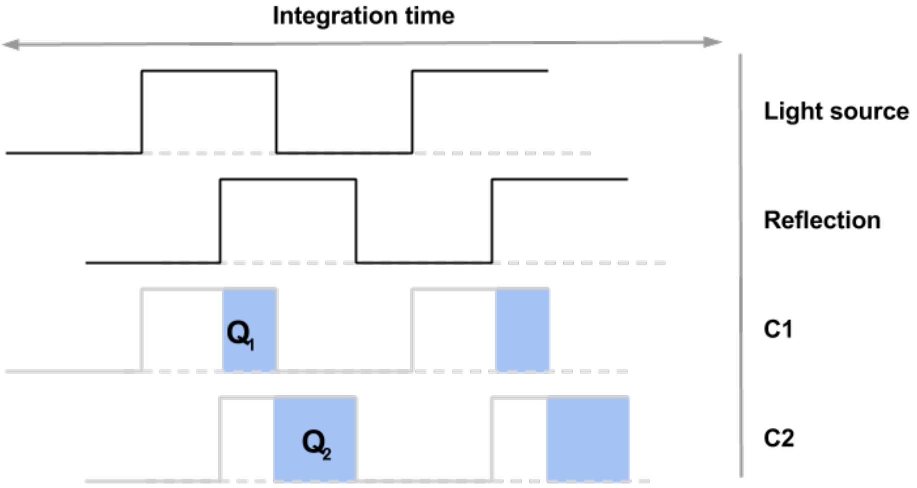


Figure 1.3: Light sampling with pulse modulation.

1.1.3.2 ToF sensors with wavelength modulation

ToF sensors based on wavelength modulation derive the depth data of the surrounding by continuous wave intensity modulation (CWIM) [55]. A modulated NIR signal o_τ is emitted by the LED of the sensor, reflected by the scene and collected by the sensor's lens. The photons collected by the lens are transformed into electrons pixelwise by a CMOS chip integrated into the collector for each pixel. The offset between emitted and collected signal is calculated as the difference in capacitance and thus makes the exact distance to each object measurable. This calculation is realized directly on the chip, also referred to as smart pixels, and results in a high frame rate of the sensor.

The *phase-shift algorithm* calculates the autocorrelation function (ACF) between the emitted optical signal and the electrical reference signal. To this end, four samples m_1, m_2, m_3, m_4 are taken at equidistant points in which allow for the calculation of the phase shift φ by

$$\varphi = \arctan\left(\frac{m_3 - m_1}{m_0 - m_2}\right) \tag{1.2}$$

the offset B by

$$B = \frac{m_0 + m_1 + m_2 + m_3}{4} \tag{1.3}$$

and the mean amplitude A by

$$A = \frac{\sqrt{(m_3 - m_1)^2 + (m_0 - m_2)^2}}{2} \tag{1.4}$$

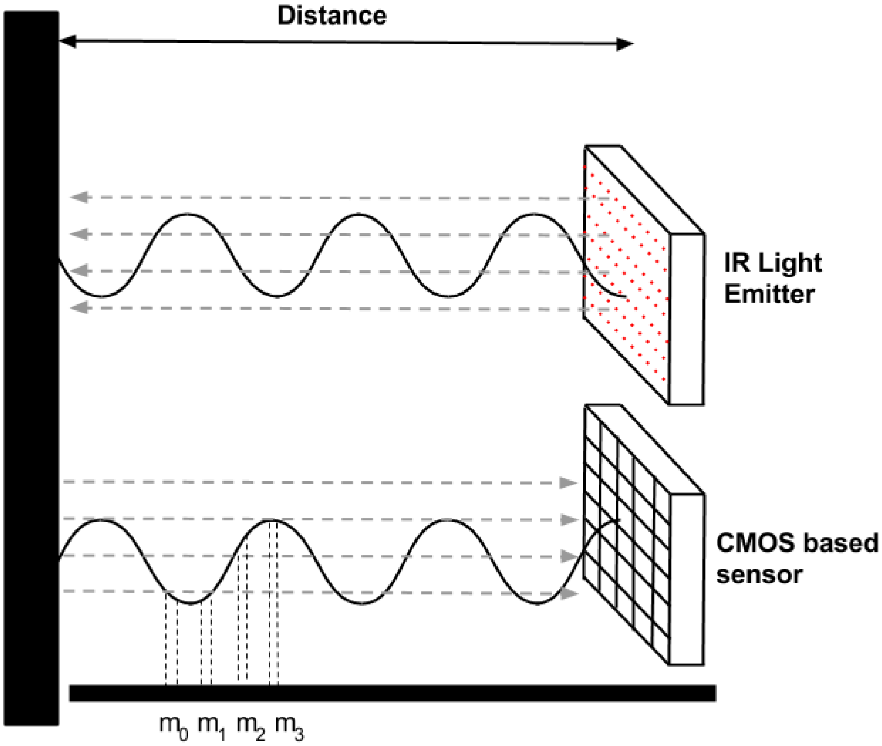


Figure 1.4: Measurement of phase offset at four equally distant points in time.

This process is also known as *four-bucket-sampling* (see Figure 1.4). It is a demodulation of the phase and permits to calculate the depth

$$D = L\frac{\varphi}{2\pi} \tag{1.5}$$

The mean amplitude A determines the strength of the signal. The modulation frequency f_m of the light emitted by the LED defines the so-called *ambiguity-free distance range* of the sensor

$$L = \frac{c}{2f_m} \tag{1.6}$$

At a modulation frequency of $f_{mod} = 20MHz$, for example, the wavelength λ_{mod} is 15 meters, and therefore the maximum distance of an object is 7.5 meters, as the signal has to travel forth and back.

Background light is suppressed via an SBI chip responsible for deducing the uncorrelated from the correlated amount of light via the ACF. This allows for operating a

ToF sensor with an SBI at full sun exposure of $150klx$ and is one of the main advantages of the sensor as interesting applications become realizable in outdoor scenarios under real-time conditions. Difficulties of this technology are possible ambiguities in measurement of the collected signal as the measured distance can refer to a multiple of the phase shift. Hence ambiguities occur every 2φ as the phase wraps around. The maximum distance measurable unambiguously therefore is defined by

$$d_{max} = \frac{c}{2f_{mod}} \tag{1.7}$$

As this work aims for an application recording the nearby environment, ambiguities will not occur in the current setting. Reducing the modulation frequency allows for increased distances to be measured, simultaneously reducing precision. An alternative is realizing multiple modulation frequencies and disambiguating possible ambiguities by comparing both frequencies for overlapping phases - equaling the true distance. The benefit of illuminating the scene as a whole comes at the cost of the optical output - powerful LEDs are necessary to achieve satisfactory results. Objects in further distance become measurable with higher integration times (time for illumination, light collection and distance calculation) while smaller integration times reduce the signal/noise ratio.

1.1.4 Conclusion

ToF-based measurement comes with more benefits than drawbacks in the context of HMI applications. When looking at the aim of hand gesture recognition devices, they typically acquire data in the nearby environment. Being able to capture nuances such as finger movements or slight hand tilts would require high resolution cameras for distant measurements. Hence, a problem as the turnaround distance (the maximum measurement distance) becomes obsolete for a task such as HGR. This will become more evident later on, when looking at the multitude of scenarios presented in Section 1.4, all of which placing HGR in the sensor-near environment - medical apps, VR/AR-apps or infotainment systems as in this work. The sensor chosen for the underlying thesis is the Camboard Nano (cf. Table 1.1), operating at 90 fps at a resolution of 160 × 120px and based on continuous wave modulation.

The light emitted by the IR-chip is modulated with a frequency of 30Hz, posing the problem of interferences when operating with multiple sensors at the same time, however it is explained context-dependently later on how this problem can be dealt with. In contrast, the great advantage of ToF technology is the ability to retrieve

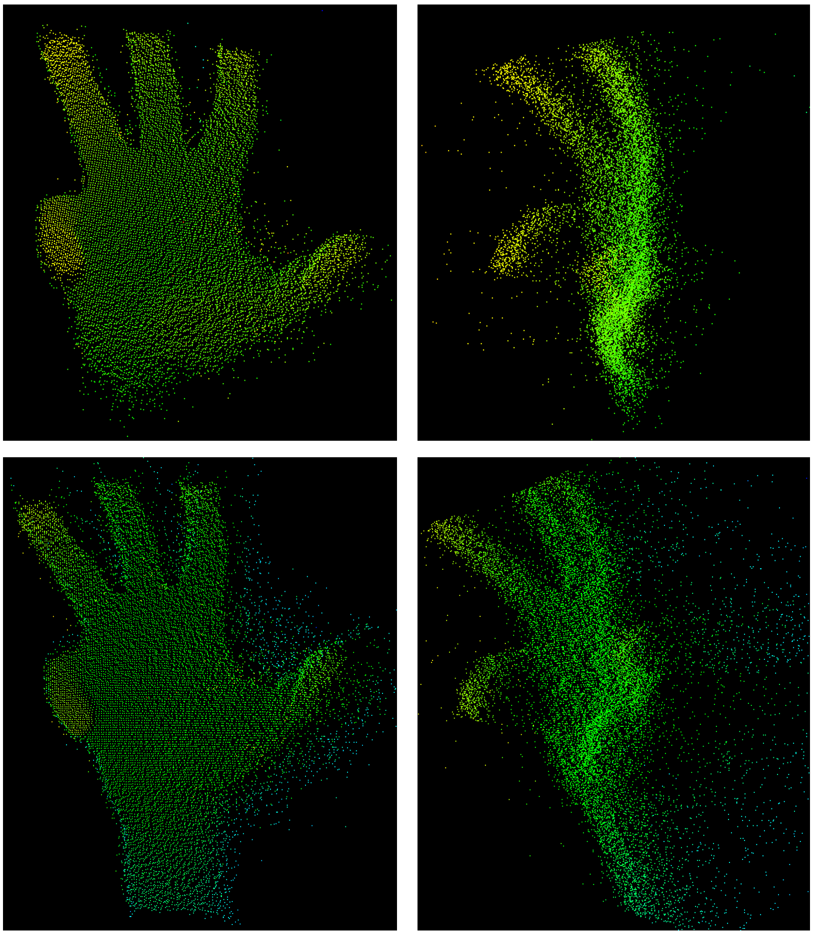


Figure 1.5: 3D data retrieved from a ToF camera. Top row: Front and side view of the same point cloud. Bottom row: Front and side view of the same point cloud without PCA cropping. The number of outliers increases towards the sides of the recorded object due to the major amount of light not being directly reflected to the camera.

depth data in outdoor scenarios operating at high frequencies. The inaccuracy of the depth measurements can sometimes be difficult for recognizing details, especially when trying to describe the fine-grained structures of the shapes overall.

Figure 1.5 displays the depth images of a hand for two separate shots and from two different angles respectively, recorded by a ToF camera. It is obvious how this technology can be utilized to catch fine-grained details while also revealing its susceptibility to noise, especially towards the border parts of the structure. This noise can be accounted to the insufficient and varying amount of light directly reflected by the object towards the infrared sensor, resulting in measurement fluctuations.

In order to overcome such hurdles, sophisticated 3D point cloud descriptors will be introduced and described later on in Section 2. The following sections are a general introduction to the topic of hands with a transition to the research field of HMI.

1.2 On the nature of hands

A number of theories have been developed to explain the evolutionary difference of the human and the animal world. Putting the emphasis on one or only some of them would probably not do them justice, also considering the fact that many coherences remain unclear, one of them being, whether the development of the hand has triggered the increase in brain size or vice versa. Trying to link the ideas on the origin of human intelligence, Frank Wilson [128] introduces the hand-thought-language nexus, basing it on the pillars of the two problem-solving strategies of tool manufacturing and communication via language. While tools are also used by a number of animals it remains undisputed that the variety and complexity of those created by man is unreached, besides the fact that we use them in order to create even more (complex) tools. All this would not have been possible if it wasn't for the complexity of the hand's bone arrangement along with its 27 degrees of freedom (DoF), this being one of the reasons, which makes the study of the underlying topic so interesting.

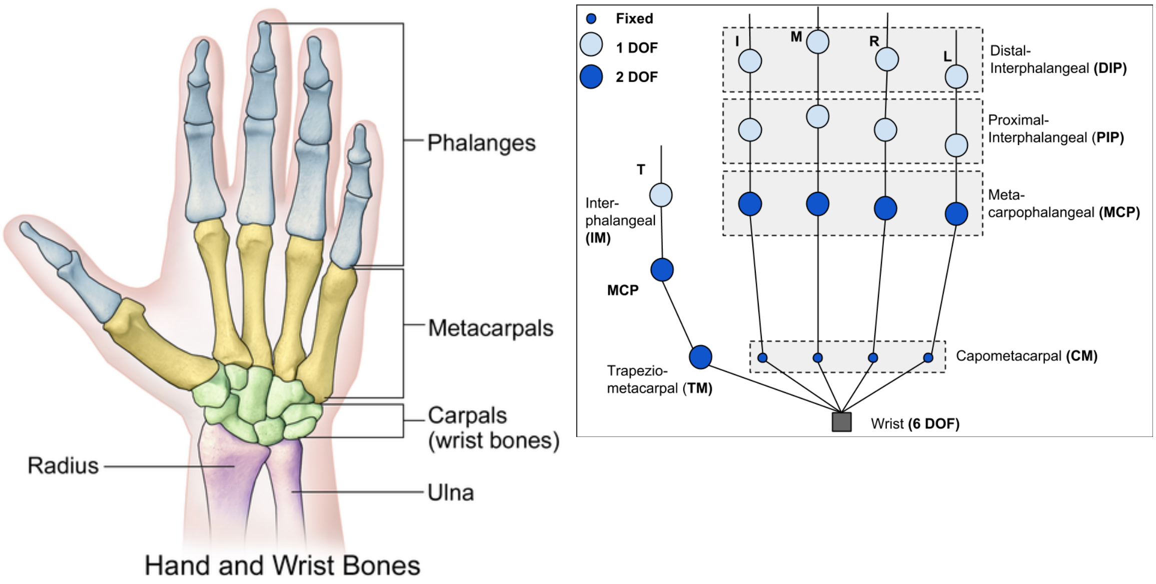


Figure 1.6: The hand with its complex bone structure (left, Source: Blausen.com staff. "Blausen gallery 2014".) and its resulting 27 degrees of freedom (right).

So where exactly does this complexity come from? Figure 1.6 displays the bone structure of the hand (left) and the resulting DoF (right). Being able to move some parts of the fingers in one, others however in 2 dimensions along with the 6 DoF for the wrist results in the overall 27 dimensions.

Creating a tool to be able to make machines interpret and understand the complexity of the hand is a challenging and unsolved task. There is a lot of research directed toward this topic as the applications, which will be laid out in detail later on in this work, are countless. One could imagine replicas of human arms to be helping disabled people, robots mimicking human behaviour to achieve complex tasks or systems controllable by the varieties of hand gestures as already happening in 2D on smartphones and tablets, or envisioned in movies and partly realized by virtual reality environments.

1.2.1 Gesture taxonomy

On a general level, hand gestures are classified into two categories - static and dynamic hand gestures. The former are considered as being non-changing, i.e. remaining static in shape and position over several points in time (cf. Kanniche [41]). The latter by contrast are considered as changing in position and/or in shape. For example, pointing with a finger towards a specific item would be seen as a static gesture (possibly connoted to selecting something) while moving the hand and leaving the fingers in position would be a dynamic gesture and could resemble altering the position of an item. Static gestures are often referred to as poses, though in literature both terms are often used interchangeably.

Problems can arise, stemming from different cultural backgrounds, as a hand pose can be referring to a harmless indicator in one part of the world while being seen as offensive in some other part. This was, in fact, the case for the 'three gesture' (cf. Figure 4.2) as, in a study conducted for this thesis, one participant was reluctant to demonstrate this pose due to ethnocultural reasons. Neuliep [69] explains how seemingly meaningless gestures are misconceived in different parts of the world.

One of the reasons for setting up the database, as described later on, was to have a set of complicated hand poses which are meaningful on the one hand and difficult to disambiguate (for vision-based approaches) on the other hand. Some of the challenges one is faced with when implementing a gesture alphabet is e.g. the fact that counting from 1-5, as included in our database, is performed differently on various continents hence it can be concluded that a gesture set, as chosen for this thesis, might be difficult to be motivated. Nevertheless, because of the aforementioned reasons and furthermore with regard to the intention of developing a software tool for daily use, the database comprises the displayed gestures which, not least, allow for the extension to dynamic gestures as described towards the end of this thesis.

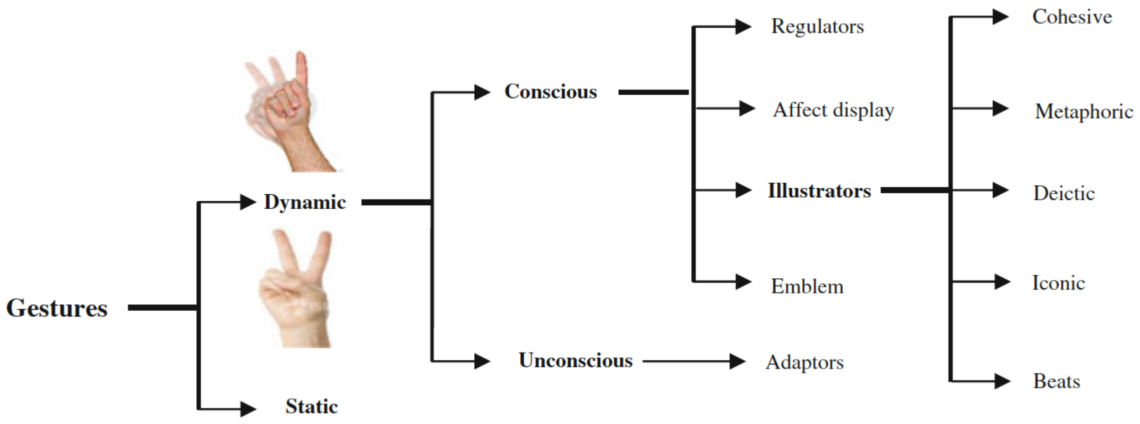


Figure 1.7: The gesture taxonomy according to Kanniche [41]

The whole taxonomy of dynamic hand gestures, as depicted in Figure 1.7, goes beyond the scope of this thesis, however, within this frame, this work aims for detecting static and dynamic gestures and within the dynamic category, for detecting conscious gestures, i.e. the ones being performed intentionally.

1.2.2 Hand gesture recognition approaches

Several surveys have been conducted and give a concise overview over the vast body of work done so far in the field of hand gesture recognition [17], [20], [65], [83], [92], [117], [131], [64], with different foci applied, e.g. to underlying techniques, HMI-related questions, algorithmic approaches or current challenges in general.
Regarding the algorithmic approach there are two different kinds of techniques, one being the model-based approach in which a hand model is created to resemble the 27 DoF of the hand and the other being the *data-driven* or *appearance-based* approach aiming at interpreting e.g. the colors or the silhouette of a certain region in the data. Naturally, both methods can be combined to improve recognition results - with one obvious drawback being the impeded computation time - however the underlying thesis follows the purely data-driven approach as one of the main goals is to demonstrate that hand gesture recognition can be performed in real-time without the need to formalize a complex model.
With respect to the interpretation task, one usually follows the three-step approach of detecting a region of interest, tracking it and implementing a recognition algorithm at the end to connote some sort of meaning (hand pose/gesture) to the data. Each of

these individual steps can be achieved by numerous techniques (cf. Rautaray [92] for an overview), however the underlying thesis shows that more than satisfying results can be achieved by establishing a large database and using one classifier with sophisticated fusion techniques, coupled with a simple definition of dynamic hand gestures, making the need for Hidden Markov Models (HMM) or Finite-State Machines (FSM), and even the need for computationally expensive tracking algorithms, obsolete.

1.3 Human-Machine Interaction

One definition of HMI refers to the research and development of interfaces focusing on the behaviour of humans during an interaction process as well as the design of computer technology. The term *usability*[1] frequently co-occurs when speaking of HMI and it denotes, on a general level, a measurement of how easy something can be used or be learned to use. HMI aims at improving the usability of interfaces to be developed and therefore, within the context of this thesis, the goal is to improve the usability of an interface controlled by human hand gestures. More specifically, the aim is provide an interface which is intuitive[2] to use, provides a low hurdle of entry and optimally lowers the cognitive load for the subject.

Computers have increased in computation power, and with simultaneous decrease in cost and size, new means of interaction have been developed. As stated by Wigdor et al. [127] these evolution steps have taken place more or less discontinuously, rather in phases. The outcomes of these phases still persist today as e.g. the command typing via terminal, followed by the graphical user interface (GUI), are probably still the most widely used interfaces in most households and offices around the world. During these phases naturally some interfaces proved to be more predominant over others, which is why they prevailed, especially for certain tasks. However, as the authors further claim, in many situations there exist hybrids which can be seen by a more in-depth analysis. For instance, the console-like element is available in many different OS as a means to quickly search for items stored on some hard drive because this simply has shown to be the easiest and most efficient way to solve the given task (opposed to cumbersome inspecting the contents of each folder) for trained users. Another example of console-like elements is the possibility to enter complex formulas.

[1] ISO definition of usability: The effectiveness, efficiency and satisfaction with which specified users achieve specified goals in particular environments.

[2] *Intuitiveness* is a fuzzy term with little methodology to define it. However, gestures are considered as an intuitive means of control, as a study conducted for this thesis demonstrates.

It is difficult to imagine that a GUI controlled by a mouse could prove superior for this task. These examples demonstrate what the phases referred to by the authors have evolved into, namely some form of interface merged together, stemming from pieces, each being optimal for its specific, designated task.

With this evolution the way we learn to interact with machines has evolved as well, as the number of tasks human beings are capable to perform with the computer evolves with it. One of the questions researched by the field of HMI is whether this evolution was driven by the fact that many more people were able to use computers because it became easier to interact with them or vice versa. This question is beyond the scope of this thesis, however there is a certain factor immanent to the nature of gestures, namely the degree of naturalness associated to the medium of interaction, which this work also intends to shed some light on. Again, whether this naturalness was driven by some reason or the other way around is not the focus of this contribution. Nevertheless, the author believes that a certain degree of naturalness is contained with gestures as a means of interacting with machines. Thus it is of imperial interest - and as a natural result of the research conducted here in first place - to have a look at to what extent gestures indeed fulfil this presupposition, since this may be one reason for why interfaces controlled by gestures carry this notion of *naturalness* with respect to the interaction phase.

As Wigdor et al. claim, one possible step in the evolution is the development of natural user interfaces (NUI). NUIs seem to be in a similar position to the GUIs developed in the 1980s with respect to reducing the barriers to computing even further and simultaneously increasing the power of the user. In order to not try and predict the future (too much) they further claim that NUIs are 'here to stay' either as a dominating trend or by finding themselves a niche. In this work, hand gestures are considered as one possibility of creating such a natural user interface. Hand gestures are, by no means, an exhaustive way of interacting with machines as the limits, within which one has to confine oneself to, become evident very quickly. However, this also applies to mouse and keyboard, possibly the most common means of interaction with computers, but is clearly observable that the two degrees of freedom cannot emulate the three dimensions of space.

The advantages of an interaction via hand gestures are obvious as well. Be it in situations, where access to a regular mouse and keyboard interface is not feasible - as e.g. in an operating room in a hospital - or in a possibly less danger-critical scenario as interacting with an infotainment system within a car, where the driver is assisted in controlling various menu points while retaining most her/his attention on the traffic. This is the scenario of choice in this thesis for showing the possibilities as well as the limits hand gesture interaction can provide.

The following section places hand gestures in the context of in-vehicle HMI scenarios along with the challenges and possibilities, after which the related work section provides an extensive overview of hand gesture interaction with a focus on HMI concepts, technologies applied and the algorithmic ideas in the background.

1.3.1 HMI for Advanced Driver Assistance Systems

Over time, the interior of the car has become more and more complex when looking at the technologies installed to assist the driver in various tasks. The main goal of these Advanced Driver Assistance Systems (ADAS) is to increase driver safety or road safety in general as well as the driver's comfort. Examples of well-established representatives include the Adaptive Cruise Control (ACC) regulating the vehicle speed with respect to the vehicle running ahead, Lane Departure Warning Systems responsible for emitting warning signals when the current lane is being left or Collision Avoidance Systems aiming at reducing the damage in case of an accident. Depending on the degree of 'intrusiveness' these systems vary in the way they interfere in the driving process. A hand gesture recognition interface for in-vehicle infotainment systems would be a non-intrusive variant aiming at reducing the gaze diversion of a driver, optimally lowering the cognitive load during a driving task.

Some of these technologies are taken for granted and have become indispensable in the in-vehicle context, however, the goal remains to introduce something of use for the driver or passenger while, at the same time, not increasing the driver distraction, but rather decreasing it. Driver distraction and driver safety, some of the major topics discussed at the Intelligent Vehicles Symposium (IV 2015), and the statistics presented list the number of fatalities related to texting in ongoing traffic in the US nearly as high as the number of fatalities due to drinking, moreover studies suggest that the use of mobile phones during a car drive is a real, albeit tangible, threat [25]. While it cannot be assured that a technological solution for writing a text message during a drive would on the one hand reduce this number significantly nor on the other hand guarantee that everyone would make use of this technology, it can be assumed that at least some fatalities could be prevented.

Within this context the field of HMI also takes into account the *user experience*[3](UX) a user has with a system which can be, of course, a main selling factor. Perceiving

[3] ISO definition of user experience: A person's perceptions and responses that result from the use or anticipated use of a product, system or service.

an infotainment system as too complicated to use, too outdated in terms of design or just overloaded with unnecessary functions, which may be even unintuitive to interact with, can be a counterargument for buying a car. Usability, as mentioned earlier, the ease of use and learnability of a man-made system, can be defined through quality components (as in Nielsen [70]):

- **Learnability**: How easy is it for users to accomplish basic tasks the first time they encounter the design?
- **Efficiency**: Once users have learned the design, how quickly can they perform tasks?
- **Memorability**: When users return to the design after a period of not using it, how easily can they reestablish proficiency?
- **Errors**: How many errors do users make, how severe are these errors, and how easily can they recover from the errors?
- **Satisfaction**: How pleasant is it to use the design?

Moreover he stresses the importance of utility, being the question of whether something provides the features a user needs. He forms the *usefulness* in the context of HMI as the synergy of usability and utility.

When projecting these ideas onto the contribution of this thesis, namely the development of a free hand gesture recognition system of in-vehicle application, the learnability is a very important factor of the system. Gesture control should be perceived as easy and intuitive, adding little to optimally no cognitive load to the driving task. When coupling free hand gestures with e.g. an infotainment system, basic functions such as channel selection could be mapped to a single, easily learnable gesture.

Looking at efficiency, hand gestures should provide a faster means of interaction. There are well-established user interfaces in a car developed over the course of years, and a new technology introduced should not slow the driver down during the interaction process. As for memorability, such a system should not demand the driver to learn a large gesture alphabet, hence mapping every function in a modern car to a single gesture is ill-advised. Either the most important functions should be targeted, with a small gesture alphabet, or the additional effort to *explore* these gestures must remain minimal.

Considering errors, the following has to be taken into account: First of all, free hand gestures theoretically have no perceivable 'context area' as e.g. touch gestures performed on a touch display. There is a clear frame within which touch gestures can start and end - such a frame is difficult to realize mid-air in three dimensions, thus it is the task of the developer and the designer to make sure the user can interact clearly with freehand gestures. Moreover, such a frame would make it easier to recognize a gesture, this counts especially for dynamic gestures. In 3D however, the recognition algorithms have to take this into account. While the 2D-technology counterpart on smartphones or tablets has to 'only' deal with the different finger size of each individual, dynamic hand gestures are more complex: how far is the user away?, is she/he holding the hand in a certain direction?, are the user's fingers aligned as assumed?, is the hand size different? etc. Sometimes, as will be seen in a user study conducted for this thesis, the users expect something else e.g. a different system behaviour or reaction, hence this is perceived as an error.

Lastly, satisfaction plays a crucial role for such a system. Luckily, at the time of development of this thesis, free-hand gestures have just begun to emerge as a mass media interaction technique thus user satisfaction could be easily achieved, due to the experience of interaction being perceived as something new, special or sometimes even called *magical*. This was an additional bonus when testing the system with new subjects. Hand gestures however, have some innate naturalness to them and the added benefit of not having to touch any buttons directly, or even use a medium such as a handheld device in between, provides for highly satisfied users, ergo high user experience[4].

When stating the necessity of (re-)designing the interface in the light of usability, Nielsen often relies on savings being made in terms of time saved due to efficient work flow and money saved due to uninterrupted processes being brought to an efficient end. While this may hold true for HGR systems developed, the main focus here is put on natural, efficient control of a system leading to reduced distraction of the driver and therefore reduced gaze diversion.

Young et al. [133] provide an extensive overview, distinguishing between driver inattention and driver distraction, the latter being one variant of the former. Their research pays special attention towards wireless communication, at a time when mobile phones were common and smartphones were just emerging on the mass market, and point to figures estimating a quarter of vehicle crashes in North America related

[4] It has to be noted here that while this may sometimes be considered an advantage, touchless interaction is missing the haptic feedback, which can be beneficial in certain situations.

to driver distraction. They further claim that as more wireless communication is about to enter the vehicle, these figures will even increase strongly. This work intends to improve the interaction with in-car systems via hand gestures, hence it is of uttermost importance that any new technology entering the vehicle does not further distract the driver but provides a reduction in terms of distraction and the cognitive load during a car drive. The author of this thesis believes that a well-integrated system can provide these demands as gestures - e.g. learned by a fixed alphabet - would allow the driver to interact with an infotainment system without having to redirect gaze, simultaneously allowing to keep the eyes on the road.

Regan et al. [94] also raise the concern that driver distraction and driver inattention, for which there is growing evidence that they are major responsible contributors for vehicle accidents [44] [77], will increase with the number of introduced technologies in the car. However, their focus is on building a taxonomy in order to correctly define the terms driver inattention and driver distraction and put them into a correct taxonomic relationship to be able to properly assess their roles in car accidents. This goes way beyond the scope of this work, yet it shall be remarked the authors also define driver distraction as one form of driver inattention, which is the definition also adopted in this thesis.

1.4 Related work

The introduction of the Kinect to the consumer market by Microsoft in November 2010 spawned numerous applications and fields of research, including hand gesture recognition. Being able to fuse information from its depth sensor as well as its RGB camera allowed for powerful algorithmic design, however it was mainly restricted to scenarios with controllable lighting environments. A more general overview over Kinect applications developed so far can be found in [33] [135].

The number of applications derived from hand gesture controlled interfaces is immense as is the number of approaches taken to solve this task. They differ in the technological means utilized, the algorithms processing the data, the detection algorithms adapted and tailored to the task, HMI-related questions or possible application scenarios which accordingly impose additional parametric specifications. The vast amount work already dedicated to solve this problem can therefore be assigned to one or more of the listed differences but as this thesis deals with a vision-based approach focusing on object recognition in 3D, the related work is assigned to either the two- or three-dimensional category (Section 1.4.2 and 1.4.3) or to the hybrid

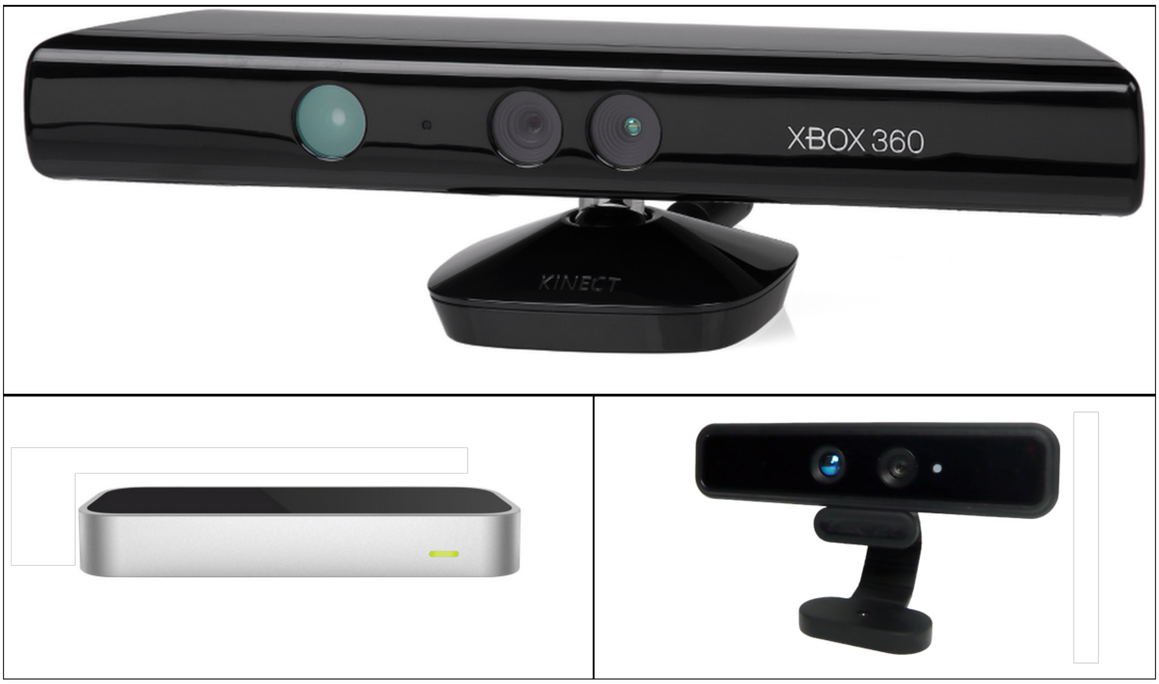


Figure 1.8: The Microsoft Kinect Sensor (top). The Leap Motion Controller (bottom left). The SoftKinetic Cam (bottom right). Source: Wikipedia.

category (Section 1.4.4), fusing both kinds of approaches. Towards the end of the hybrid section special attention is paid to the Kinect sensor being the driving force for research conducted in the area of HGR. Some of the work relevant for this thesis deals with HMI-related questions which is dealt with in Section 1.4.5. For the sake of completeness, related work utilizing contact-based devices is presented first.

1.4.1 Contact-based approaches

The technologies behind hand gesture recognition systems can be divided into two categories: The ones using contact-based devices and vision-based approaches. The former group covers all systems where the user needs to use some kind of device by e.g. touching or wearing it, such as a data glove or an accelerometer. Such devices usually have the prerequisite that the user must be taught in some kind of way to be able to use it. Furthermore the range of application scenarios is limited as the hand is somehow occupied - e.g. by holding an accelerometer device - or the wires potentially impose some kind of limited range. Wireless gloves of course have a time limit determined by the battery life, moreover they are not always available and not desirable from a user's perspective.
Liu et al. [60] demonstrate a promising system based only on a single personalized

hand gesture performed by each individual user with an accelerometer device as they argue a hand gesture can be defined via a series of forces. They claim that hand gestures lack a properly defined vocabulary as e.g. in speech recognition environments, which is also partly shown by research conducted for the underlying thesis, as many users on the hand have a different connotation to e.g. counting from one to five by gestures (shown by experiments) and furthermore slight variations, which are unavoidable, result in unforeseeable consequences i.e. misinterpretations. They apply their work to consumer electronics and mobile devices which makes sense, as these frequently contain accelerometers nowadays. The research for HGR systems directed towards mobile devices is restrained mostly for employing the accelerometer of a mobile phone as in the work of Kallio et al. [42].

Gesture recognition has spawned a plethora of work for virtual (VR) and augmented reality (AR) applications as it seems natural to interact in such an environment with the own hand, given that the idea is to emulate or 'extend' the real world experience. Weissmann et al. [125] present one of the first ideas for collecting user data with data gloves and interpreting it with neural networks in order to control a robot hand in a virtual environment. An AR system, also based on markers, is presented by Reifinger et al. [95] differentiates between static and dynamic gestures via distance models and HMMs to manipulate objects. The average task, although performed by the users with increased discomfort, was finished faster and felt more intuitive to be completed.

The Nintendo Wii released to the market in 2006 has a control device similar to a remote control, allowing for tracking the user's movement by an infrared camera recording the scene and an accelerometer within the device. The idea of employing Wii-sensor data to define user-specific gestures and interpret them via HMM, has been presented by Schlömer et al. [110] and applied to build up a multimodal interface for photo browsing and TV control. The gesture set however as well as the evaluation are comparatively small.

A further area of research covers the area of sign language recognition (SLR). Liang et al. [59] propose using a DataGlove for recognizing a vocabulary of 250 signs from the Taiwanese Sign Language with HMMs. Improved recognition rates were achieved by Starner et al. [115], as well by using a data glove and HMMs, however on the American Sign Language (ASL) and with a more in-depth analysis of the problems as well as the perspectives.

1.4.2 2D vision-based approaches

2D vision-based approaches usually demand a controlled lighting environment. Oka et al. [76] developed a desktop system called *EnhancedDesk* for detecting and tracking fingertips via an IR-Camera recording the scene top-down and interpreting the movement via Hidden Markov Models (HMM). The IR camera is adjusted to the human's body temperature for segmenting hand pixels from non-hand pixels. They base their system on the assumption that users tend to use the thumb and index finger for fine-grained interaction in order to manipulate virtual and real objects at a working desk. Another desktop application presented by Sato et al. [109] uses the fused information from multiple color cameras directed at a working desk to present a GUI manipulation system via hand gestures. Neural networks trained on a database are used to distinguish a set of six different static hand poses. The same system is set up in front of a large immersive display to demonstrate the application range as well as its simple extensibility.

A typical generic household scenario, as suggested by Roberts et al. [101], can comprise of a camera recording part of the room's environment reacting to a set of user inputs which are interpreted by a processing unit of the whole system controlling various home appliances. A more concrete proposal by Irie et al. [38] describes an intelligent room within which the interior is recorded by CCD cameras to detect gestures as waving or pointing based on color images and skin pixel detection. Their system controls various home appliances via simple hand and finger gestures, however frequent misinterpretations still occur, albeit being installed in a place with rather controllabe lighting. Moreover the authors make no suggestions of how to deal with typical problems such as multiple users or unwanted user input.

Walter et al. [124] transport the idea of employing hand gestures on public displays, potentially outdoors, however with a focus on raising the awareness of the users to a system with the underlying controllable interface. One of the more interesting findings of their work, regarding the application side of this thesis, is that once the awareness of the user is raised, coupled with positive feedback, she/he will tend to explore the gesture alphabet in a certain direction. Nancel et al. [68] levitate the power of mid-air hand gestures to the interaction of humans with man-sized walls. Their work is primarily directed to the question of what kind of key factors have to be considered for the design of such systems. Although their findings refer to a different scenario, common factors as arm fatigue and a reduced accuracy after elongated usage can be observed and should be considered when designing a system for hand gesture interaction.

The idea of employing hand gestures for computer games goes back into the 90s, as Freeman et al. [29] present their idea of developing games for an artificial retina module and control game avatars with the human hand and body, based on image moments and orientation histograms obtained from visual input.

A possible scenario for interacting with a virtual environment is presented by Lyons et al. [61]. It is non-immersive as a camera observes user input in 3D for manipulating objects in VR environments. AR applications intend to intensify or extend the real-world experience for the user, or to blend the virtual and real world in a seamless way. Buchmann et al. [12] argue, as hands are our natural means to interact with objects in the real world it is only natural to integrate them into an AR system. They present *FingARtips*, a system for near-range interaction and manipulation of objects on a table in an urban planning situation. The scene is captured with a webcam, extra markers help with finger and hand tracking while the recognition is based on the finger positions via simple thresholding. Electronic buzzers provide extra haptic feedback to recognize e.g. grabbing gestures as this can be difficult in AR environments, however the overall performance, although satisfactory, is strongly impeded by lighting problems during marker detection.

A popular field of application and research for mid-air hand gestures is the domain of robotics. Malima et al. [62] present a system for controlling a robot via a static five-gesture set. The robot interprets user input from a video signal which is segmented according to skin-colored pixels from where the hand palm position and fingers are calculated through the center of gravity. Many problems of this approach are addressed but remain untackled and left to further work. Brethes et al. [10] present a hand gesture recognition algorithm coupled with a face recognition module, both based on the detection of skin colored pixels and contour extraction from video images, with possible applications in the robotics area. Many of the problems as lighting variation and object tracking are addressed coupled with possible solutions for the investigated scenario.

Hand gestures provide various advantages, one of them being the direct interaction possibility, i.e. omitting the need for a glove or an accelerometer. When considering the medical scenario, where an operation needs to be performed, usually the participants are wearing gloves to keep the environment sterile. The systems coming to use so far, as e.g. touch screens, would then be reduced to resistive technologies, as the electricity cannot be carried through the human body when wearing gloves, which can prove to be disadvantageous in many situations. Here, mid-air hand gestures provide an interesting alternative as contactless interaction allows for direct interaction with the hand, without the aforementioned drawbacks. Wachs et al. [123] address the idea of providing a safer and more sterile environment in medical scenarios, presenting a system for hand gesture recognition for improved HMI based on extracting Haar-like features from visual data and classifying them with a fuzzy C-Means Clustering Algorithm. Recognition rates are satisfying - $> 94\%$ - (with minor drawbacks due to training data), however the number of gestures covered by their system is not very large (4 different gestures, 19-20 trials).

Althoff et al. [3] put HGR within the vehicle into the multimodal context - hand gestures are viewed as one possible form of input subordinate to safety-critical functions as steering and braking. An infrared camera is mounted into the roof of the vehicle recording the near-driver environment top-down and a frame-wise evaluation coupled with active lighting to stabilizes the performance in quickly changing environmental conditions. Adaptive thresholding is used to exploit the fact that human skin shows high reflectance under infrared light which raises questions of functionality for users wearing gloves. Forearm segmentation is done by the silhouette calculation and circular cropping. To interpret temporal user input, HMMs are trained and used for testing. Overall, accuracy scores of 86% to 93% are achieved on a 17 gesture set coming from 6 users and a total number of 1530 samples.

1.4.3 3D vision-based approaches

3D approaches for HGR have a number of benefits as easy segmentation capabilities and illumination independence within their area of application. Soutschek et al. [113] set up a complete framework for exploration and navigation through medical 3D-data via hand gestures using a single ToF-camera. While the number of gestures is not very high, recognition rates are satisfying ($> 90\%$) and their study is complemented with a user evaluation showing e.g. that the use of hand gestures in such a scenario is found intuitive and strongly accepted by most of the users.

Opricescu et al. [78] use an SR-3000 ToF camera and propose a system for detecting nine different static hand gestures based on contour calculation from the depth data. Their approach shows an accuracy of over 93% using decision trees however on a fairly small sample set of 50 samples per gesture. Another approach based on depth information only is presented by Chen et al. [18] which introduces region-growing for hand shape analysis, also based on finding the center of a region of interest.

Introducing hand gestures into the car interior comes with a couple of challenges such as dynamic lighting, possibly multiple users or object occlusion to name a few. Kollorz et al.[45] present an approach by implementing a single ToF-camera into the car interior and using the projections of the hand data onto the x- and y-axis as fast and simple features which are classified via a k-NN approach. Simple depth thresholding is used for segmenting the hand from the background. The system's performance is evaluated on a set of 12 gestures coming from 34 persons with a total sample set of 408 samples, and it achieves accuracy scores above 90% with the leave-one-out method.

1.4.4 Hybrid approaches

Hand gesture recognition for a rock-paper-scissors game (also shown to work for a mid-air hand gesture controlled Sudoku game with the same methods [96]) as well as an arithmetic GUI demo was proved to be effectively realizable in real-time on a 10-gesture set with the Microsoft Kinect by Ren et al. [97]. Further work, directed towards the entertainment sector by Zhang et al. [134], combines accelerometer data with multi-channel surface electromyogram (EMG) sensors to interpret user input for a Rubick's Cube game. Again HMMs are used to classify each input into one of 18 possible classes. Gallo et al. [31] present the Kinect as a low-cost off-the-shelf alternative for realizing a scenario where, after a short calibration process, browsing through medical image data is made possible through a set of simple pointing and moving mid-air gestures.

Ohn et al. [75] recently presented a fusion approach combining RGB and depth data tested on a sample set of 886 samples. It is tested on a challenging number of 19 gestures, however split into subgroups for context purposes. SVM classification scores achieve accuracies of above 90%, however these figures drop significantly on cross-subject validation. In other work they also propose a multi-ROI scene observation to be able to better react to unseen behaviour as well as for improved input prediction [73].

More recently, the work of Molchanov et al. [66], motivated by the fact that driver distraction is a major reason accounting for 22% of all accidents in the US, shows how deep networks can be facilitated to make the fusion of three different kinds of sensors - radar, ToF and optical - feasible for dynamic hand gesture recognition within the car. They combine the depth and visual data from the SoftKinetic cam (cf.1.8, bottom right) with a radar sensor serving as input to a convolutional neural network (CNN) [54] and achieving interesting results on a set of 10 different dynamic gestures.

There exist various research approaches employing ToF sensors for HGR systems. ToF sensors have the advantage of being robust vs daylight interferences, but usually suffer from lower resolution and increased noise in the data. Van den Bergh et al. [122]

therefore combine ToF sensors with RGB cameras which increases the performance but somehow contradicts the idea of installing a system being e.g. robust against daylight interferences. Their system needs to be calibrated in order to map depth to color pixels. Hand recognition relies heavily on skin color detection therefore making it again unsuitable in situations in which users wear gloves for example. Lit et al. [58] propose a system working in real-time using a distance transformation of the depth data to classify hand shapes. This system has problems with accuracy as soon as more than just hand pixels are included in the distance transformation (DT) algorithm as finding the center of a region, a crucial point for the approach, is then naturally distorted. Additional fusion of RGB data for potential improvement of recognition is suggested, however this again is a potential noise factor.

As the Kinect was the origin of large body of work conducted in numerous research applications, the most important approaches utilizing this device shall be presented here. One of the first contributions includes the work of Li [57], motivated by developing a framework for sign language recognition or medical applications. A three-step system for detecting the hand and fingers and gesture recognition is presented with satisfying accuracy, however on a fairly simple problem set. The hand is segmented via depth thresholding then reduced to only contain the relevant structure via a convex hull algorithm. The fingertips are identified by their relative distances and classified by predefined feature vectors. Ren et al. [98] present a new metric called Finger Earth's movers distance (FEMD) to compare histograms or signatures for hands, which introduces the novel idea of treating each finger as a cluster instead of the target shape as a whole. While this may be very problem-specific it proves to perform well for the task as results show high accuracy scores for very similar looking poses. The approach is tested on data collected from the Kinect and supposed to work in challenging conditions, but which remains unspecified. Their work is extended to further prove the real-time applicability and high accuracy on a 1000 samples dataset [97]. The work of Biswas et al. [8] attempts to recognize eight different dynamic hand gestures with data recorded from the Kinect. Again, depth thresholding is used to segment the hand from the background while generating normalized grey scale histograms allows for feature extraction from the ROIs. Motion information is extracted by comparing two consecutive frames and subtracting the depth information from the images. SVMs are used to classify the samples which are unevenly distributed, however this approach seems to perform on a satisfying level while it has to be stated that the gestures are rather easily distinguishable.

The work of Raheja et al. [89] demonstrates the finger and palm detection and tracking via Kinect by again using depth thresholding and subsequently applying a circular filter for cutting out the palm of the hand. While this proves to work well, it fails to work for gestures, where the palm is hardly visible in the scenario. Tang [119] proposes fusing the RGB and depth data from the Kinect sensor and compares three different descriptors - the Radial Histogram, the Raw Pixel Estimates and a modified version of SURF - for detecting two gestures - open and closed hand. For the given problem, tested on less than 3000 samples with a trained SVM, SURF is found to perform best. Kurakin et al. [52] present a real-time light-invariant approach with the Kinect working solely on depth data and being purely data-driven. It is tested on 12 dynamic hand gestures from the ASL on a rather low test set. Here, the human body is detected to allow for a first estimate of the position of the hand. The center of the hand blob serves for calculating its silhouette and HMMs are used to perform the dynamic HGR task. Although the approach seems promising due to its independence of complex models and RGB data, it is difficult to assess its validity as each gesture is performed only three times by ten subjects.

Doliotis et al. [23] also fuse RGB and depth data from the Kinect based on the dynamic time warping algorithm (DTW) which seems to perform well, however only when it can be assumed that the user faces towards the sensors. Van den Bergh et al. present an HGR via Kinect and apply it to give directional commands for a robot. They build upon an existing hand detection and tracking algorithm from the OpenNI and present a method for robust arm wrist cropping coupled with a hand gesture recognition method based on intensity and grey scale values exemplary tested on four different poses. Patsadu et al. [81] demonstrate how human gestures can be classified efficiently with the help of the Kinect and present a study of various classifiers amongst which the neural networks trained with backpropagation seem to perform best on a set of 3600 samples, followed by SVMs, decision trees and Naive Bayes.

Yao et al. [132] further demonstrate the range of applicability of the Kinect sensor applied to AR frameworks which, although aided by a coloured glove, shows to what extent contour models can be used for the HGR task. As Caputo et al. claim [14] the Kinect's ability of recognizing hand gestures is sufficient for ranges of up to 1m, hence they propose coupling it with another sensor - here, a webcam - to increase its applicability range. Multiple Kinects are positioned in an orthogonal angle and used for initial tracking. The high-resolution camera is used for distinguishing the fingers.

As the authors claim, this is not robust and thus a coloured glove is used to further boost detection performance.

1.4.5 In-vehicle gesture recognition systems

In an early review of gesture-based HMI for vehicles Pickering et al. [86] find it is necessary to identify the functions suitable for HGR systems in order to alleviate the adaption of such systems to the given environment. Not every function within the car is suited for such a control system, even when referring to the infotainment systems alone. Furthermore, according to their findings, the main challenges for vision-based approaches - the ones researched the most - lie in developing a system workable under dynamic background, changing lighting conditions, robustness and real-time capability. They find user acceptance to be mixed toward this technology indicating that HGR systems have - at the time of writing - not yet permeated the general driver's awareness. Further works of interest include [87] [82] [84] being directed towards HMI concepts such as Natural User Interfaces (NUI) within the automotive environment. This topic, of creating an NUI within a vehicle, is addressed later on in the own work chapter in more detail. The work of Ohn-Bar et al. [74] [75] suggests using a multi-stage approach. The vehicle interior is divided into subregions to first determine in which subregion the hand is found and then conclude whether or not there is an object in one of the hands. The dataset consists of RGBD data recorded with a Kinect comprising 19 different hand gestures. The classification step is realized via SVMs with recognition results peaking when RGB and depth data is combined (up to 95% for certain gestures). While their system works in real-time on a challenging dataset, as many parts of the hand are often occluded, the issue of varying lighting conditions is left to further research. The work of Parada et al. [80] compares the use of a touch-based interface to a gestural interface with a survey on 23 volunteers. To their findings, while the touch-based interface is perceived as more reliable and easier to use, the hand gesture interface is found to be more useful, less distracting and more secure. Moreover, the users found a gestural interface more worth buying, perceiving it very pleasant during the interaction phase.

1.4.6 Conclusion

To sum up, this section provided an extensive overview over the most important works of research conducted for the topic of HGR, with a focus on vision-based approaches as they allow for seamless interaction. First of all, it can be concluded that there is no single approach being able to deal with all the challenges described in this section. Most of the solutions rely on RGB data for detecting human body parts through skin-colored pixels, therefore being unable to deal with challenging lighting conditions. This is also the major drawback for employing the Kinect sensor, as it is only able to provide reliable depth data under controllable environmental scenarios, ruling it our for e.g. in-vehicle use. First work conducted with ToF sensors demonstrates the advantages of providing data under high frame rates and direct sunlight influence. The fairly low resolution is one of the main concerns requiring more attention, however, it becomes obvious that tasks as segmentation can be easily performed with 3D-capable cameras.

As for the algorithmic part, most approaches rely on various classifiers trained for the task, without being able to discern a certain preference for a single approach standing out in terms performance. HMMs seem to be employed frequently when it comes to dynamic hand gesture recognition, although it is difficult to assess their performance as often either the sample set or the number of gestures are fairly small.

To account for the possibly noisy data, sophisticated 3D-algorithms have been tailored towards the task, as presented in the following section. Finally, the first part concludes with an overview of the most relevant ML approaches (Section 3) for this work. Further emphasis is put on the topic of multiclass classification as the core algorithms rely on the very fact that multiple classes are to be distinguished.

Chapter 2

3D data algorithms

This section first provides a short introduction into typical applications of 3D data processing and presents the necessary background for understanding how depth data can efficiently be analyzed and processed for the task of hand gesture recognition. A depth sensor provides the surrounding environment in the form of x-y-z coordinates, termed point clouds (PC). Depending on the sensor employed, depth data can vary in precision and noise (e.g. the outliers in data). Finding good features in data of this form, i.e. a low-dimensional description of characteristic properties of an object, is a challenging task. of To this end, so-called descriptors are introduced in Section 2.3 with regard to creating an instrument which describes an object in a fast, robust and precise manner.

2.1 Introduction

In order to make sense from depth data obtained from one or multiple sensors various efficient algorithms have been developed and extended. The Point Cloud Library (PCL) is an open source project initially released in 2011 for image processing in 2D and 3D [107].

Figure 2.1: The point cloud library (Source: [104])

Typically, the field of application for 3D-algorithms is embedded in the area of HMI, robotics or Computer Vision in general hence the requirement for PCL applications is to perform efficiently in real-time.
Rusu et al. [105] present, how models of task-relevant objects within indoor environments are acquired. Further indoor applications include indoor mapping of rooms

[108] or indoor scene segmentation and object recognition (Caron et al. [15]). Data in e.g. the latter case comes from a Kinect sensor mounted to a mobile robot.
Numerous applications have been developed in the field of robotics. Rusu et al. [106] introduce a descriptor for point clouds, and the authors demonstrate how it can be utilized for object recognition and pose detection for tasks such as object manipulation by robots in real-time. Kehoe et al. of [43] show applications in the closely related field of cloud based grasping and more specifically utilizing the pose estimation algorithm *iterative closest points* for pose estimation.
Further applications are in the field of e.g. object recognition [2], [118], person tracking [67], [19], facial recognition with a Kinect sensor [36] or hand pose estimation [46].

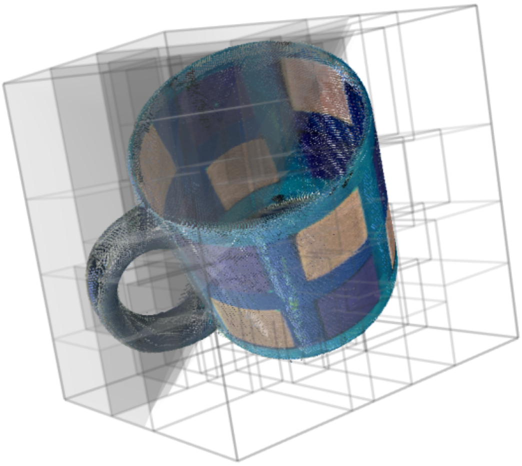


Figure 2.2: The KdTree structure in 3D, dividing the search space into further subspaces (Source: [104]).

As the name states, the main purpose of PCL algorithms lies in the processing of point cloud data, i.e. data obtained from a 3D-sensor and usually representing an object or a scene in the form of either structured or unstructured x-y-z-coordinates. As 3D-data typically is high-dimensional, there exist algorithms for spatial partitioning, downsampling and search operations on the point cloud data set such as the *octree* or *KdTree* algorithms (see Fig.2.2).

In order to be able to distinguish individual objects in a scene, algorithms for segmenting the individual parts are utilized, as can be seen in Figure 2.3. Here, the point cloud containing the data points of a mug standing on a table is segmented, visualized by the red and green colors respectively. As the algorithm recognizes planes and cylinders the white artifacts in the image refer to unmatched point data and noise.

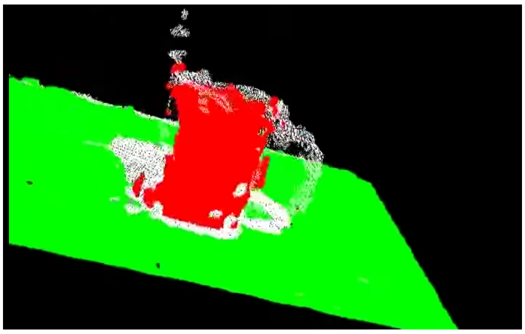


Figure 2.3: Segmentation of a mug from the table in 3D (Source: [104])

Finding good features of a target object is a crucial task when it comes to object recognition. There exist various approaches with different degrees of complexity depending on the specific task or the size of the data. Figure 2.4 shows the same scene recorded from two different angles at two different points in time t_1 and t_2. Due to variations in sensor position, sensor orientation or simply noise the question arises, how to find corresponding points in 3D within both images.

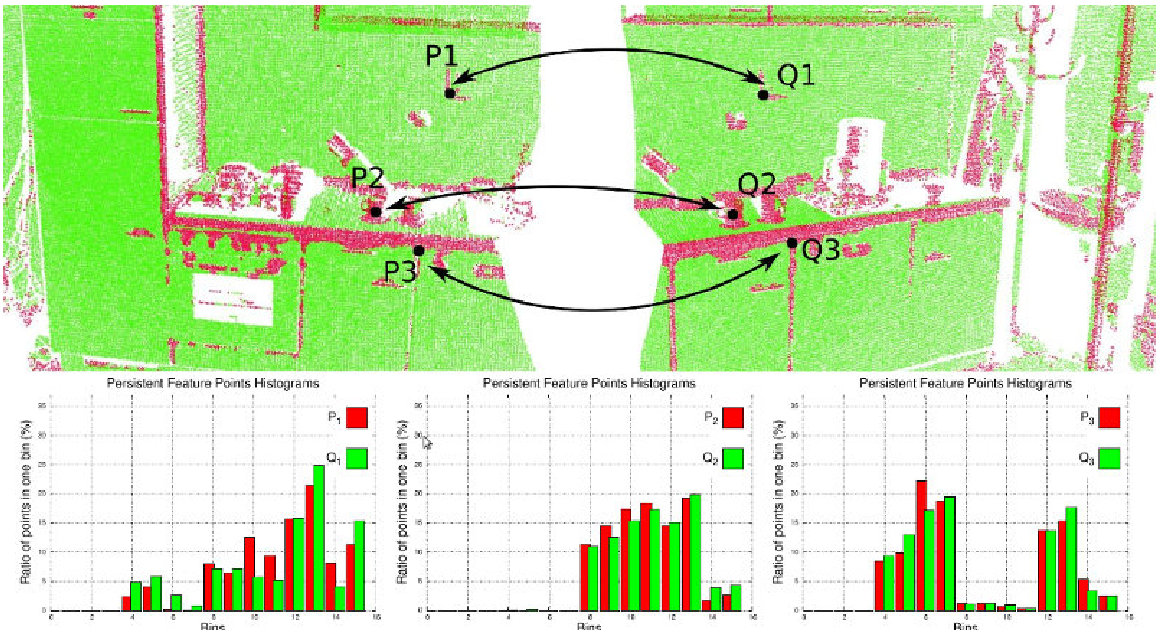


Figure 2.4: Features in the same scene, represented by various histograms. (Source: [104])

A simple euclidean distance metric does not provide enough security as for instance depending on the precision of the sensor the number of candidates for matching pairs can vary significantly. This problem is depicted by the mapping of corresponding points $P_1 - P_3$ to $Q_1 - Q_3$. A primitive distance metric is not sufficient, therefore

point features (or **point feature representations**) have been introduced in the PCL. Another common term is that of a **(local) descriptor** as it introduces a new concept, namely the description of a point via possibly multiple metrics. In contrast, a **global descriptor** describes the point cloud as a whole. In general the quality of a feature is measured whether it retains its descriptiveness under rigid transformations (i.e. translations, rotations), under varying sample density (how much information is available) and its robustness vs. noise. Typically, features are collected and represented via a histogram as can be seen in the lower part of Fig. 2.4. *Similar* features in this example are represented by *similar* histograms and are compared with an adequate comparison measure.

2.2 Estimating surface normals in a point cloud

For a point p_i in the cloud and its normal n_i if the following equation holds

$$\vec{\mathbf{n}}_i \cdot (\mathsf{v}_p - \mathbf{p}_i) > 0 \tag{2.1}$$

where v_i is the viewpoint towards the scene, then the orientation of the normal can be determined as pointing towards the viewpoint.

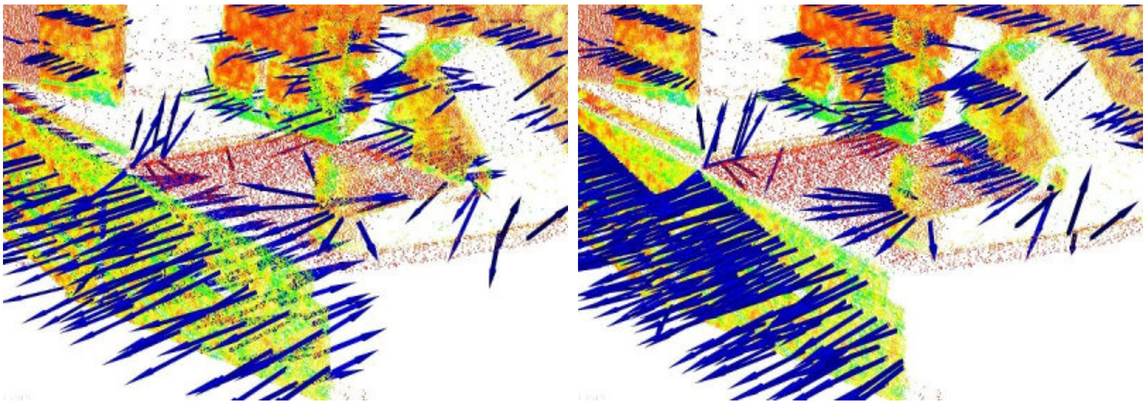


Figure 2.5: Normals of a point cloud. If the viewpoint is not known, normals appear on both sides of a plane (left). The same scene disambiguated with normals on the *correct* side if the planes. (Source: [104])

Depending on the problem at hand the question of calculating the normal of a point depends on the so-called **k-neighborhood** or the search radius **r** which is determined by the search algorithm or the radius of the sphere referring to the query point respectively. The normal of a point is defined as standing perpendicular on a plane tangent to the query point, hence choosing the number of neighboring points which define the plane is crucial as a **k** or **r** chosen too small may not represent the surrounding correctly (e.g. loss of detail) while choosing the parameter too large leads to distorting artifacts, as surrounding areas have too much influence on the calculation. Therefore, the parameter has to be chosen problem-dependent and is determined via selected experiments. Another crucial factor, especially when aiming at real-time capable applications is the calculation cost scaling with increased **k** (or **r**) and depends furthermore on the point cloud density, i.e. the sensor's resolution. The following section seals with the variety of descriptors and their relevance for this thesis.

2.3 3D shape descriptors

In order to be able to describe the geometry of an object, **local** and **global descriptors** are introduced. A local descriptor catches the geometry of a single point within a point cloud and accordingly a global descriptor describes the point cloud as a whole. The latter are also denoted as **holistic descriptors**. A number of descriptors, each demonstrating different kinds of properties, has been introduced to describe objects or scenes in 3D, of which only the most relevant shall be described in this section.

A commonly known descriptor is the *Signature of oriented Histograms* (SHOT descriptor) introduced by Tombari et al. [120] which calculates multiple histograms in a locally defined reference frame and combines these to one descriptor. The *normal aligned radial feature* (NARF descriptor, [116]) attempts to capture the geometry of a single point via its normal, initially calculated from the viewer's perspective. The local geometry is defined by a so-called star pattern capturing the positions of the surrounding points in a histogram. Spherical harmonic invariants [13] as well as spin images [39] have been introduced early on as robust descriptors for 3D shapes, performing well when cloud data is dense and less noisy (cf. del Bimbo et al. [7]), preferably coming from laser scanners. Data obtained from sensors as the Camboard Nano is potentially highly noisy and has a comparably low resolution thus spin images are unsuitable. Curvature maps introduced by Gatzke et al. [32] describe the curvature of the object by a series of meshes defined on the surface. However, they also suffer from noisy data acquired by sensors typically coming to use in HMI or robotics and generally perform best when data is densely sampled. In order to solve complex HMI tasks such as hand gesture recognition for real-time applications one is confined to rely on fast ToF sensors (as the Camboard Nano), for the purpose of real-time applicability as well as portability due to its low dimensions. These benefits usually come at the cost of resolution and noise thus describing the scene under such circumstances poses additional basic prerequisites - in this case the robustness, descriptiveness and computational complexity of the descriptors. This section presents the functionality and computation of the descriptors coming to use along with its main advantages and drawbacks. These descriptors are essential for the core algorithms introduced in this work, i.e. for improving hand gesture recognition with neural networks.

2.3.1 Point Feature Histograms

The idea of the Point Feature Histogram (PFH) is to capture the geometry of a point in a point cloud in relation to its neighbors. This relation is described via the definition of the normals of the points in question. To this end the relation between two points p_s and p_t (source and target point respectively) can be computed by defining a so-called darboux frame - a generalization of an ordered basis in a vector space - at one of the points. For two points p_s and p_t the corresponding normals are defined as n_s and n_t. The point for the darboux frame to be fixed at is determined by comparing the angle of the each of the normals with the connecting line $\vec{p}_{st}$. Thus if $arccos(\vec{n}_s \cdot \vec{p}_{st}) \leq arccos(\vec{n}_t \cdot \vec{p}_{ts})$ the darboux frame is fixed at point p_s, the point with the smaller angle between its normal and the connecting line.

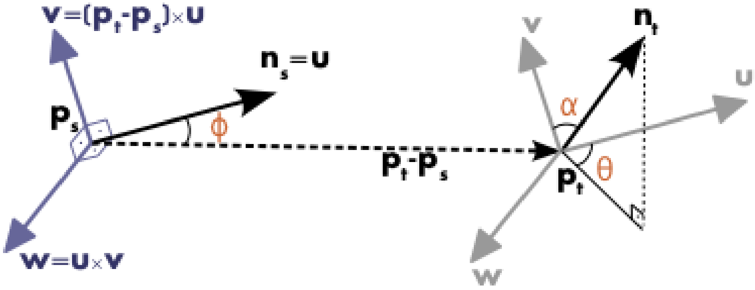


Figure 2.6: The darboux frame at the source point p_s. This image depicts the positioning of the two points p_s and p_t with respect to each other. (Source: [104])

The axes of the coordinate system allow us to compare the positioning of the point in relation to a target point and its normal.

$$\begin{aligned} \mathsf{u} &= \mathbf{n}_s \\ \mathsf{v} &= \mathsf{u} \times \frac{(\mathbf{p}_t - \mathbf{p}_s)}{\|\mathbf{p}_t - \mathbf{p}_s\|_2} \\ \mathsf{w} &= \mathsf{u} \times \mathsf{v} \end{aligned} \tag{2.2}$$

This relation is depicted in Figure 2.6: The darboux frame is spanned on the source point p_s with its corresponding normal n_s. The axes **u**, **v** and **w** spanning the frame are labeled on both the target and the source point (p_s and p_t respectively) - this gives a better visualization of the relationship between the two points in terms of the angles:

$$\begin{aligned}\alpha &= \mathsf{v} \cdot \mathbf{n}_t \\ \phi &= \mathsf{u} \cdot \frac{(\mathbf{p}_t - \mathbf{p}_s)}{d} \\ \theta &= \arctan(\mathsf{w} \cdot \mathbf{n}_t, \mathsf{u} \cdot \mathbf{n}_t)\end{aligned} \tag{2.3}$$

with **d** being the Euclidean distance between the two points p_s and p_t, $d = \|\mathbf{p}_t - \mathbf{p}_s\|_2$. These angles are sometimes referred to as pan, tilt and yaw angles respectively.

For a single point its **k**-neighborhood is defined. Then, for all pairs in this neighborhood the quadruplet $\langle \alpha, \phi, \theta, d \rangle$ is calculated and binned into a 125-dimensional histogram. The resulting computational complexity of the PFH descriptor for a point cloud of size n is $\mathcal{O}(nk^2)$ as the quadruplet has to be computed for all pairs in a k-neighborhood of a single query point. To make a step further towards a real-time applicable feature calculation the Fast Point Feature Histogram is introduced in the following section.

2.3.2 Fast Point Feature Histograms

In order to reduce the computational complexity of the PFH and make feature calculations real-time applicable the Fast Point Feature Histogram (FPFH) is introduced. It maintains the descriptive power of the PFH by reducing the number of calculations for the local curvature just enough to capture the main features of the surrounding of a query point p_q. It is computed as such:

$$FPFH(\mathbf{p}_q) = SPFH(\mathbf{p}_q) + \frac{1}{k}\sum_{i=1}^{k} \frac{1}{\omega_k} \cdot SPFH(\mathbf{p}_k) \tag{2.4}$$

where the SPFH is the simplified PFH - the set of tuples α, ϕ, and θ computed as described in Sec. 2.3.1. For every point k in the neighborhood of p_q the SPFH is also determined and added with a weighting parameter ω_k to the SPFH of the query point thus creating a weighted scheme of the surrounding. The computational complexity is reduced to $\mathcal{O}(nk)$. The results are binned in a 33-dimensional histogram.

2.3.3 The ESF descriptor

The Ensemble of Shape Function (ESF, see Wohlklinger et al. [130]) is another global descriptor which is however not geometric as it does not rely on the calculation of the normals.[1] The resulting descriptor consists of ten histograms each itself comprising 64 bins. In an initial step 20000 points are sub-sampled from the cloud. Now in turn for each iteration, three points are randomly sampled from the first step and four measures are calculated. The D2 measure checks whether the points on the line connecting two of the sampled points lie inside or outside the recorded surface or both. This 'convexity'-measure is binned into a histogram. The D2 ratio measures the ratio of these lines lying on the cloud or free. The D3 ratio calculates the square root of the area spanned by the sampled points and again checks for the position of the area relative to the cloud. Lastly the A3 measure calculates the angles between the three points and creates the histogram in an analogous manner to the other measures. Although this descriptor does not rely on any normal information it represents the general shape of the data while retaining its global descriptive power.

2.3.4 The VFH descriptor

The Viewpoint Feature Histogram (VFH) was introduced by Rusu et al. [106]. The idea is to maintain the descriptive power of the FPFH (cf. Section 2.3.2) while adding viewpoint variance and remaining invariant to scaling. The resulting histogram consists of two components - the viewpoint component and the extended FPFH component, i.e. two histograms concatenated to a single one. The viewpoint component is calculated by computing the angles the viewpoint direction makes with each normal of every point in the data set. It has to be noted that this is not the direction **to** each normal, but the viewpoint direction translated to each point (and its normal), since this retains the invariance to scale. The second component calculates the pan, tilt and yaw angles as described before between the viewpoint direction at the centroid of the cloud and every normal in the cloud (see Figure 2.7 for visualization of the VFH).

The resulting histogram is of size 308 with 128 bins for the viewpoint component and $3 \cdot 45$ bins for the angles. Time complexity is reduced to $\mathcal{O}(n)$ thus resulting in a real-time applicable global shape descriptor. The clustered Viewpoint Feature Histogram (CVFH) from [1] adds another 45 bins for the *Shape Distribution Component* to distinguish planes with similar normal distributions in order to reconstruct shapes

[1] This section appeared with slight modifications in Kopinski et al.[47].

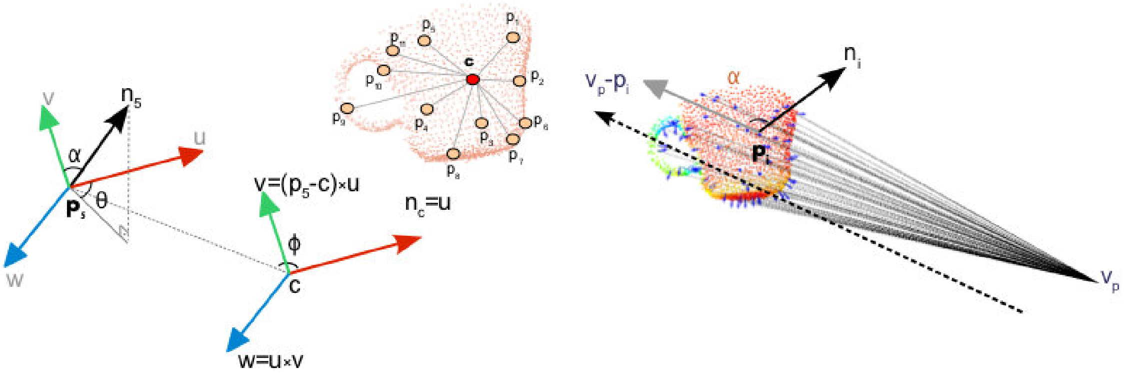


Figure 2.7: Calculation of the VFH with the first component (left) and the second component (right). (Source: [104])

of objects partially occluded by other objects. This is however not necessary for the set goal, since the VFH proves to perform satisfactory as experimental results show later on.

2.3.5 The randomized PFH descriptor

The randomized PFH descriptor (RPFH), as introduced by Caron et al. [15], is an extension of the PFH descriptor described aforehand. It relies on the computation of the three angular features as well as the distance measure. For two randomly selected points from the cloud these features are calculated, normalized and binned into a 625-dimensional histogram capturing the curvature of the point cloud. This is repeated for a maximum of 10,000 points in order to limit computational complexity of the histogram. Point clouds for the task of hand gesture recognition focused on in this work usually vary between 200 - 3000 points, thus a limit of 10000 randomly subsampled points (or 5000 point pairs) captures the geometry of the cloud well.

2.4 Conclusion

This chapter provides an understanding for the 3D algorithms employed, building the basis for the HGR framework developed in this thesis. In order to describe a point cloud, a variety of histograms have been analyzed and presented. These histograms are either based on the calculation of the normals of a certain point in the cloud (or the cloud as a whole) or are based on describing the shape via a convexity measure. The analysis conducted for this thesis will explain the advantages and disadvantages of the chosen descriptors in terms of recognition accuracy and real-time performance - the two crucial factors for developing an HGR framework.

The next section delves into the topic of Machine Learning and Classification to provide the necessary basics for the HGR algorithms in this thesis.

Chapter 3

Machine Learning and multiclass classification

3.1 Object recognition

The goal of this thesis is to improve Object Recognition techniques with a special emphasis on multiclass classification with Neural Networks. This is a very general definition as Object Recognition performed by Machine Learning algorithms is a popular topic with a plethora of applications in various fields as Human-Machine Interaction, robotics, face and person detection or Computer Vision in general. The task at hand is embedded in the field of HMI, more specifically in the field of hand gesture recognition, however the proposed methods for improving multiclass classification can be transferred to any other field of research dealing with MCC.

When restricting the task to identifying objects within data streams coming from any kind of sensor, different kinds of problems arise as partly occluded objects, different viewpoint angles or the problem of the same object being scaled, translated or rotated within a scene. To deal with these kinds of problems various ML approaches are proposed and thoroughly examined, hence the following chapters provide the necessary overview of the most popular ML algorithms being used in research and industrial applications with a focus on providing the preliminaries for understanding the research conducted in this thesis. These ML approaches belong to the set of *Supervised Learning* techniques where a (large) number of samples is being presented during a *training phase* while a certain number of samples is retained for the *testing phase* to test the performance of the trained model. If a model predicts the correct class for a large number of unseen samples - i.e. samples not used during training - it is said to *generalize* well. Models showing a good performance on samples from the training set but performing badly on unseen samples have learned the problem by hard due to *overfitting*.

The following sections provide the theory of support vector machines, neural networks as well as a short outline on Deep Learning (DL). Building upon this, the possibilities of handling the task of multiclass classification with the ML approaches along with the occurring advantages and disadvantages are discussed in the final section.

3.2 Introduction to Machine Learning

3.2.1 Support vector machines

Support Vector Machines (SVM) - also termed *Large Margin Classifiers* - are a popular alternative amongst the large family of classifiers. They aim to find an optimal separating hyperplane, with a maximum distance to its nearest training points (hence the large margin) in order to achieve optimal generalization performance. Only some of the training samples are needed in order to define this margin, these samples are called the *support vectors*.

For a set of training data $\{\mathbf{x}_i, y_i\}$ $i = 1, \ldots, l$, $\mathbf{x}_i \in \mathbb{R}^n$, and $y_i \in \{+1, -1\}$ each sample $\mathbf{x}_i$ taken from n-dimensional feature space and is assigned a label +1 or -1 belonging to either of the two classes. This can be represented by the Equations 3.1 and 3.2 respectively and can be combined to define the optimal separating hyperplane in Equation 3.3

$$\mathbf{x}_i\mathbf{w} + b \geq +1, \quad y_i = +1 \tag{3.1}$$

$$\mathbf{x}_i\mathbf{w} + b \leq -1, \quad y_i = -1 \tag{3.2}$$

$$y_i(\mathbf{x}_i\mathbf{w} + b) - 1 \geq 0, \qquad \forall y_i \tag{3.3}$$

for the linear separable case. Here, the points taken from the training data are the support vectors. However, real-world problems tend to not have data presented in a linear separable way, thus the notion of finding a hyperplane which separates the two classes by determining the maximal margin is extended to allow for misclassifications to occur. Slack variables ξ_i are introduced to penalize misclassifications and hence Equations 3.1 and 3.2 are extended to

$$\mathbf{x}_i\mathbf{w} + b \geq +1 - \xi_i, \quad y_i = +1 \tag{3.4}$$

$$\mathbf{x}_i\mathbf{w} + b \leq -1 + \xi_i, \quad y_i = -1 \tag{3.5}$$

respectively. The whole problem becomes an optimization task to minimize the term

$$\arg\min_{\mathbf{w},\xi,b} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i \right\} \tag{3.6}$$

If we allow for nonlinear decision planes the **kernel trick** is applied. Here we take points from $\mathbb{R}^d$ to some space $\mathcal{H}$:

$$\Phi : \mathbb{R}^d \to \mathcal{H} \tag{3.7}$$

Choosing the kernel function K such that

$$K(x_i, x_j) = \Phi(\mathbf{x}_i)\Phi(\mathbf{x}_j) \tag{3.8}$$

and thus the mapping of the training data is transformed to a higher-dimensional space, becoming linear separable.

3.2.2 Neural networks

Neural networks (NN), also termed artificial neural networks (ANN), are function approximators stemming from the biologically inspired neural networks in the human brain where millions of neurons, responsible for information processing, are interconnected via synapses, responsible for transmitting information. This pattern was the inspiration to build the theory ANNs in order to create ML algorithms for mimicking intelligent human behaviour in e.g. object recognition tasks.

Neurons are represented by nodes, synapses by weights connecting the neurons with each other. Neural networks usually consist of a number of nodes in an input layer, in one or more hidden layers and, depending on the number of classes, nodes in an output layer. Depending on the kind of network, different kinds of connection patterns exist for the connections between the neurons. If information is processed layer-wise from input to output layer, i.e. the nodes (neurons) form a directed acyclic graph via the directed edges, this NN is called a feedforward neural network. If feedback edges exist between neurons in a layer of higher hierarchy to neurons in a layer of lower hierarchy, these NNs are termed recurrent neural networks. In the underlying thesis, only fully connected feedforward Neural Networks come to use in order to demonstrate the functionality of various fusion algorithms. The weights, learned during the learning process, describe potentially multiple hyperplanes splitting the search space. These networks are called Multilayer Perceptrons (MLPs); this terminology is mainly used for the rest of the underlying work.

Except for the input layer, each neuron has a nonlinear activation function. Typical choices are $tanh(x) = (e^x - e^{-x})/(e^x + e^{-x})$ or the logistic sigmoid function $f(x) = 1/(1 + e^{-x})$. Training of an MLP aims at determining the network's parameters and is achieved via the *Backpropagation* algorithm [102],[126]. The Backpropagation algorithm is a gradient descent method minimizing a sum-of-squares function. For a set of training inputs $\mathbf{x}_n$ with $n = 1, ..., N$ with a corresponding set of target vectors $\mathbf{t}_n$ minimize the error function

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} ||\mathbf{y}(\mathbf{x}_n, \mathbf{w}) - \mathbf{t}_n|| \tag{3.9}$$

with respect to the weight matrix $\mathbf{w}$ of the MLP.

The procedure is as follows:

1. Feed the input vector $\mathbf{x}$ into the MLP and forward propagate the input through the network to calculate the activations of the neurons in all hidden layers
2. Calculate the output vector $\mathbf{o}$, i.e. the activations of the output neurons
3. Calculate the error of the output vector
4. Backpropagate the error for each output unit through the hidden layers

Since the output of the net depends on the weights between the layers, finding the global minimum of the error function is achieved by taking the partial derivative of E with respect to the weights

$$\frac{\partial E}{\partial w_{ij}} \tag{3.10}$$

and by application of the chain rule this equals

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial o_j}\frac{\partial o_j}{\partial \mathrm{net_j}}\frac{\partial \mathrm{net_j}}{\partial w_{ij}} \tag{3.11}$$

However, two cases have to be distinguished when adapting the weights with

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} = \eta \delta_j x_i \tag{3.12}$$

because

$$\delta_j = \begin{cases} \varphi'(\text{net}_j)(t_j - o_j) & \text{if } j \text{ is Neuron in output layer} \\ \varphi'(\text{net}_j) \sum_k \delta_k w_{jk} & \text{if } j \text{ is Neuron in a hidden layer} \end{cases} \tag{3.13}$$

Hence the term *Backpropagation* - once the input is presented to the net and propagated forward, the nets output can be determined, the error calculated and the weight adaptation can begin, propagating the error back through the network.

3.2.3 Deep Learning

Deep Learning has emerged as a subfield of ML due to impressive results on problems in the field of Computer Vision or Text Recognition. Definitions around this term vary, however when relating it to Neural Networks one generally speaks of an MLP containing multiple hidden layers with nonlinear operations, i.e. activation functions. Deng et al. [22] provide, at the time of writing, an extensive and up-to-date overview over the current state of the art methods along with a historical overview of Deep Learning. Having been abandoned in the 90s due to being outperformed by the SVMs, deep nets have undergone a renaissance since they were successfully employed in a number of problems [35] [5]. The main inspiration is derived from the architecture of the brain being a deeply connected network itself, but due to the aforementioned performance increase from 2006 on, deep architectures have gained increased attention. The main goal is to learn *feature hierarchies* i.e. learning a complex set of features out of sets of low-level features, as is shown exemplary in Figure 3.1.

Here, higher-order features as e.g. shapes or edges can be generated by features lower in the hierarchy, i.e. the greyscale values on the lowest level.

There exist various forms of Deep Learning architectures but Convolutional Neural Networks (CNNs) have perhaps gained the most attention due to their successful application in image recognition tasks. CNNs work on a two-dimensional input array of data and consist of multiple convolution layers. In each of the layers multiple kernels are used to extract specific features from previous layers, thus creating increasingly abstract features with each layer, which are subsequently reused, creating very complex convolutions. The extracted features are compressed after every layer,

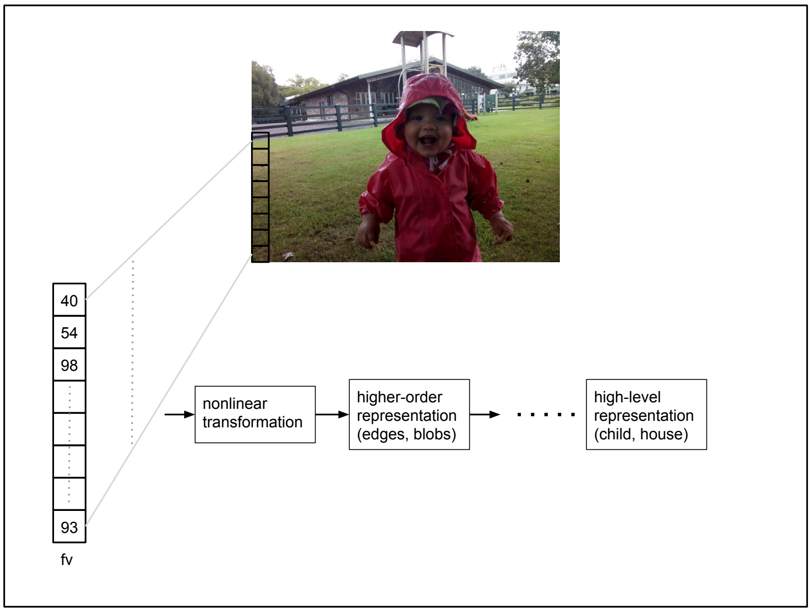

40
54
98
93
fv
nonlinear transformation
higher-order representation (edges, blobs)
high-level representation (child, house)

Figure 3.1: Learning higher-order features from low-level features or feature vectors (fv).

reducing the amount of data while increasing its significance in relation to the problem. After several convolution layers, an MLP is used to identify the final result. The major advantage of this approach is the ability to work directly on the image data with very abstract definitions of features, while not defining special features to be detected, but instead leaving the setup of these features to the training algorithm. Furthermore each layer consists of many copies of the same trained kernel, which allows the network to detect the trained features even on large scale images amongst much noise and unclassified data. This robustness makes this kind of classifier especially useful in scenarios where objects have to be detected in a dynamic scene, like a gesture in front of a background. For this objective, it is also possible to duplicate the output layers and increase the size of the input data on a network that previously learned on a smaller scale to detect the learned object in large input data, without the need to recalculate all layers, as the kernels share the same weights on each layer. This approach of learning in multiple layers, increasing the complexity of calculated features in each layer, is what defines Deep Learning. The crucial point about Deep Learning is, that unsupervised learning methods are used on very flexible algorithms, like Convolutional Neural Networks. This allows the algorithm to learn pretty much every mathematical description of the original input data, which results in a very simple implementation of the basic structures that is able to imitate the work of much more complex supervised algorithms. For the underlying work, the 3D images have been transformed into 2D representations and CNNs have been used to classify the multiclass problem of hand gesture recognition from ToF sensor data. The results are presented in the Outlook section of this thesis in order to compare the presented approaches with state-of-the-art ML algorithms and to outline future work in this direction. To this end, the implementation by LeCun et al. [54] was facilitated and adapted to the task.

3.3 Introduction to multiclass classification

3.3.1 Multiclass classification with support vector machines

SVMs are binary classifiers therefore need to be adapted to the multiclass case by decomposing it into a series of binary classification problems. Two strategies are typically followed, the one-against-one (OAO) and the one-against-all strategy (OAA). A simple OAA approach is the winner-takes-all strategy where the classifier with the highest value from the output function is the one to determine the class. OAO is typically realized by a voting scheme - each of the $\frac{n(n-1)}{2}$ classifiers decides for a class. The class voted for the most is the one determined for the current instance.
Crammer and Singer [21] propose a an approach where the multiclass classification problem with SVMs becomes a single optimization problem.

Schoelkopf and Smola [111] argue that when approaching the problem with a OAA scheme by training all pairwise combinations of binary classifiers and using the *winner-takes-all* approach the scores coming from the classifiers are somewhat *heuristic* which is problematic as they were trained on separated binary problems thus raising the question of comparability of the scores. It becomes more complex when introducing a threshold Θ to measure the *confidence* a certain classifier must surpass in order to assign the sample to a certain class. If e.g. none of the classifiers are confident enough and their outputs are not on comparable scales the question of how to determine the class becomes more problematic. They also mention the problem of *asymmetry* for the OAA scheme as a binary classifier trained in this way learns from a many more negative than positive samples which can of course be regularized by a constant to adjust the ratio between positive and negative samples.
They argue in favour of pairwise classification, i.e. the OAO scheme. This increases the training time, however the number of training samples for each problem is significantly smaller than for the problem as a whole. In terms of execution time the reduced number of support vectors, due to smaller training samples, and the reduced problem complexity (due to less class overlap) also reduces the execution speed. It is

also noteworthy that evaluating $\frac{n(n-1)}{2}$ can be slower as e.g. two classes which already scored low on several classifiers need not be compared to each other. This can be taken into consideration when the cascade of classifiers is coded properly. They furthermore argue that approaching the problem in one take, i.e.modifying the objective function of the SVM so that the multiclass classification task can be addressed directly is the idea which is mostly in the sense of Vapnik's notion of solving the problem directly. While the results seem comparable at times with OAA scheme the single optimization problem needs to deal with all SVs at once thus being very disadvantageous in terms of training time. Generelizations to *multi-label* problems, meaning that a sample can belong to multiple classes at once, are mentioned [24], however they are beyond scope of this discussion and of no practical value for the contribution of this work. In general they argue that a simple OAA approach produces acceptable results bringing all the mentioned advantages while referencing that it always comes back to choosing the appropriate scheme problem-depending [111] .

If the underlying are classifiers well-tuned and regularized, Rifkin and Klatau [99] argue in favor of the OAA voting scheme within which $n =$*number of classes* are trained and the one with the highest value in the output function determines the class. Moreover they show that the two most popular approaches, namely the *single classifier*-approach where a single optimization problem has to be solved and the error correcting approach where a collection of binary classifiers is trained and combined, do not necessarily outperform the OAA scheme. They also compare their findings to the all-against-all (AAA) scheme in which $\binom{n}{2}$ binary classifiers are trained, each separating a pair of classes and - although training time can be reduced when coded accordingly - conclude that it does not outperform OAA in terms of classification rates on reference data sets. They argue for *controlled* experiment runs within which the hyperparameters of the algorithms are carefully chosen without adapting them too much to the training set or the task at hand (mining the test set) as it easy to show that methods tailored to combine a *weak* set of binary classifiers easily improve the overall result. The main argument in favor of OAA remains that, opposed to popular opinion, this approach can perform just as well as the other approaches while maintaining its simplicity in terms of computational complexity and conceptual modeling. Their extensive experiments demonstrate that a simple OAA scheme can achieve adequate performance compared to complex error-correcting-schemes or complex single machine optimization problems. They also address the problem of training speed which shows that an OAA scheme i trained significantly more slowly (up to six times more slowly) than an AAA scheme for their largest data of 15,000 data points.

This work takes these findings into account as the improvement of HMI techniques requires significant training and parameter tuning in order to achieve satisfactory performance for the core ideas developed.

An extensive overview on various data sets comparing the OAO, OAA and directed acyclic graph (DAG) is presented in [37]. The DAG approach basically is an AAA approach with a directed acyclic graph being used for finally determining the membership of the class. Their findings, tested on various data sets containing a a few hundred up to 40,000 samples show that the OAO and DAG methods are more suitable for practical use while the more detailed results also indicate there the OAA approach can also contest although training times naturally are smaller for the former two, especially if the data set is small.

Platt et al. [88] present the Decision Directed Acyclic Graph (DDAG) method and compare it to the standard algorithm as well as the max wins approach showing that accuracy is maintained while increasing training time. They claim that the OAA approach of training n SVMs, with the highest class determined by the highest output values of all SVMs, has no bound on the generelization error. The OAO approach was shown to be Bayes-optimal [30]. They claim that the overall classifier in the OAO approach tends to overfit if the classifiers are not well regularized (which is the case for SVMs). The max-wins approach has no bounds on the generalization error and the OAO scheme grows superlinear with the number of classes. They show on three different data sets that their method stays competitive, however for a problem of 7 classes, trained on 11340 samples and tested on 565,893 samples a classification error of 29% is achieved.

Foody et al. [27] compare the performace of discriminant analysis, decision trees, feedforward neural networks to an OAA-SVM approach by arguing that a single optimization strategy requires fewer support vectors to be determined and the choice for parameters C and γ needs to be determined only once. The decision function for the OAA-SVM is therefore $\text{argmax}_{m=1,\ldots,n}(\mathbf{w}_m^T\phi(x_i) + b_m)$. The parameter choice for the feedforward NNs is, though not precisely mentioned, determined by selecting from several hundreds of candidates. Evaluation is performed on a visual image data set and results show that the SVM approach outperforms the NNs by approx. 2% however the data set is very small (100 samples for 6 classes) thus the findings are difficult to assess.

SVMs can perform as well as humans on the real-world problem of traffic sign recognition [114], however are outperformed by a combination of CNNs and MLPs approach.

Li et al. [56] present a study on the performance of multiclass SVMs on gene expression data, outperforming KNNs and Naïve Bayes classifier methods. Their main insights show that generalization performance is high when the number of classes is small and decreases significantly with a large number of classes (60).

3.3.2 Multiclass classification with neural networks

Windeatt et al. [129] argue in the favor of splitting the multiclass problem into several complementary binary subproblems and use the Error Correcting Output Code (ECOC) technique. The idea is to employ several base classifers each of which is responsible for solving a subproblem with different class labeling. The motivation of using base classifiers in this way is to keep the results of them as uncorrelated as possible. The authors furthermore point out that one of the advantages of this approach is the fact that subproblems can better be focused on and even further that parameters are more easily to be fine-tuned. They argue against using the encoding approach with a multiclass classifier as all output nodes share the same weights defined in the training process leading to errors being not independent from each other and thus making the benefit of combining several classifiers negligible.

Brimberg et al. [11] use 2-layer and 3-layer MLPs within the Automatic Speech Attribute Transcription project in order to incorporate attributes of the speech signal for Automatic Speech Recognition (ASR). The individual phonemes are mapped into feature vectors used as network input and trained on the TIMIT database containing 2342 different sentences and demonstrate that multiclass MLPs outperform SVMs in terms of accuracy.

Multiclass classification is devoted to the task of mapping a sample from an input space to an output space with more than two classes. The applications are manifold especially as typical real-world applications demand being able to distinguish between more than two classes, e.g. speech recognition tasks, object classification in general or the task at hand - hand gesture recognition. There exists an extensive body of work on two class classification - as opposed to multiclass classification - as the latter is frequently being treated as an extension of the former which may result in increasing complexity or decreasing performance.

The multiclass classification problem is usually approached by decomposing it into multiple binary classification tasks and handling each one by one. Approaching a *k-class* classification problem with neural networks can be done by either employing a system of multiple neural networks combined with a decision module or a single neural network handling the disambiguation task on its own. Depending on the approach chosen, the pattern classes have to be modelled accordingly which can be done by the one-against-one approach (OAO), one-against-all (OAA) or the P-against-Q approach (PAQ). Each of the mentioned modelling possibilities has its own advantages and disadvantages in terms of parameter choice, modelling overhead, training and learning depending on the size of the data set and learning capabilities. We will address these issues and demonstrate why the approach chosen in this work is convenient for the task of hand gesture recognition.

The problem of distinguishing between k classes is the problem of mapping an n-dimensional *feature vector* $\vec{x}$:

$$\vec{x} = \begin{pmatrix} x_0 \\ \vdots \\ x_n \end{pmatrix} \tag{3.14}$$

from an n-dimensional feature space to its corresponding label $l \in \{l^1, ..., l^k\}$. For an input $\vec{x}$ its label is defined unambiguously i.e. $l^h \neq l^i$ for all $h \neq i$. We can train a neural network in such a way that can make such a prediction for an input vector.

When the problem is approached by the OAA scheme, i.e. each of the classes is trained against all other classes, this can be realized by either a single neural network with the number of output nodes $m = k$, the number of classes. If it is realized by multiple binary neural networks, the system consists of k binary neural networks. In the OAO approach the system can only be realized with $\frac{k(k-1)}{2}$ binary neural networks. For the PAQ scheme, p of the k classes are trained versus the remaining q classes which can be realized by a single neural network or a system of multiple neural networks, each discriminating between binary or multiple classes.

Figure 3.2 shows that decision boundaries drawn by hyperplanes in the feature space for the various approaches can be optimal (bottom left) or optimal for the specific task but problematic in general, as the hyperplanes go through the feature space of the remaining classes (top right).

In this work the focus is put on the OAA scheme with a single neural network. The network architecture thus consists of input, hidden and output layer. The input layer has size n - the dimensionality of the feature vector. The size of the hidden layer varies as it is problem-dependent and hence determined by a number of test runs on the respective task. The output layer comprises k output neurons. Each pattern class is a codeword of k bits, a codeword for the ith class then equals $O_1 = ...$, $O_{i-1} = 0$, $O_i = 1$, $O_{i+1} = ... O_n = 0$. Neural networks trained in this manner can suffer from

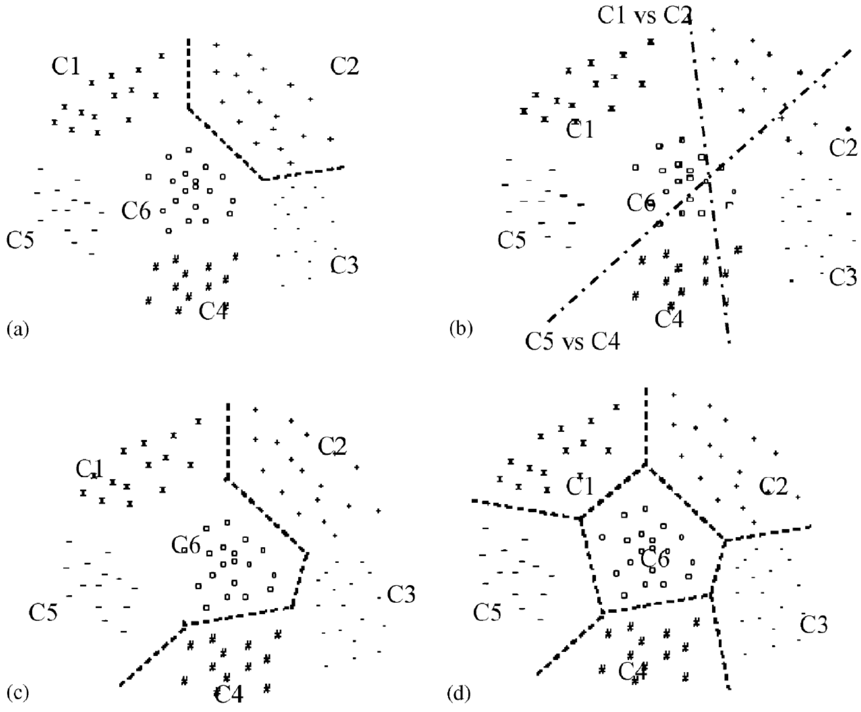


Figure 3.2: The decision boundaries depending on the model choice. Top left: decision boundary drawn by a single neural network trained with the OAA approach. Top right: two decision boundaries by two neural networks with the OAA approach. Bottom left: Decision boundary of a single neural network for the PAQ model. Bottom right: The optimal separation of all classes in the feature space. Source: [79]

ignoring minority classes, i.e. classes represented by many less samples than other classes. However optimal decision boundaries become possible as seen in Figure 3.2 (bottom right). In this case, features are developed by units in the hidden layer, therefore some neurons become responsive to certain classes while being ignored by others (see [16]), while features common for multiple classes can be shared which is achieved by smaller or larger weights depending on to which class the respective neurons belong to. Training time scales with the number of weights to be adapted, which in turns depends on the number of layers and neurons within each layer, as well as on the number of training samples.

As the task at hand requires setting up a large database with around 600,000 samples while additionally the size of the feature vectors is high dimensional, this partly leads to complex neural network architectures. In this thesis, the number of hidden layers as well as neurons within this layer is determined by multiple experiments in order to achieve satisfactory generalization performance. Training time rose to potentially several hours (>8) per network (see also [79]) which in some cases proved impracticable as keeping the number of hidden neurons reasonably low also led to satisfactory performance. Furthermore, depending on the fusion technique, this could potentially scale even further. Further details about the optimal choice of parameter, training technique and network architecture will be explained in Part II.

Part II

Own contribution

Introduction

In order to tackle the challenging multiclass problem of static and dynamic hand gesture recognition, a number of different algorithms have been developed, some of which are based on the insights from the initial findings, and continually improved over the course of this thesis. The backbone of this contribution is represented by the hand gesture database described in Chapter 4 which, towards the end of this work, contains more than 1.2m samples. In its initial form, the number of data samples included was significantly less (as was the number of different persons in the data set) but due to the complexity of the problem and the developed algorithms' demand for data, it was continually extended to its current size.

The rest of this thesis is organized as follows: After the description of the database, the contribution to the topic of hand gesture recognition is split into three different chapters, namely the theoretical background of the ML algorithms outlined in Chapter 5, followed by the real-time applicable hand gesture recognition system described in Chapter 6 and concluded with research conducted towards the HMI-related question of what makes up an intuitive and natural gesture-controlled interface explained in Chapter 7.

The end of this thesis is formed by a critical discussion of the most relevant results, complemented by research conducted towards the end (therefore it did not find its way into this work), and finalized by an outlook and a perspective on forthcoming research topics.

Chapter 4

Hand gesture database

In order to evaluate the performance of the various algorithmic approaches and fusion techniques described later on, a database consisting of a large number of point cloud samples, partially recorded in a setup with multiple ToF cameras, was established and extended[1].

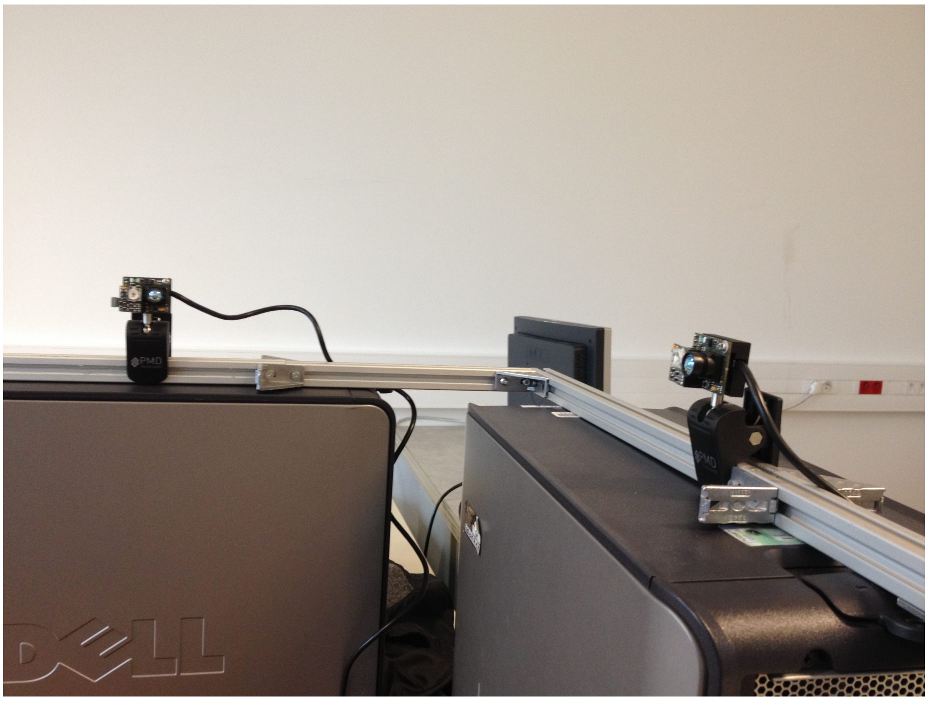


Figure 4.1: The recording setup for the multi-sensor database. Both sensors are aligned in a perpendicular angle to each other at a distance of approx. 49.5cm.

[1] The Hand Gesture Database lays the groundwork for the algorithms and the framework developed in this thesis. In its original form it contained roughly 80,000 samples recorded from 2 angles, stemming from 4 persons. It was extended to contain more than 1.2m samples from 20 persons. This and the subsequent chapter therefore appeared with minor modifications in multiple publications [46],[47],[48].

Figure 4.1 shows the setup for the recording of the multi-sensor database. Data is recorded by two ToF sensors of type Camboard Nano which provides depth images at a resolution of 160x120px with up to 90fps. The illumination wavelength is 850nm which makes the cameras applicable in various light conditions whilst maintaining robustness versus daylight interferences. Since the ToF-principle works by measuring the time the emitted light needs to travel from the sensor to an object and back pixel-wise the light is modulated by a frequency of 30MHz in order to be able to distinguish it from interferences. In a multi-sensor setup however this may lead to a distortion of measurements since both sensors have the same modulation frequency. To avoid such measurement errors, the data was recorded by taking alternating snapshots from each sensor.

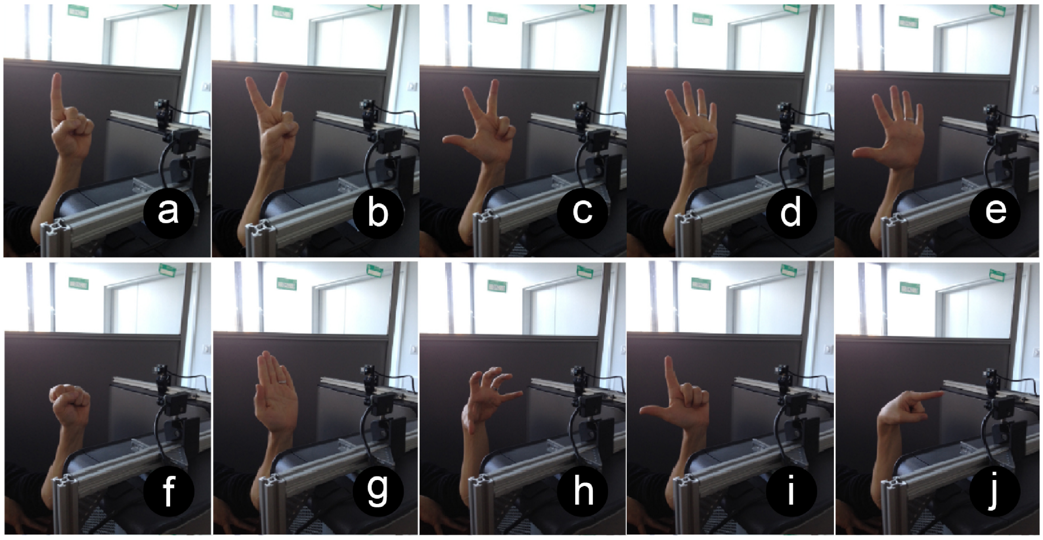


Figure 4.2: The static hand pose database consisting of 10 different hand poses: Counting from 1-5 (top row) and *fist*, *flat hand*, *grip*, *L*, *point* (bottom row).

As can be seen in Figure 4.1 the cameras are mounted in a fixed position at a distance of approx 49.5cm and a perpendicular angle from the recorded object. This allows for a recording of the database such that the hand can be placed in an equal distance of about 35cm from each camera to the centroid of the resulting point cloud data set and therefore each camera can also be calibrated to its needs. As research results proved during this thesis, an angle of 30° provides sufficient results when fusing data coming from multiple sensors, hence the subsequent recordings were adapted to this angle[2]. In order to maintain a certain variability in the data, each set of poses

[2] It is important to note that initial research results yielded similar performance independent of camera angle and the algorithm of choice. Hence the database in its primary form was discarded and replaced by a similar setup at 30° camera angle to be realizable in a car setup.

was recorded with a variation of the hand posture in terms of translation and rotation of the hand and fingers. This resulted in an alphabet of ten hand poses: *point*, *fist*, *grip*, *L*, *stop* and counting from 1-5 (cf. Figure 4.2). For each pose a set of 2000 point clouds was recorded for each camera yielding a total data set of 40.000 samples.
This basic hand pose data set was extended in order to achieve more variation in the data. However, the focus is put on data coming from one sensor only. To this end, data now comes from 20 different test persons, each posing for 3000 samples for every single hand pose. To induce scaling into the point cloud data, three range zones *near*, *intermediate* and *far* (20-30cm, 30-40cm and 40-60cm respectively) were defined, within which each individual subject performed a hand pose until a minimum number of samples was recorded. For one person, this resulted in 1000 samples being recorded in each of the mentioned range zones for each hand pose. All in all, 30,000 samples per person were recorded summing up to 600,000 data samples for the whole data set (ergo 1.2m samples for two cameras). In order to induce further variance, every individual was asked to rotate and translate their hand in all possible directions. The illumination time to 800μs as this results in a satisfactory signal-to-noise ratio and is a well-tested value for near-range interaction of up to 1m.

As every sample is recorded with a *different amount of arm* integrated into the object, the subsequent section describes the means employed to get rid of all the data points included in each cloud and carrying irrelevant information for the task.

4.1 Hand-forearm cropping with principal component analysis

The main directions of the cloud are found using Principal Component Analysis (PCA) [40].

PCA aims to find uncorrelated basis vectors for an arbitrary set of data vectors. Eigenvectors (also termed ”principal components”) are ordered by the variance of data points projected onto them, allowing efficient data compression by omitting principal components of low variance. This algorithm is applied as shown below, using as input the set of n 3D coordinates of points in a point cloud denoted x_j, $j \in [1, n]$):

- The mean value $\bar{x} = \frac{1}{n} \cdot \sum_{j=1}^{n}(x_j)$ is computed.
- The scatter matrix is calculated by:

$$S = \sum_{j=1}^{n}(x_j - \bar{x})(x_j - \bar{x})^{\top}$$

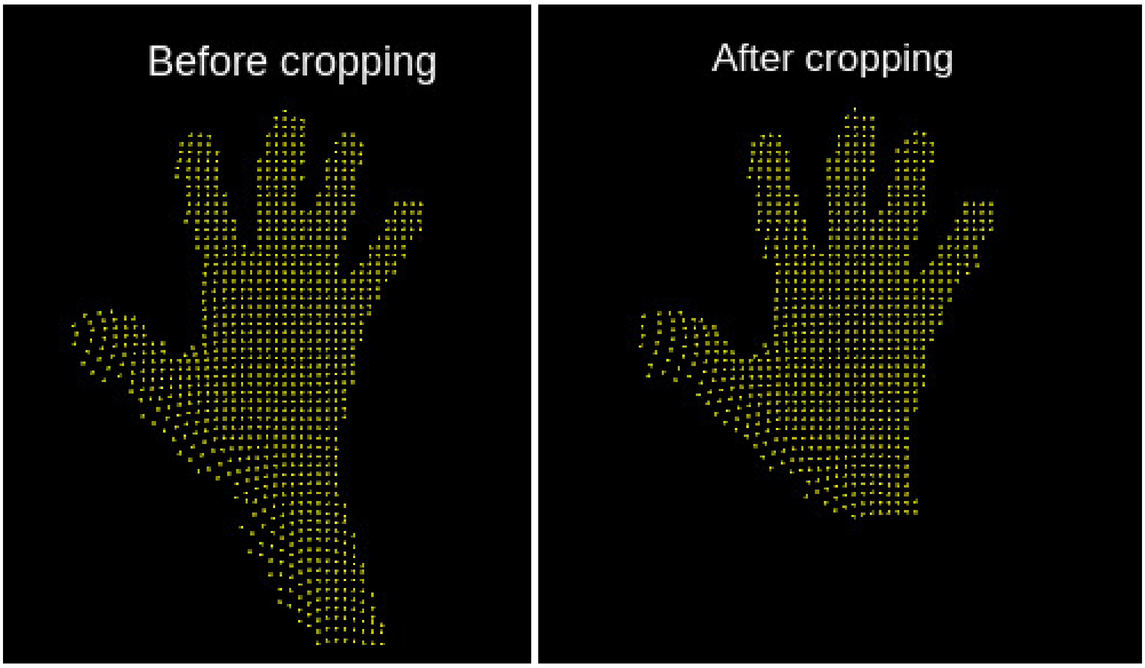


Figure 4.3: Point cloud before PCA-cropping (left) and after (right).

This matrix can be used as maximum-likelihood estimate of the covariance matrix.

- The Eigenvectors of this matrix yield the principal components.

The intention is to cut off 'unnecessary' parts of the cloud, i.e. outliers and elongated parts of the forearm. In this case, the principal components correspond to orthogonal vectors that represent the most important directions in the point cloud. The vector with the most important y-component allows to recognize the hand-forearm axis.

The wrist, as the link between the hand and the forearm, is detected in order to determine a limit for the cropping. The employed method assumes that the distance between the endpoint of the fingers and the centroid is an upper bound of the distance between the centroid and the wrist.

To find the endpoint of the hand towards the direction of the fingers, tests are made along the axis, starting at the centroid and moving progressively upward. At each step, it is determined whether there are points within a designated small circular neighborhood around the axis. The upper end of the hand is marked if this number of neighboring points equals 0. Then the bottom limit for the wrist is fixed at the same distance from the centroid, but in the inversed direction along the y-axis. All points below this wrist limit are cut out which is exemplary shown in Figure 4.3.

Chapter 5

Fusion techniques for multiclass classification with MLPs

This section is dedicated to the various fusion techniques developed within the frame of this thesis. In Section 5.1 the results of fusing multiple ToF sensors to improve the overall recognition rate are presented. It is researched how employing more than one sensor can significantly boost recognition results in especially difficult cases and get a first grasp on the influence of the descriptors for this task as well as the influence of the choice of parameters on the calculation of the descriptor. Moreover, the introduction of confidence parameters stabilizes the overall performance of the system. The performance of MLPs with standard parameters is compared with the performance of SVMs for which the parameters have been obtained via grid search.

In order to improve the classification performance of the MLPs without the need for sophisticated algorithm design or extensive parameter search, a simple method is proposed in Section 5.2, which makes use of the existing recognition routines by exploiting information already present in the output neurons of the MLPs. This simple fusion technique combines descriptor features with neuron confidences coming from a previously trained net, and it proves that augmented results can be achieved in nearly all cases of problem classes and individuals respectively. Moreover its flexibility becomes visible as it can easily be extended and adapted to all kinds of problems, requiring only some adaptation to the training and classification processes.

5.1 An approach to multi-sensor data fusion

In this section the initial approach of improving hand pose recognition by employing more than one ToF-sensor is presented.[1] To this end, the initial database for one of the two angle constellations consists of 40,000 samples being recorded for a single person where each hand pose was recorded 2000 times. This suffices as the focus of this contribution goes to show that multiple sensors in fact do improve the overall performance while simultaneously demonstrating that the magnitude of the angle is not the most important factor. However the performance varies with the introduction of confidence measures and it furthermore can be shown to which extent the choice of descriptor affects the overall performance. The subsequent sections describe the details of the parameters as well as the experiments more in-depth.

5.1.1 Descriptor parametrization

All used global descriptors were calculated using methods of the publicly available Point Cloud Library (PCL) as outlined in Section 2.3. Two descriptors were used - the ESF and the VFH descriptor - which shall be described briefly with regard to the choice of parameter.

5.1.1.1 The ESF descriptor

The ESF [130] is a global descriptor which does not rely on the calculation of the normals. First, 20,000 points are sub-sampled from the input point cloud. Then, the algorithm repeatedly samples three points, from which four simple measures are calculated, which are discretized and used for histogram calculation. The whole concept is detailed in Section 2.3.3.

5.1.1.2 The VFH descriptor

The VFH (cf. 2.3.4) [106] is a global descriptor partially based on the local FPFH [103] descriptor. It uses normal information, taking into consideration the view angle between the origin of the source and each point's normal. It furthermore includes the SPFH (Simplified Point Feature Histogram) for the centroid of the cloud, as well as a histogram of distances of the points in the cloud to the centroid. When calculating the VFH for the various hand poses, the influence of the normals on the results has

[1] Some parts of this section appeared with modifications in Thomas Kopinski, Alexander Gepperth, Stefan Geisler, and Uwe Handmann. Neural network based data fusion for hand pose recognition with multiple tof sensors. *ICANN*, 2014

to be taken into consideration. In the described case the search parameter r guides the influence of the surrounding for the calculation of the normal. Choosing a small r can result in low descriptive power while a large r results in high computational load while possibly also overly distorting the captured information. A value of $r = 5cm$ is chosen empirically and therefore the resulting descriptor is denoted as VFH5.

5.1.2 Neural network classification and fusion

With M cameras, N descriptors will be produced per frame (here: M=N) according to the descriptors described above. An MLP is used to model the multiclass classification function, either implicitly by concatenating all N descriptors and subsequent NN classification (”early fusion”) or explicitly, by performing classification individually on each of N descriptors and then combining results (”late fusion”). Networks contain a bias unit at each layer, the training algorithm is ”RProp”[34] with hyperparameters $\eta^+ = 1.2$, $\eta^- = 0.6$, $\Delta_0 = 0.1$, $\Delta_{\min} = 10^{-10}$ and $\Delta\text{max} = 5$. Network topology is NK-150-10 (hidden layer sizes from 10-500 were tested without finding significant performance improvements), K indicating the size of the used descriptors and N the number of cameras, here $N = 2$. Activation functions are sigmoid throughout the network. As the MLP classifiers have 10 output neurons, corresponding to the 10 gesture classes, with activities o_i, the final classification decision is obtained by taking the class of the neuron with the highest output. However, not every classification shall be taken seriously, therefore several confidence measures are defined $\text{conf}(\{o_i\})$ to this effect. Final decisions are thus taken in the following way:

$$\text{class} = \begin{cases} \text{argmax}_i o_i & \text{if } \text{conf}(\{o_i\}) > \theta_{\text{conf}} \\ \text{no decision} & \text{else} \end{cases}$$

Three different confidence measures are tested, which perform a mapping from $\mathbb{R}^{10} \rightarrow \mathbb{R}$: ”confOfMax”, ”diffMeasure” and ”varianceMeasure”. Each of these measures is derived from the idea of approximating an entropy calculation, based on the information-theoretic idea that low entropy means high information content. The precise definitions are as follows:

$$\begin{aligned} \text{confOfMax}(\{o_i\}) &= \max\, o_i \\ \text{diffMeasure}(\{o_i\}) &= \max_i\, o_i - \max_i^2\, o_i \\ \text{varianceMeasure}(\{o_i\}) &= \sqrt{\frac{1}{N}\sum_i (o_i - E(\{o_i\}))^2} \end{aligned} \tag{5.1}$$

where $\max_i^2 o_i$ indicates the second-strongest maximum over the neural outputs. For performing late fusion, that is, obtaining two independent classifications o_i^1, o_i^2 based on each camera's features, the arithmetic mean of both output vectors is calculated: $o_i^F = 0.5(o_i^1 + o_i^2)$. This intrinsically takes into account the variance in each response, as an output distribution strongly peaked on one class will dominate a flat (or less peaked) distribution. The resulting output distribution o_i^F can then be subjected to the decision rule of Equation (5.1).

5.1.3 Experiments with MLPs

An MLP is implemented as described in Section 5.1.2 using the freely available OpenCV library [9] and its C++ interface. Each experiment is performed 10 times with different initial conditions for the MLP, and the best result is retained. In these experiments, the influence of different confidence measures ("confOfMax", "diffMeasure" or "varianceMeasure", see Section 5.1.2) is systematically evaluated on the fusion strategy ("add", see Section 5.1.2) while measuring the performance of the first camera, the second camera as well as an "early fusion" or a "late fusion" of the two cameras. In order to test the influence of different 3D descriptors, an identical evaluation is performed except that the VFH5 point cloud descriptor is replaced by the ESF. Additionally, the same evaluation is performed on an analogous database using the VFH5 descriptor where the angle between ToF sensors is 90 deg. Results are evaluated by default according to whether one among the S strongest output neurons coincides with the true class of a point cloud ("S-peak measure"). Unless explicitly states, $S = 1$. Results are given in Figure 5.1.

Several important aspects can be perceived: First of all, fusion strongly improves results in comparison to any single sensor, w.r.t. to the efficiency of sample rejection but also in absolute terms when no samples are rejected, corresponding to the intersection of the graphs with the right boundary of the coordinate system. Secondly, early fusion has slightly superior performance than late fusion but the difference is marginal, potentially giving a preference to late fusion due to reduced computational complexity. Lastly, the different confidence measures are consistently ranked throughout all experiments, with the "diffMeasure" being the best-performing one, closely followed by "confOfMax". This is encouraging as especially confOfMax is computationally very lightweight, again favoring real-time execution. Thirdly, the angle between cameras does not seem to play a crucial role even though individual camera results differ considerably. Here, the beneficial aspects of fusion can be clearly demonstrated. And lastly, the ESF descriptor seems to perform slightly better than

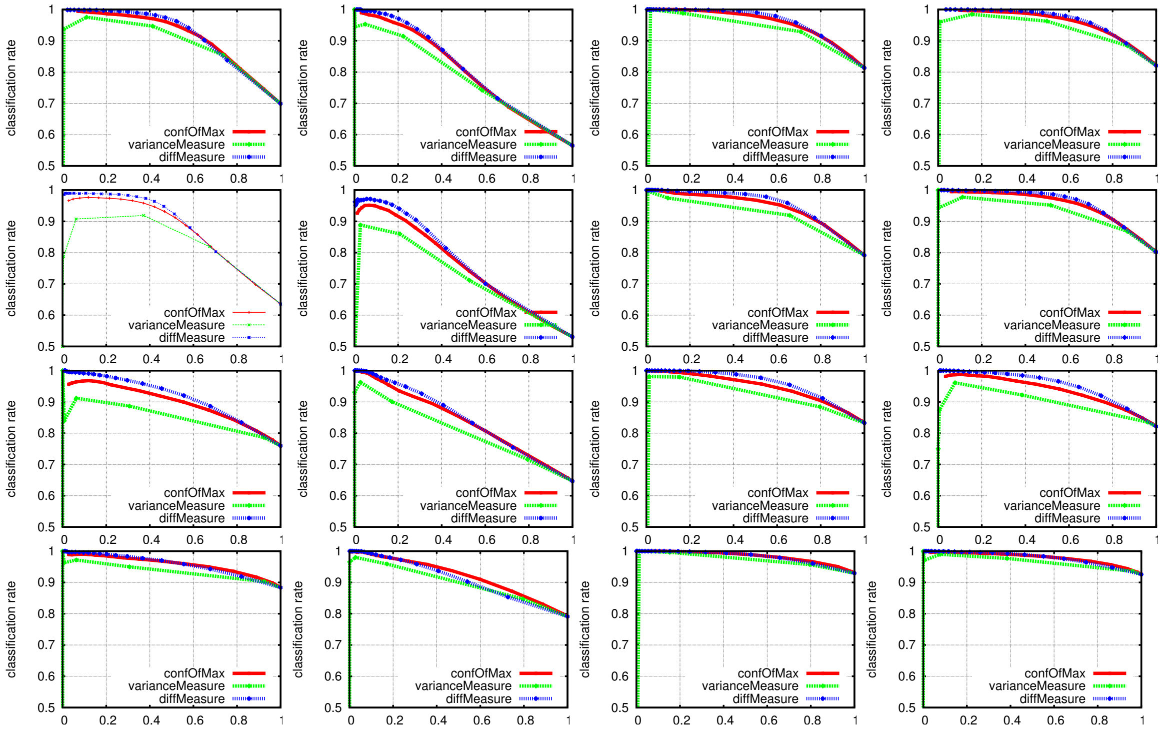


Figure 5.1: Experimental results. First row: VFH5 descriptor, 30° between cameras. Second row: VFH5, 90° between cameras. Third row: ESF descriptor, 30° between cameras. Last row: Same as third row, only classification errors evaluated using the two-peak measure, see text. In all rows, the order of diagrams is, from left to right: 1,2) first/second sensor 3) late fusion 4) early fusion. Individual plots show the effects of varying confidence thresholds on classification accuracies for several possible online confidence measures. The method-dependent confidence thresholds are not shown, but rather the acceptance rates which change if thresholds are varied. At the far right of each diagram, the classification performance is obtained when not rejecting anything, naturally leading to reduced performance.

VFH5, which might lead to prefer this descriptor as it is computationally simpler and requires constant execution time regardless of point cloud size. An interesting observation is that the two-peak measure enormously improves classification rates in all conditions. This is very useful for an application, especially for temporal filtering, as the behaviour of the second-strongest output can obviously also provide valuable information about the true pose class.

Training times are around 10min per single experiment, which outperforms an equivalent SVM-based (Support Vector Machine) implementation by a large margin as this would require 10 one-vs-all training runs with 80.000 examples each, which would take a few days at the very least.

Execution times vary around 5-8 Hz depending of the use of the descriptor (ESF vs. VFH), the influence radius of the normals and the size of the point cloud itself. On average the point clouds contain 1300-1600 points. This depends on the angle of the camera, the distance of the recorded hand to each camera and lastly of course also on the size of the hand of each of the four users.

5.1.4 Experiments with SVMs

This section compares the performance of SVMs to the performance of MLPs on exactly the same database and the same choice of descriptors. In the experiments, two types of kernels are evaluated, the ordinary scalar product and the gauss kernel.[2] The parameters have been obtained via a grid search over an exponentially sampled parameter space. The crossvalidation set for ESF descriptors consisted of a total of $40,000$ feature vectors $x \in \mathbb{R}^{1280}$. All numerical values were sampled with double precision and special attention towards compare operations. In order to narrow down the search range, a coarse crossvalidation with $n = 5$ is applied. As shown in [100] this reduces the variance of the obtained parameters yet increases the corresponding bias. Thus, once the area had been narrowed down, a thorough gridsearch with $n = 100$ was commenced. This method can be regarded as a two-stage approach, Figure 5.2 depicts the first stage results for an RBF kernel and VFH descriptors. One can see that the kernel parameter's influence on the classification result becomes smaller for large penalty factors.

Using the scalar product, the best SVM obtained a classification performance of

[2] This section appeared in the underlying form in Thomas Kopinski, Darius Malysiak, Alexander Gepperth, and Uwe Handmann. Time-of-flight based multi-sensor fusion strategies for hand gesture recognition. In *Computational Intelligence and Informatics (CINTI), 2014 IEEE 15th International Symposium on*, pages 243–248. IEEE, 2014

$\approx$ 98.7% for 30 and 98.8% for 90°. The time needed for the grid search was approximately 16 hours. Regarding the gauss kernel, a classification rate of 99.8% for 30 and 99.6% for 90° was obtained with a total training time of approximately 2 days.
The setup and evaluation procedure for the VFH descriptors is identical to the ESF case. The only difference lies in the reduced size of the VFH descriptors which contain only 616 elements. The results are summarized in table 5.1. It must be mentioned that the VFH descriptors allowed a much faster parameter estimation due to the reduced data volume. Using the scalar product the needed time was 3 hours while $\approx$ 23 hours were needed for the gauss kernel. Although training has proven to be a slow procedure, classification of a single feature vector can be done very efficiently and takes only fractions of a second (e.g. 4μs / 250kHz for ESF descriptors on a modern CPU).

Descriptor	ESF 30	ESF 90	VFH 30	VFH 90
Classif. rate scalar kernel	98.7%	98.8%	96.9%	94.2%
Classif. rate gauss kernel	99.8%	99.6%	98.8%	93.1%

Table 5.1: Classification results for both descriptor types. The SVMs were obtained through a two-stage grid search.

5.1.5 Discussion

Analyzing the results it can be stated that, first of all, fusion with data from a second ToF sensor improves results tremendously in all investigated conditions, camera setups and point cloud descriptors. Interestingly, late fusion performs globally just as well as early fusion, which is important as it has the potential to be much more computationally efficient. However, even when considering individual ToF sensors, the computation of confidence measures from output activity distributions is of tremendous impact as well. Confidence can be efficiently extracted at execution time (no need to see the class labels for this) and used to avoid classification decisions when they are likely to be incorrect anyway. A number of information-theoretically motivated measures were tested and luckily the most efficient measures seem to perform best. Concerning the influence of the used 3D descriptors, the ESF descriptor yields best performance with or without fusion. As this descriptor does not require normals computation and has approximately constant scaling behavior w.r.t. point cloud size, it is the more appropriate choice for real-time applications in the targeted automotive domain.

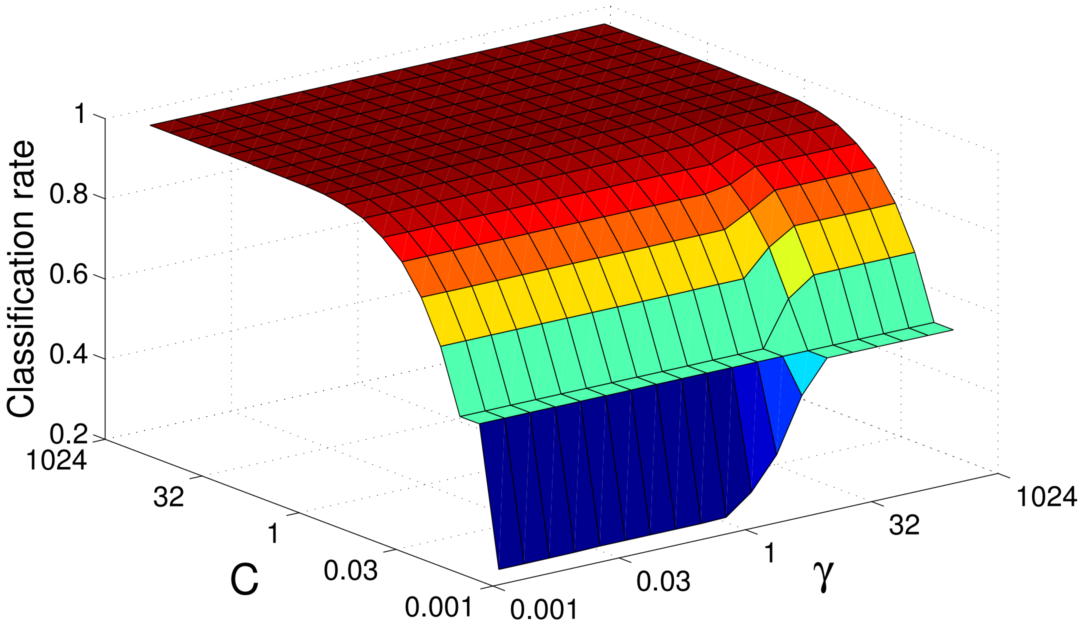


Figure 5.2: Development of classification rate during a grid search for VFH descriptors using an RBF kernel. For large values of the penalty factor C the kernel parameter γ loses some of its significance.

The use of support vector machines showed classification rates similar to these of neural networks. Gauss (or RBF) kernels are very common when it comes to SVM-based classification as they usually show the better results. In this case it was found that for ESF descriptors, the standard scalar product allows the training of SVMs with a recognition rate close to RBF-based SVMs. The difference between both kernel types lies in the range of $\approx 1\%$ (with the RBF SVMs being better). There is effectively no difference between ESF descriptors for 30° or 90°. The VFH descriptors showed very similar results, they as well lie only slightly apart when comparing both kernel types. Interestingly, the VFH descriptor degenerates in its classification power for the 90° case. Overall the VFH descriptors seem to perform less effectively compared to the ESF features, which seems to indicate that the ESF descriptor retains more of the sample's variance. Thus, using the scalar product, one can construct classifiers which are faster and structurally simpler than neural networks. As only a single scalar product has to be evaluated compared to more complex matrix operations of neural networks. Yet with the inherent drawback of huge training times one has to carefully assess the applicability of support vector machines. Neural networks should be favoured especially if it comes to online or mini-batch learning, whereas SVMs should be used in areas which do not require retraining and rapid deployment. Summarizing, an adaptive data fusion approach for multiple ToF sensors, addressing the generic task of 3D point cloud categorization in a multiclass setting, is presented. Using a neural network for this purpose is of high advantage (besides very favourable database size scaling and multiclass issues) as the ensemble of normalized output confidences contains valuable information as well that can be efficiently exploited at runtime to improve results. Neural network learning furthermore removes the need for precise multi-sensor calibration as long as only categorization is targeted.

How to further exploit information hidden in MLP layers is the focus of the subsequent sections.

5.2 Extracting information from neuron activities

This section compares a novel *fusion approach* with a simple *cascade learning* approach. More specifically, it is demonstrated how fusing features extracted from data with neural activities from previously trained MLPs can effectively be utilized to improve the overall classification results. It can be shown that although there is information carried by the neurons, which can be exploited to improve the hand gesture recognition task, it is, in fact, the combination of features and neuron activities which brings the most significant improvements.[3]

5.2.1 Contribution and novelty

This contribution proposes a pragmatic and application-oriented approach to MCC when performing an OVA decomposition. Instead of making a priori assumptions about distributions or conditional independence properties, it is attempted to learn such properties from data, and to exploit this knowledge for improved accuracy. For the concrete case of an MLP classifier trained on a difficult hand gesture recognition task, the question of how the addition of another MLP stage that operates on the class output activities (COAs) of the first can affect performance is investigated. The basic assumption is that the selective inter-class correlations in the COAs, which can be observed in Figure 5.3 and, supposedly, will exist for any problem, contain useful information that the second MLP stage can extract and harness. This technique is evaluated in a number of experiments (sections 5.2.5 - 5.2.9) and extended by various

[3] Some of the following sections appeared in Thomas Kopinski, Stephane Magand, Alexander Gepperth, and Uwe Handmann. A pragmatic approach to multi-class classification. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, page to appear. IEEE, 2015

modifications to demonstrate its versatility. Furthermore, it can be shown for the well-known MNIST classification benchmark for handwritten digits that the addition of this second stage does not always greatly improve performance, but that it causes no degradation either.

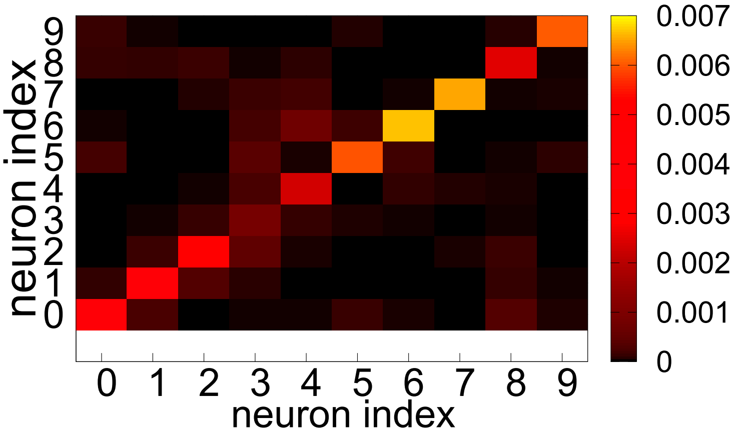


Figure 5.3: Unnormalized covariance matrix between the activities of output neurons in a neural network trained on the multiclass classification task considered in this study. It can be observed that some neurons are not at all correlated, and thus are rarely active together, whereas others are correlated quite strongly. The hypothesis is that this structure contains information about the task that can be used to further improve classification accuracy.

The novelty of this approach lies in its simplicity and generality, as well as its practical applicability. As the classifier hierarchy attempts to model the structure of the data by itself, no explicit assumptions need to be made by the user, other than issues of classifier design and parametrization, for which standard techniques exist.

5.2.2 Methods

In this section, mainly two different training techniques are presented, one of which can be extended n times, depending on the problem. The dataset has to be prepared accordingly which is covered separately for each approach. Furthermore, the databases used for all experiments of Section 5.2.4 are presented.

5.2.2.1 Exploiting information from output neurons

This section describes the cascading of two MLPs, where the basic idea is to let the first MLP classify a feature vector, and subsequently the second MLP takes as input the vector of output neurons of the first MLP. Training is performed sequentially, and care must be taken to prevent overfitting as each MLP is trained in a purely supervised fashion.

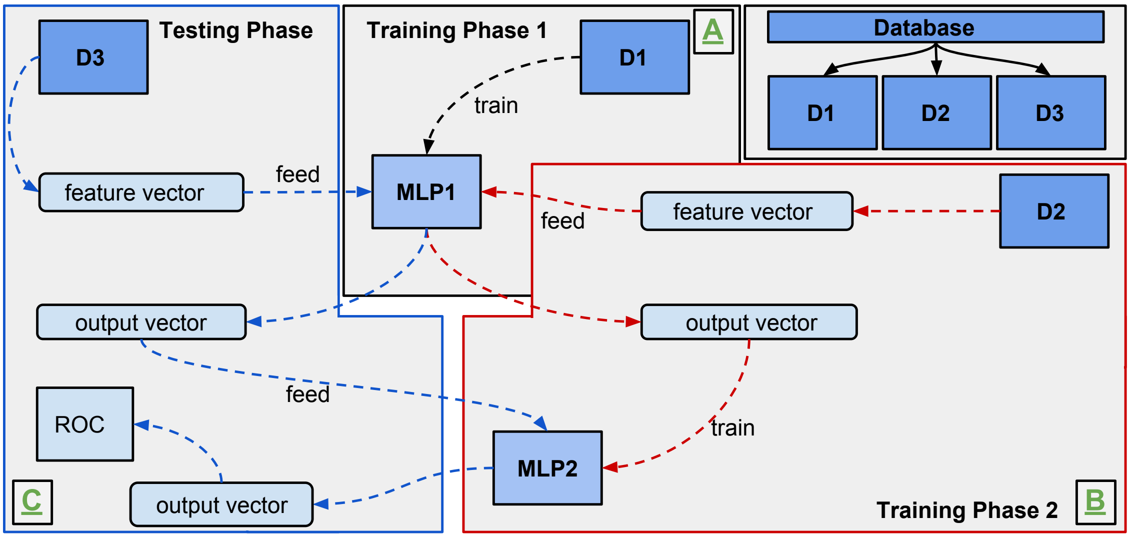


Figure 5.4: Training and testing procedure as described in Section 5.2.2.1. The whole database is randomly split into three subsets D1-D3. There are Training Phases 1 and 2 and one Testing Phase (denoted A, B and C respectively) during which the MLPs are trained and evaluated.

The procedure consists of three stages A, B and C which are schematically depicted in Figure 5.4. At first, the whole data set has to be divided up randomly into three equally sized sets D1, D2 and D3. In an initial step, the first MLP, here denoted MLP1, is trained with standard parameters (cf. Section 5.2.3) on D1. Once MLP1 has converged, training of the second MLP, denoted MLP2, begins on data set D2. The training is contrasted in such a way as now each individual training sample from D2 first has to be fed into MLP1. The input is propagated to the output layer and each output neuron makes a real-valued prediction for the possible class. These values form the output vector of length m equalling the number of classes in the MCC task at hand. This output vector in turn corresponds to the input value of each neuron in the input layer of MLP2. Therefore, for this approach, the size of the input layer of MLP2 always equals the size of the MCC in the given task. This propagation of information is shown in Figure 5.5.

In this way, MLP2 is trained until convergence. The performance of this three-stage approach is then measured on data set D3. Every sample is first fed into MLP1

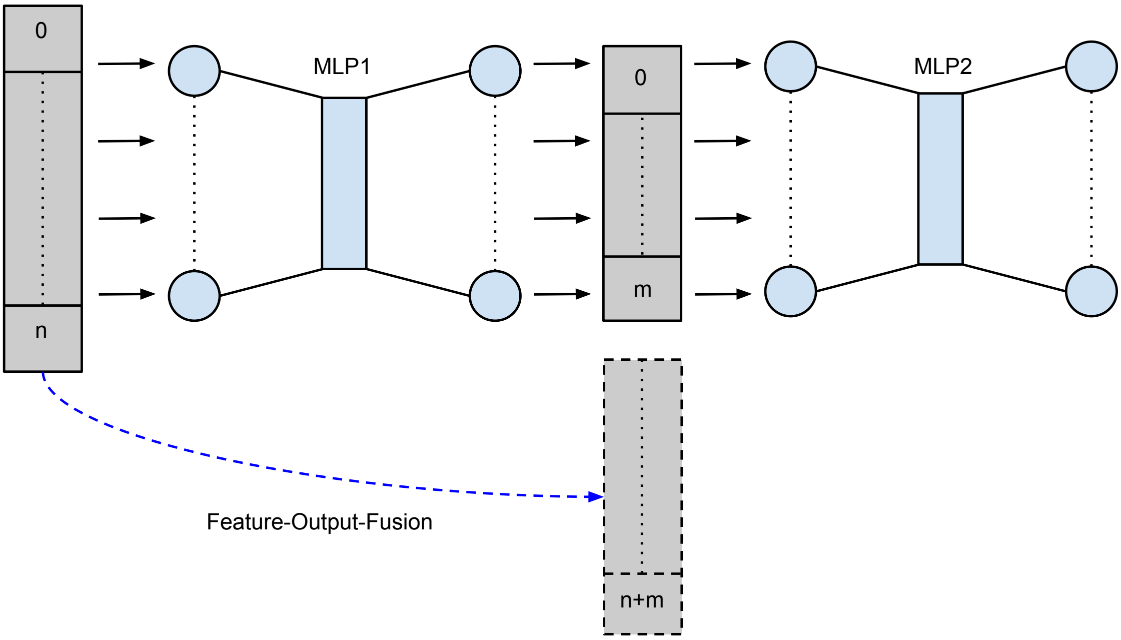


Figure 5.5: Schematic procedure of sample propagation and fusion technique. MLP1 is always trained with the unchanged samples taken from the training data set. A sample, represented by a feature vector of length n, is fed into the MLP's input layer. After MLP1 has propagated the input and calculated each neuron's activation in the output layer, MLP2 is trained on the output vector of size m. Optionally, the output vector is fused with the feature vector itself (cf. Figure 5.6), forming the new input of size $m + n$.

which again calculates the values of its output neurons. These values are then, analogously to the training phase, presented as inputs to MLP2 which in turn calculates its own outputs. The determined class for a sample S corresponds to the neuron with the highest activation in the output layer:

$$class\{S\} = argmax\{O_i\}, 0 \leq i \leq m, \tag{5.2}$$

m being the number of classes of the MCC and O_i representing the output neuron corresponding to class i.

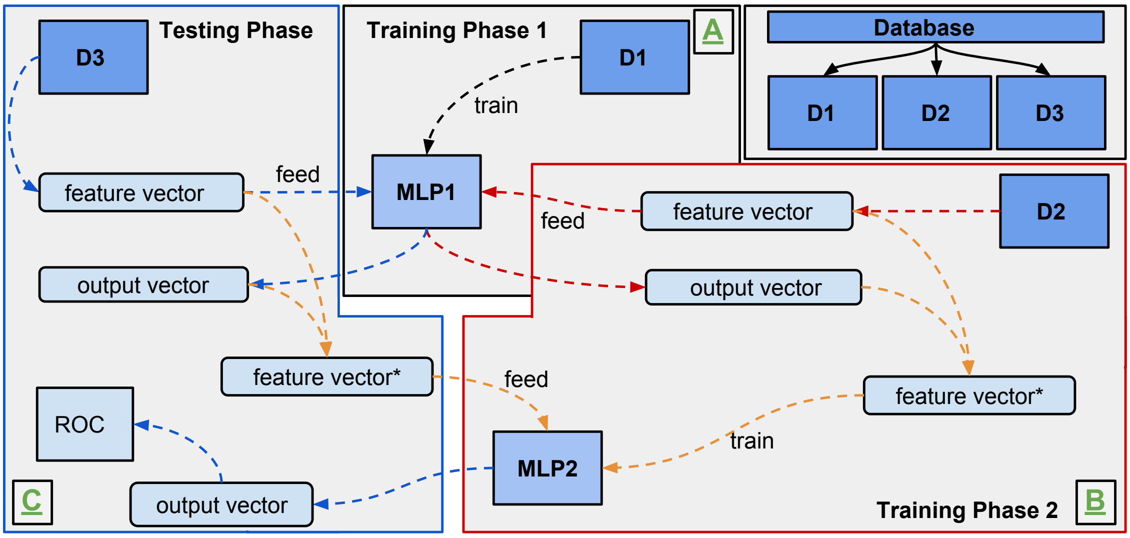


Figure 5.6: Training and testing procedure as described in Section 5.2.2.2. The whole database is randomly split into three subsets D1-D3. There are Training Phases 1 and 2 and one Testing Phase (denoted A, B and C respectively) during which the MLPs are trained and evaluated. The main differences are the fusion steps forming feature vector* from the original feature vector and the output vector. This occurs for MLP 2 in Training Phase B and Testing Phase C, highlighted in orange (cf. with Figure 5.4).

5.2.2.2 Fusing features and neural activities

The approach presented in the preceding section is extended in such a way that the training of MLP2 (and also the evaluation) is performed on the neuron values of the output layer of MLP1 but also on the features of the sample itself, i.e., the input to MLP1. The whole procedure again comprises three stages A, B and C and is schematically depicted in Figure 5.6, the main differences to the methodology shown in Figure 5.4 are highlighted. The data set is randomly split in an analogous manner to the approach described in Section 5.2.2.1 into three equally sized sets D1, D2 and D3. As before, MLP1 is trained on the whole data set D1 until it converges.

However the input for training MLP2 on D2 is now significantly different, comprising the feature vector of the sample plus the output vector of MLP1, $\vec{v}_{out} = [O_0, ..., O_n]$, n equaling the number of classes and O_i being the activation of neuron i in the output layer. Thus a new training input $\vec{f}_{\mathbf{n}}$ is formed resulting from the merging of the current feature vector $\vec{f}_{\mathbf{c}}$ and the output vector $\vec{v}_{\mathbf{o}}$ by simple concatenation. Its length $\text{len}(\vec{f}_{\mathbf{n}}) = \text{len}(\vec{f}_{\mathbf{c}}) + \text{len}(\vec{v}_{\mathbf{o}})$ is determined by the size of the descriptor on the one hand and the number of classes on the other hand. Figure 5.5

shows the propagation of the features and the formation of the input, in this case the Feature-Output-Fusion. Opposed to the procedure described in Section 5.2.2.1 the input now has length $n + m$, n being the size of the feature vector and m the size of the MCC.

A variation of this approach with only a slight modification of the original algorithm is simply achieved by training two separate MLPs (MLP1-A, MLP1-B) on D1. Now it is possible to concatenate both MLP outputs with the feature vector coming from training and test set D2 and D3 respectively. The main difference is that each sample, in training and testing phase, is now presented to both MLP1-A and MLP1-B which propagate their inputs so that both outputs generated from each MLP can be concatenated with the feature vector. The length of the newly formed feature vector now of course differs, depending on the size of the classification task, i.e. for the approach just described it is formed as $\mathrm{len}(\vec{f_{\mathbf{n}}}) = \mathrm{len}(\vec{f_{\mathbf{c}}}) + \mathbf{k}\cdot\mathrm{len}(\vec{v_{\mathbf{o}}})$, with $\mathbf{k}$ being the number of different MLPs trained on D1 or on separate data sets.

5.2.3 MLP structure and data set

Within the frame of the experiments, each MLP comprises one input, hidden and output layer, each layer being fully connected to its successor. The output layer depends on the size of the classification task as each class is represented by a neuron. Depending on the training technique used (cf. Sections 5.2.2.1, 5.2.2.2) the input layer varies in size. During the first stage, the input layer of the first MLP always equals the size of the feature vector. For the second stage, this varies depending on the technique employed as the input here is either formed by the output vector alone or the output vector concatenated with the feature vector. Therefore the input layer is of size n for MLP1 during training and testing and of size m or $m + n$ for MLP2, the number of classes or the number of classes + length of feature vector respectively, depending on the technique. When employing the extension of the method described in Section 5.2.2.2, the size of the input layer increases to $n+k*m$, k being the number of individual MLPs trained on a data set.

For the experiments, varying hidden layer sizes were tested within range of $[20, 150]$ neurons, having a noteworthy but no excessive effect on the results. Depending on the size of the input layer, which can vary due to the techniques described in this paper, choosing a different number of hidden neurons may be beneficial. However the focus of this contribution is to demonstrate the advantages of the presented methodology, hence it shall only be noted that a proper parameter search may lead to some improvement in terms of classification performance.

The whole data set comprises 480,000 samples from 16 persons. 10% of all data samples from each person and gesture were randomly retained for testing, thus in the following confusion matrices each row adds up to approx. 16,000 samples.

5.2.4 Experiments - overview

All methods were implemented using the FANN library [72]. The training algorithm is RPROP, the activation function is the sigmoid function in both hidden and output layers. The rest of the MLP parameters are standard parameters left fixed during the course of the experiments, as a series of initial test runs were conducted to determine proper parameters for the described methodology.

To evaluate the performance of the generated models the mean squared error (MSE) is evaluated:

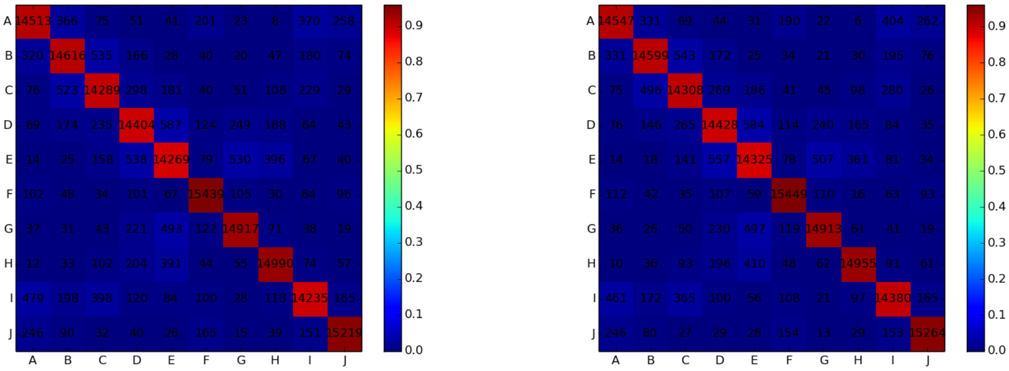
$$\mathrm{MSE}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{5.3}$$

with $\hat{y}_i$ being the predicted value, y_i the true value and n being the number of samples.

The experiments were taken out on the hand gesture database (cf. Section 4). All experiments are divided into two different testing phases (Phase 1 and Phase 2) in order to be able to directly compare the effects of the underlying techniques on the classification performance of each MLP. The results are depicted in the confusion matrices which allow to compare the overall performance of the MLPs, the performance on each individual class as well as the correlations between all classes.

5.2.5 Experiment 1 - training on output activities

In the first experiment the effect of the technique described in Section 5.2.2.1 is tested. The whole database is randomly split into three subsets D1-D3 (two for training and one for testing). Both MLPs have 100 neurons in the hidden layer and were trained on subset D1 and D2 respectively until they converged. The results can be seen in Figure 5.7. The overall performance of MLP1 (trained on the feature vectors) is at around 91.80% and at around 91.98% for MLP2 (trained on the output vector of MLP2) which is an improvement of around 0.2%. Overall, moderate improvements can be observed in nearly all cases, two cases are subject to negligible decrease in performance ($<$ 0,001%).



	A	B	C	D	E	F	G	H	I	J
A	14513	366	75	51	41	201	23	8	370	258
B	320	14616	535	166	28	40	20	47	180	74
C	76	523	14289	298	181	40	51	108	229	29
D	69	174	235	14404	587	124	249	188	64	43
E	14	25	158	538	14269	79	530	396	67	40
F	102	48	34	101	67	15439	105	30	64	96
G	37	31	43	221	493	122	14917	71	38	19
H	12	33	102	204	391	44	55	14990	74	57
I	479	198	398	120	84	100	28	118	14235	165
J	246	90	32	40	26	165	15	39	151	15219

	A	B	C	D	E	F	G	H	I	J
A	14547	331	69	44	31	190	22	6	404	262
B	331	14599	543	172	25	34	21	30	195	76
C	75	496	14308	269	186	41	45	98	280	26
D	76	146	265	14428	584	114	240	165	84	35
E	14	18	141	557	14325	78	507	361	81	34
F	112	42	35	107	59	15449	110	16	63	93
G	36	26	50	230	497	119	14913	61	41	19
H	10	36	93	196	410	48	62	14955	91	61
I	461	172	365	100	56	108	21	97	14380	165
J	246	80	27	29	28	154	13	29	153	15264

Figure 5.7: Left - Experiment 1, Phase 1: Confusion matrix for the first MLP trained only on the feature vectors. Right - Experiment 1, Phase 2: Confusion matrix for the second MLP trained only on the net outputs of the first MLP.

5.2.6 Experiment 2 - fusing output activities with features

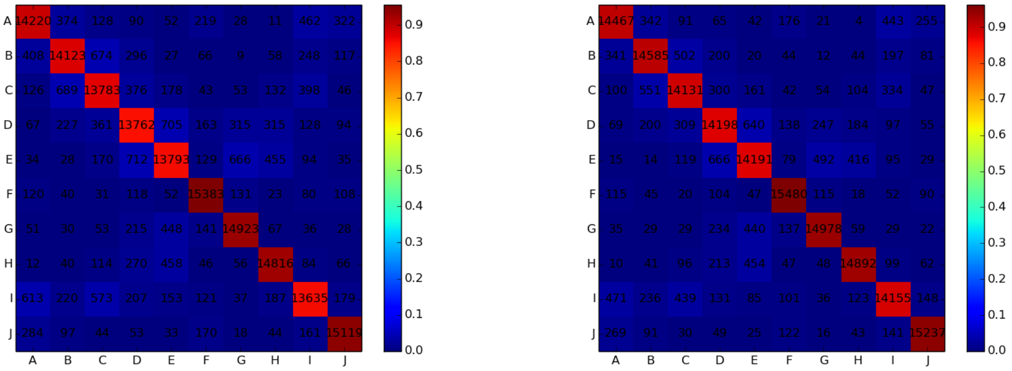
In the second experiment the effect of the technique described in Section 5.2.2.2 is tested. The whole database is randomly split into three subsets D1-D3 (two for training and one for testing). Both MLPs have 80 neurons in the hidden layer and were trained on subset D1 and D2 respectively until they converged. The results can be seen in Figure 5.8. The overall performance of MLP1 (trained on the feature vectors) is at around 90.0% and at around 91.0% for MLP2 (trained on the fused vector) which is an improvement of around 1.0%.

	A	B	C	D	E	F	G	H	I	J
MLP1	89%	88%	87%	85%	86%	96%	93%	93%	86%	94%
MLP2	91%	91%	89%	88%	88%	96%	94%	93%	89%	95%

Table 5.2: Classification results for MLP1 and MLP2. The ten classes are named A-J.

Table 5.2 gives more insight into the improvements of classification performance related to each individual gesture class. There is a performance increase for all cases (which is below 0.5% in cases F and H) and ranges between 1-3% for all other remaining cases. Most notably, the presented approach significantly boosts performance in situations where MLP1 performs poorly (cf. cases D + I) as opposed to little improvement in cases where MLP1 already performs well (e.g. cases F + J).

When comparing the confusion matrices of these cases it can be seen that the improvement stems mainly from those classes which contain most false positives, i.e. class I is most likely to be mistaken for class A or C (cf. Figure 5.8). The number of false positives for this specific example drops by a rate of >20% which is significant as it allows an improved disambiguation procedure (613 → 471 and 573 → 439 respectively).



	A	B	C	D	E	F	G	H	I	J
A	14220	374	128	90	52	219	28	11	462	322
B	408	14123	674	296	27	66	9	58	248	117
C	126	689	13783	376	178	43	53	132	398	46
D	67	227	361	13762	705	163	315	315	128	94
E	34	28	170	712	13793	129	666	455	94	35
F	120	40	31	118	52	15383	131	23	80	108
G	51	30	53	215	448	141	14923	67	36	28
H	12	40	114	270	458	46	56	14816	84	66
I	613	220	573	207	153	121	37	187	13635	179
J	284	97	44	53	33	170	18	44	161	15119

	A	B	C	D	E	F	G	H	I	J
A	14467	342	91	65	42	176	21	4	443	255
B	341	14585	502	200	20	44	12	44	197	81
C	100	551	14131	300	161	42	54	104	334	47
D	69	200	309	14198	640	138	247	184	97	55
E	15	14	119	666	14191	79	492	416	95	29
F	115	45	20	104	47	15480	115	18	52	90
G	35	29	29	234	440	137	14978	59	29	22
H	10	41	96	213	454	47	48	14892	99	62
I	471	236	439	131	85	101	36	123	14155	148
J	269	91	30	49	25	122	16	43	141	15237

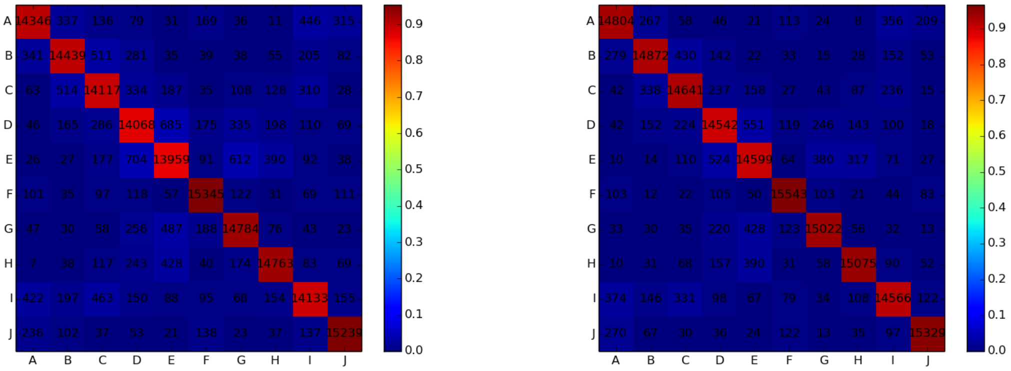
Figure 5.8: Left - Experiment 2, Phase 1: Confusion Matrix for the first MLP trained only on the feature vectors. Right - Experiment 2, Phase 2: Confusion Matrix for the second MLP trained on the fused feature vectors (features + outputs).

5.2.7 Experiment 3 - output neurons plus features with multiple MLPs

The third experiment evaluates the effect of the extended technique described in Section 5.2.2.2. Again, the whole database is randomly split into three subsets D1-D3 (two for training and one for testing). The main difference here is resembled by the fact that two MLPs (MLP1-A, MLP1-B) are trained on set D1, instead of just one. Therefore initially every sample is fed into both MLPs to calculate their output vectors. These are in turn then concatenated with the feature vector to form the new input vector for MLP2 during training and testing. The results of this extended technique are shown in Figure 5.9. The overall performance of MLP1 (trained on the feature vectors) is at around 91.0% and at around 93.0% for MLP2 (trained on the fused vector) which is an overall improvement of around 2.0%.

	A	B	C	D	E	F	G	H	I	J
MLP1	90%	90%	89%	87%	87%	95%	92%	92%	89%	95%
MLP2	93%	93%	93%	90%	91%	97%	94%	94%	91%	96%

Table 5.3: Classification results for MLP1 and MLP2. The ten classes are named A-J.



	A	B	C	D	E	F	G	H	I	J
A	14346	337	136	79	31	169	36	11	446	315
B	341	14439	511	281	35	39	38	55	205	82
C	63	514	14117	334	187	35	108	128	310	28
D	46	165	286	14068	685	175	335	198	110	69
E	26	27	177	704	13959	91	612	390	92	38
F	101	35	97	118	57	15345	122	31	69	111
G	47	30	58	256	487	188	14784	76	43	23
H	7	38	117	243	428	40	174	14763	83	69
I	422	197	463	150	88	95	68	154	14133	155
J	236	102	37	53	21	138	23	37	137	15239

	A	B	C	D	E	F	G	H	I	J
A	14804	267	58	46	21	113	24	8	356	209
B	279	14872	430	142	22	33	15	28	152	53
C	42	338	14641	237	158	27	43	87	236	15
D	42	152	224	14542	551	119	246	143	100	18
E	10	14	110	524	14599	64	380	317	71	27
F	103	12	22	105	50	15543	103	21	44	83
G	33	30	35	220	428	123	15022	56	32	13
H	10	31	68	157	390	31	58	15075	90	52
I	374	146	331	98	67	79	34	108	14566	122
J	270	67	30	36	24	122	13	35	97	15329

Figure 5.9: Left - Experiment 3, Phase 1: Confusion matrix for the first MLP trained only on the feature vectors. Right - Experiment 3, Phase 2: Confusion matrix for the second MLP from the extended fusion technique, i.e. the fused feature vector coming the original feature vector concatenated with the outputs from two separately trained MLPs.

Individual improvements range from 2-4% for all classes (cf. Table 5.3). As before (cf. Section 5.2.6) misclassification rates drop around 20% - 25% for the most difficult cases (compare class E: 704 → 524 and 612 → 380 for cases D and G respectively).

5.2.8 Experiment 4 - Generalization on unseen data

This contribution presents further experiments conducted with the fusion technique presented in Section 5.2.2.2, however it includes differences in the MLP topology and an in-depth comparison of the algorithm's performance with respect to the generalization on unseen persons. The early fusion technique described in Section 5.2.8.2 further divides the training and test set to provide more independence within the data[4].

5.2.8.1 Neural network topology

The experiments were performed in a 2-step approach involving two distinct MLPs. Each network has three layers - input, hidden and output layer with the input and hidden layers fully connected to their subsequent layers. The input for the first net is formed by the point cloud descriptor of the current sample, the output layer consists of 10 output neurons, one for each class. The input for the second net differs as it not

[4] Some parts of the following sections appeared in similar form in Thomas Kopinski, Alexander Gepperth, and Uwe Handmann. A simple technique for improving multi-class classification with neural networks. *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2015

only takes into account the descriptor of the current sample but also the output values of the first net which classifies the same sample. Hidden layer sizes were determined empirically for all nets, the output layer of the second network consists of 10 neurons.

5.2.8.2 Preparation of the data sets

In order to train the two-stage architecture as described in the previous sections, the available data samples need to be divided into three disjoint sets[5]. One can distinguish between the training (D1 and D2) and generalization (D3) sets, the latter consisting of all the samples (about 30,000) coming from person $1 \leq i \leq 15$. The remaining samples are divided into the two equally sized sets D1 and D2. In order to monitor training performance, each set D1 and D2 is randomly split in a ratio 9:1 into the sets D1.train, D1.test, D2.train and D2.test. These sets are used during the training of each network to achieve a more independent measurement of the current generalization performance. D1.train is used for training the first network while the MSE measured on the D1.test set is used for selecting the best-performing net among all conducted iterations. As soon as the first net is trained, the second net is trained on D2 in an analogous manner. The main difference is that the input for the second net is formed as follows: For a sample taken from D2.train, feed this sample into the first net and memorize the output of all 10 output neurons. These values are concatenated with the descriptor (size n) of the sample, thus having length $n + 10$, and form the new training sample fed into net 2, with supervision coming again from the known class of the sample. The rest of the procedure remains the same, D2.test is used as the performance measure for the decision process of the second phase of training.

5.2.8.3 Experiments and results

All experiments were conducted with the FANN library and implemented under Ubuntu in C++. Several configurations were tested in order to determine the hidden layer size and optimal training parameters. Initial test runs have shown that the best generalization results can be achieved when setting the size of the hidden layer between 25-40 neurons for both networks. The remaining parameters were chosen as standard values and testing various configurations provided no meaningful insight regarding this problem. The training algorithm is RPROP, the activation function is the symmetric sigmoid function for both hidden and output layers.

[5] Note: Experiments in the prior sections contained more data samples, as they were conducted at a later stage than the experiments in this section thus containing data from more persons.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
.81	.43	.68	.43	.83	.49	.53	.68	.81	.70	.88	.73	.95	.66	.76
.83	.44	.67	.54	.84	.53	.55	.67	.78	.70	.91	.74	.97	.73	.90

Table 5.4: Comparison of the generalization performance of net 1 (upper row, 40 hidden neurons) vs. net 2 (lower row, 40 hidden neurons). The indexed columns indicate the generalization data coming from person i.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
.81	.40	.68	.39	.81	.48	.52	.68	.79	.68	.88	.71	.94	.64	.74
.81	.45	.67	.50	.83	.53	.52	.67	.77	.67	.90	.71	.96	.68	.89

Table 5.5: Comparison of the generalization performance of net 1 (upper row, 25 hidden neurons) vs. net 2 (lower row, 20 hidden neurons)

On average the generalization results were improved by 2-4% considering all classes, as Table 5.4 shows, when employing two nets each having 40 neurons in the hidden layer. The upper row shows the results for net 1 alone, being fed the feature vector only while the lower row displays the fused feature and neuron values. In 3 cases a slight drop in generalization performance is observable, in one single case no change is notable while all the remaining cases show partly significant improvements when fusing the data with output activities. In some selected cases there are improvement rates of 5-9% averaged over all the gestures of one person.

As a comparison Table 5.5 displays the performance of both nets each with 25 and 20 hidden neurons respectively. The main difference in this case is the fact that the nets with smaller hidden layer seem to perform slightly more poorly while one can still show an improvement with the presented fusion technique when individually comparing each generalization case. Moreover, the most significant improvements can be found in both experiments.

Table 5.6 shows the change in recognition rate in reference to the individual hand gesture classes averaged over all persons.

a	b	c	d	e	f	g	h	i	j
+1.3	+4.8	-2.1	-2.8	+3.8	+2.0	+3.1	+3.2	+4.7	+2.5

Table 5.6: The average increase/decrease in recognition rate for all 10 gestures averaged over all the persons for net 1 compared to net 2 (both 40 hidden neurons).

In 8 of 10 cases the recognition rate is improved except for gesture classes c and d (-2.1% and -2.8%). The best improvement amounts to 4.7% and 4.8% which is, for this problem, of special interest since these are two of the more difficult cases to disambiguate (cf. Figure 4.2).

Table 5.7 shows the corresponding improvements for the recognition rate averaged over all persons for all individual gesture classes. This approach is tested by randomly splitting the whole data set (i.e., not taking into account individual persons) into two equally sized data sets, one for training and one for evaluation. Since the sets were not completely independent, the overall classification rate was 86% for net 1 and 87% for net 2 which is not surprising but still shows the results are improvable with the presented fusion approach. A comparison of the presented approach with the MNIST data set yielded, for 15 different test runs conducted in the same manner (as there is no person information available), neither a significant drop nor an improvement since the classification error remained around 5.5% for net 1 and net 2 with a slight variance. This variance can be ascribed to the random initialization of the weights. It should be noted that the data set was split into three parts namely 40% for training each net and 20% for testing, which is a significantly reduced training data set. Lastly, MNIST contains a set of 60,000 training and 10,000 test samples, which is a significantly lower number of data samples compared to the hand gesture database in this thesis.

a	b	c	d	e	f	g	h	i	j
+.01	+.02	+.01	-.03	+.07	+.01	+.05	+.04	+.01	+.04

Table 5.7: The average increase/decrease in recognition rate for all 10 gestures averaged over all the persons for net 1 compared to net 2 (25 and 20 hidden neurons respectively).

5.2.9 Experiment 5 - Temporal integration of information

It could be shown in the previous sections how the fusion approach of features and neuron outputs improves the generalization performance significantly. In order to levitate this approach into the temporal domain this section demonstrates how information obtained over time can be harnessed and incorporated employing the same technique. In order to improve the recognition rates of the MLP-based approach, a temporal fusion technique is presented. To this end the training and testing data have to be prepared as follows: Firstly, training and testing data are split for each run in such a way that data taken from a single designated person is not included in the training set, thus being able to measure the generalization performance of the system on previously unknown data. Secondly, the training data has to be presented in a chronological way in order to make the fusion technique viable. The overall procedure is split into a 2-step approach. In an initial step an MLP is trained on all the data from the training set. To induce temporal information into the second MLP, the training step has to be modified in such a way that, at a given point in time t, not only the input from the feature vector is presented to the MLP, but also the values of the output neurons of the first MLP classifying the sample at time $t-1$. To achieve this, the training data for MLP 2 have to be presented in a chronological order. Therefore the size of the input layer of the second MLP is determined by the length of the feature vector + number of the output neurons of MLP 1 and for this case sums up to (625 + 10), the size of the feature vector + the number of classes[6]. This approach can be motivated as follows: During the interaction of the user with the system, multiple snapshots are taken from the camera for a single hand pose. Thus information is observed 'over time', i.e. classifications coming time points shortly before the current point in time are be used to stabilize the results.

5.2.9.1 Neural network topology and training parameters

For the described two-step fusion approach the networks are trained with standard parameters with variations in the network topology. The input for the first network is formed by the RPFH descriptor of a processed point cloud as described in Section 2.3.5. The input for the second network is formed by first classifying the previous sample with the first network, calculating the output neurons' activities, and then concatenating these activities with the RPFH of the current time step. Thus, the input layer of the second network is of size $n+10$, since both networks have as many output neurons as there are classes in the classification problem, i.e., 10. The network topologies are therefore $625-h-10$ and $635-h-10$, respectively, with variations possible in the hidden layer size h. Several experiments are conducted to obtain good values for h, which are found in the range of 30-50 hidden neurons.

The network is implemented using the FANN library [72]. The training algorithm is the standard RPROP algorithm and the activation function is the sigmoid function for both hidden and output layers.

5.2.9.2 Experiments and results

Tests on the data set comprise all 16 persons. In order to compare the approach for the regular MLP and the fusion technique, two classifications are conducted for any input at a given point in time. Moreover, the number of neurons in the hidden layer

[6] The feature generation as described in Section 2.3.5 is utilized here.

person	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Acc.
MLP1	81	50	69	55	76	59	56	68	79	68	88	72	95	72	87	79	72%
MLP2	83	51	72	60	79	62	58	72	85	74	89	73	97	75	91	83	75%
MLP1	83	49	69	58	79	62	57	68	84	70	89	75	90	74	90	80	74%
MLP2	86	52	74	63	83	65	60	72	89	74	90	75	98	75	93	82	77%

Table 5.8: Generalization results for all 16 persons and both MLPs, each with 30 and 50 neurons respectively. The upper two rows refer to MLPs containing 30 neurons, the lower two rows are MLPs containing 50 neurons in the hidden layer respectively. The last column is the mean accuracy averaged over all persons.

is varied (cf. Table 5.8) as to measure whether the performance of the presented algorithms improves. Table 5.8 sums up the most important results. Each row shows classification accuracy for an MLP trained on all the data except the person it is tested on, each test being a generalization performance test. As an example, column 1 then represents the performance of all four MLPs, trained on persons 2-16 and tested on person 1. The overall classification accuracy of an MLP - averaged over all persons - depending on the number of neurons in the hidden layer and the kind of fusion technique is shown in the last column.

First of all, it is observable that this fusion approach, denoted MLP2, outperforms the regular approach (MLP1) as the average MSE for the fused input is about 3% lower averaged over all persons. For some individual cases the results improve by up to 8% - here averaged over all gestures of a single person - as for instance for person 13 (improved from 90% to 98%) in the case of the larger MLP. Increasing the number of neurons from 30 to 50 has a negligible effect as the improvement of this approach compared to the standard MLP stays around 3% for all variations of the sizes of the hidden layer. However, it is possible to lower the average overall MSE by around 1.5% from approx. 15% down to 13%.

For those cases which perform worst in terms of recognition rate, the results still are improved by an average of 2-3% by the fusion technique. It is only for three test persons that this technique has little to no effect (persons 11, 12 and 14) as the change of MSE remains around 1% or below. On the other hand, in no case the fusion approach leads to a deterioration of performance. A more precise evaluation of the results shows that the temporal fusion approach reduces the disambiguation problem to two candidates.

More specifically, the confusion matrices (cf. Figure 5.10) show the correct classifications as well as the false positives, for all ten gestures, here presented exemplary for person 13. Each row represents the (mis-)classifications for each gesture shown

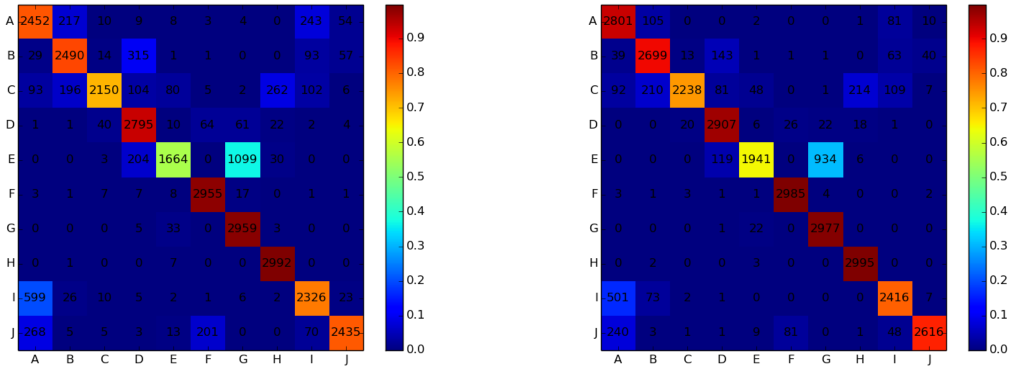


Figure 5.10: Left - Confusion matrix for the standard MLP. Right - Confusion matrix for the fused MLP.

on the left-hand border. So gesture A ('one') has been classified 2452 times (Figure 5.10, left) correctly with the standard approach and 2801 times correctly with the improved fusion technique (Figure 5.10, right). Coincidentally, the same gesture has been mistaken 217 times for gesture 'two' and this number has been lowered to 105 with the improved approach. Simultaneously the number of false positives for this gesture being mistaken for gesture I dropped from 243 to 81.
Moreover, both approaches work best for gestures D, F, G, and H nearing 100% recognition rate, followed by gesture A, B and J (around 90%) and C and I (around 75-80%) referring to the standard MLP in Figure 5.10. For this person, both approaches perform worst for gesture E ('five') being classified correctly with a precision of around 55% and 65% respectively. In addition, it is observable that this gesture is being mistaken for gesture G ('flat hand') which can be explained by the similarity of appearance of both kinds of gestures.

Furthermore, the gesture J ('point') was mistaken around 201 times (cf. Figure 5.10, left) for gesture F ('fist') before the fusion approach and 81 times (Figure 5.10, right) after fusion. This is a drop of more than 60% and it reduces the problem to disambiguating only gestures J and F from each other, which can be handled individually and would make the recognition task much easier. Similar observations can be made for gesture pairs B → D and E → D for the example depicted as well as across the group of participants as a whole.

5.2.10 Experiment 5 - MNIST

As a supplementary exercise, the approach is evaluated on the well-known MNIST hand-written digit database [53], using the fusion strategy from Section 5.2.2.2. It can be observed, without any parameter tuning, that a classification rate of around 93% is achievable which does not noticeably improve by adding the second MLP. On the other hand, no degradation of performance is observed either. What seems to be the problem here is that, being forced to divide the data set into three parts, less training examples can be used compared to other approaches, which can be the reason for the lack of improvement.

5.2.11 Discussion

This section presents a multiclass classification scheme that is intended to be useful in practical applications. As it does not make explicit assumptions about the nature of classification tasks, it contains no additional parameters beyond those that would have to be tuned any case for binary classifier training, in this case MLPs. No significant theoretical modeling of the classification task at hand needs to be performed as one relies on the capacity of the second MLP to extract those properties to the best of its capabilities. Furthermore, when observing the results, it can be noted that the *overall* improvements are modest, i.e., in the range of $\leq 5\%$. However, in an application it is often not the overall classification rate that is of importance, but the worst case, that is to say, the behavior of the classifier for specific "difficult" classes. Here, a strong benefit from using the two-stage approach is observable as the performance on some classes improves by $> 20\%$ which is highly significant in practice. Furthermore, strengthening the link to practical applicability, it can be stated that the additional computational cost of adding the second MLP is virtually non-existent, or rather, very hard to measure due to the efficient C implementation. Therefore, significant gains in applicability can be obtained at negligible computational cost, which is always an important point in practice, especially in a vehicular context where this technique is applied for the purposes of HMI.

Furthermore, it could be demonstrated how the idea can be transferred to include temporal information as well. As before, a simple MLP setup allows for improved recognition results on the presented database with the major changes to be made in terms of training and classification procedure as the order of samples has to be taken into consideration chronologically. One of the more interesting aspects is, in how far this technique can be exploited to take into account even more information coming from several time steps prior to the one considered in Section 5.2.9. No significant increase in execution time is to be expected as the main computational effort lies within training(optimizing the MLPs), however the classification of a sample lies below 1ms. In how far this technique can be exploited to include fused information from n different MLPs including n time steps is one of the questions to be discussed for further research.

The reason for the improvement as far as "difficult" classes are concerned, probably stems from the fact that those classes are very similar to certain others and thus are often confused by MLP1. The second MLP can presumably detect such confusion events by specific patterns of activated output neurons in MLP1, and correct the decision. The fact of providing the original feature vector seems to have a beneficial effect on this technique, as it is perceivable that predicting the class from information contained in the output layer alone does not suffice.

A critical point of the presented approach is its hunger for data: as the original data set is split into three parts, instead of two as is usually the case, there is less data to train the classifiers with, potentially incurring a loss of performance. Further research will lead into the direction of how to perform the presented scheme with only two data sets, one for training and one for testing. This would involve training MLP1 and MLP2, in a supervised fashion, on the same dataset, which has to be considered problematic for reasons of overfitting. Nevertheless, initial tests have shown that generalization performance is not in the least affected by this, so this avenue of research will be pursued further, possible with the aid of advanced regularization methods. After all, deep belief networks train their layers one after the other, each layer on the outputs of the previous one, all on the same training set. This methodology is tailored to the problem domain of this thesis and shows to significantly improve recognition results for an HGR system, which is the focus of the next chapter: Setting up a real-time applicable HGR system for in-vehicle use and integrating the ML techniques performing best for the underlying task.

Chapter 6

A real-time applicable hand gesture recognition system

Building on the previous findings, Section 6.1 presents the setup of the system within a car interior. The main contribution is threefold: It is shown how such a system can easily be integrated into an outdoor environment subject to strongly varying lighting conditions without the need for tedious calibration procedures. Furthermore the introduction of a modified light-weight version of the descriptor coupled with an extended database significantly boosts the frame rate and the performance of the whole gesture recognition pipeline. Finally, the introduction of confidence measures for the output of the MLPs allows, as previous results already indicate, for more stable classification results and gives an insight on the innate challenges of this multiclass problem in general. In coherence with the initial findings presented in Section 5.1, the first setup contains two ToF cameras integrated into the vehicle fusing both data streams, which is subsequently replaced by a one-camera setup, coupled with the more sophisticated fusion algorithms presented in Section 5.2.

In order to extend the developed system to allow for dynamic hand gesture interaction a simplified approach is proposed in Section 6.2.2. This method shows how recognition of dynamic hand gesture sequences can be achieved with the simple definition of a starting and an ending pose, based on a recognition module working with sufficient accuracy and even allowing for relaxed restrictions in terms of the parameter definition for such a sequence.

6.1 A multi-sensor setup for in-vehicle infotainment control

This section[1] presents the multi-camera setup of an in-vehicle HGR system, building upon the results achieved so far. As the angle is of subordinate importance, an HGR interface with two cameras positioned at 30° angle is integrated into the front console in order to demonstrate the real-time applicability of this approach. The main bottleneck of the system is the feature calculation, therefore the RPFH is implemented, providing robust recognition results under high frame rates. The database at this point contained data from ten different persons posing for 2,000 samples per gesture, yielding a total data set of 400,000 samples for both cameras.

6.1.1 System description

Two ToF Sensors are integrated into a car interior, both fixed to the centre console and connected to a Laptop with a Linux system installed, handling the computation task. Taking the previous results into consideration, a 30° angle was found to be sufficiently suitable for the setup of the cameras, in order to be able to disambiguate even the more difficult hand poses. To this end, one camera is placed in the centre of the console and the other one slightly shifted and rotated to the right from the driver's perspective due to obvious space-requirements (cf. Figure 6.1). As opposed to the previous research, focus lies on recognizing the subject's (i.e. the driver's) right hand for the desired hand poses. Therefore, a desired volume of interest is defined within which hand poses should be identified. Due to driver behaviour, possible range and convenience, as well as obstacle occlusion (e.g. steering wheel), the VOI is of trapezoidal shape with a depth of 27-35cm, a maximum width of 60cm and a minimum width of about 45cm enclosed by the FoV (Field of View) of the camera frustum. The total height covered by the cameras ranges from 30-35cm.

It is important to note that both cameras have been set up with the same parameters in order to roughly have the same distance to the recorded object as well as the same VOI at any given point t in time. Furthermore a space big enough to recognize the most important movements in the car is covered. Usually the driver has his hand on the steering wheel or close to it or, in other situations leans onto the armrest, which differs significantly in position and allows for longer interaction with

[1] The following sections appeared in this form in Thomas Kopinski, Stefan Geisler, Louis-Charles Caron, Alexander Gepperth, and Uwe Handmann. A real-time applicable 3d gesture recognition system for automobile hmi. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pages 2616–2622. IEEE, 2014

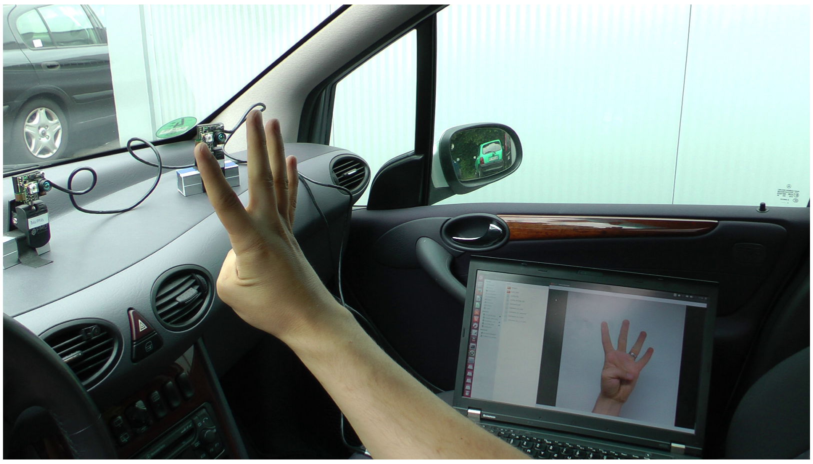


Figure 6.1: The initial system setup with two ToF sensors mounted to the front console.

the system. By defining the VOI as described, it becomes possible to cover these situations as well.

The subject is seated and positioned in the same manner as the other subjects during the recordings. The right arm is placed onto the armrest in order to be able to interact with the system properly. The lighting conditions for this experiment were the same (bright sunlight) as those for the other subjects before. The subject tests all ten hand poses in an arbitrary manner, the cameras record the environment and feed the transformed data into the NN which classifies the result. Every recognized hand pose is displayed, with the system running at a frequency of 5-6Hz.

6.1.2 Neural network classification and fusion

The descriptor of choice produced for each point cloud is utilized as described in Section 2.3.5.

With m cameras, the system produces m descriptors per frame according to the methods described above. An MLP is employed to model the multiclass classification function, either implicitly by concatenating all m descriptors followed by NN classification (”early fusion”), or explicitly, by performing classification individually on each of the m descriptors and then combining the results (”late fusion”). Neural networks contain a bias unit at each layer, the training algorithm is ”RProp”[34] with hyperparameters $\eta^+ = 1.2$, $\eta^- = 0.6$, $\Delta_0 = 0.1$, $\Delta_{\min} = 10^{-10}$ and $\Delta\max = 5$. Network topology is $m \cdot K$-16-10 (hidden layer sizes from 10-500 were tested without finding significant performance improvements), K indicating the size of the used descriptors and m the number of cameras, here $m = 2$. Activation functions are sigmoid throughout the network. As the MLP classifiers have 10 output neurons, corresponding to the 10 gesture classes, with activities o_i, the final classification decision is obtained by taking the class of the neuron with the highest output. However, not every classification should be taken seriously, and thus a simple extension based on a *confidence measure* conf($\{o_i\}$) (cf. also Section 6.1.2) is implemented. Final decisions are thus either taken by determining the neuron with maximal output, or by applying the confidence measure in the following way:

$$\text{class} = \begin{cases} \text{argmax}_i o_i & \text{if conf}(\{o_i\}) > \theta_{\text{conf}} \\ \text{no decision} & \text{else} \end{cases}$$

The confidence measure performs a mapping from $\mathbb{R}^{10} \to \mathbb{R}$ and shall be denoted "confOfMax":

$$\text{confOfMax}(\{o_i\}) = \max\, o_i \tag{6.1}$$

For performing late fusion, that is, obtaining two independent classifications o_i^1, o_i^2 based on each camera's features, the arithmetic mean of both output vectors is calculated: $o_i^F = 0.5(o_i^1 + o_i^2)$. This intrinsically takes into account the variance in each response, as an output distribution strongly peaked on one class will dominate a flat (or less peaked) distribution. The resulting output distribution o_i^F can then be subjected to the decision rule of Eqn. (6.1).

6.1.3 Experiments

The experiments were conducted in such a manner as to test the influence of the various parameters on the classification results. Various settings for the NN parameters were tested, resulting in three layers of 1250, 16 and 10 neurons - i.e. one hidden layer - for the input, hidden and output layer respectively. The number of neurons for the hidden layer was chosen empirically, as the overall performance of the NN peaked at this value during the course of the experiments.

The used NN code was taken from the OpenCV library [9] with Rprop as the training algorithm and a sigmoid activation function. The input for the network is formed by taking the input cloud and create the customized descriptor for it. Doing this for each camera and concatenating them together forms the input which is fed into the NN. This procedure is done for training, testing as well as for the live demonstration described later on.

All following confusion matrices should be read in the following way: E.g. in Table 6.1 gesture **a** was recognized 9503 correctly as gesture **a** but misclassified 87 times as gesture **b** etc.

6.1.3.1 Baseline performance

The first experiment splits all gesture and person samples 50/50 for training and classification, which leads to an overall recognition rate of 94% without even applying the confidence measure as indicated in Equation (6.1). Table 6.1 displays the number of correctly classified hand gestures, listed a-j (cf. Figure 4.2). One row adds up to ∼10,000 samples, because this is how many were included in the set to test the classification error of the NN. From Table 6.1 an overall classification error of 6.3% is obtained. A few things can be observed:

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**
a	9503	87	17	14	11	48	28	4	129	104
b	96	9470	183	109	1	36	16	10	29	30
c	145	406	8913	181	27	21	45	46	97	20
d	19	64	107	9191	241	22	138	56	8	18
e	21	15	15	191	8986	8	679	41	11	8
f	410	29	26	66	49	8971	153	34	72	85
g	13	8	8	75	30	17	9758	26	18	8
h	7	7	50	94	180	10	20	9454	54	31
i	238	30	45	17	8	25	10	12	9444	74
j	347	33	15	29	11	66	25	15	72	9193

Table 6.1: The overall classification results on the complete data set. The overall classification error is 6.3% while this varies from 2.5% to 10.8% for individual gesture classes.

- Overall the performance of the net seems to vary between gestures: most notably c, e and f are the ones on which the NN performs worst, as opposed to gestures a and g, which are recognized best - 9503, 9758 respectively out of nearly 10,000 samples recognized correctly
- For every gesture - but above all, for the ones recognized worst - the specific one is identifiable, for which it was mistaken the most as for instance gesture b is likely to be held for gesture c and vice versa; in other words, 'two' is likely to be held for 'three' and 'fist' is likely to be held for 'one'. This makes sense as these poses differ only slightly in terms of point cloud size, shape or appearance in general. Similar observations can be made for gesture pairs (a ↔ i), (a ↔ j)

- It is important to note, that while one gesture may be mistaken for another, the reverse is not always the case, at least not in the frequency (compare f $\rightarrow$ a and a $\rightarrow$ f)

6.1.3.2 Generalization to unknown persons

Table 6.2 represents the performance of the NN, trained using all persons except one, then tested on data from the person not contained in the training set. Hence each row now adds up to ~2,000 samples. The overall classification error for this experiment is about 14% - nearly twice as large if compared to the table before. A set of interesting observations can be identified here:

- The network performs best on poses f, h, i while the worst results are on pose a and e

- Similar to the experiment described above, there are poses likely to be mistaken for others - here most notably: a $\rightarrow$ i and e $\leftrightarrow$ d. While this makes sense overall (compare the hand gestures visualized in the database in Figure 4.2 for similarities), the described cases differ from the ones above and it is left to interpretation where this difference comes from. If a certain hand pose from a person differs significantly from the equivalents included in the training set, this case is prone to error

- It can be derived, that it is obviously possible to achieve good generalization performance by training an MLP on the database for the given task while the more interesting fact is that this works better for some cases that others. It is e.g. clearly observable that hand pose *a* is very likely to be interpreted as hand pose *i* which can occur due to the fact that this certain subject poses in a similar way for this specific task

6.1.3.3 Boosting by thresholding confidence

Tables 6.3 and 6.4 refer to a set of experiments conducted on the whole data set split into equal parts (~100,000 samples for training and testing each) but this time comparing the confidences to a threshold Θ, in order to classify only those samples deemed to be 'confident enough'. As before, several experiments have been run and are outlined in the following observations:

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**
a	1329	3	2	0	0	4	0	0	626	36
b	20	1710	237	11	0	3	0	4	9	6
c	7	116	1799	24	2	2	3	22	22	3
d	0	1	11	1812	145	0	1	29	0	1
e	2	0	3	349	1391	0	68	185	0	2
f	43	0	1	0	0	1900	3	0	40	13
g	1	0	2	10	251	14	1719	3	0	0
h	0	2	17	0	3	0	2	1942	27	7
i	35	6	20	2	0	1	3	22	1907	2
j	176	51	3	1	2	32	1	4	36	1674

Table 6.2: The classification results on a data set with one person excluded from the training set and entirely used for testing. The overall classification error is about 14% while this varies from 3% to 35% for individual gesture classes.

- Various confidence parameters Θ were tested for the output neurons in the range of [0.5,0.95], with 0.05 values for the steps taken; in order to exemplary demonstrate the effect, 2 sample results for the values 0.65 and 0.95 are presented

- The classified category is accepted by the MLP if the value of the highest neuron is above the chosen confidence value, else the sample is rejected. This leads to the fact that with a higher chosen confidence value more samples are rejected (6,776 rejected samples for $\Theta = 0.65$ - Table 6.3 and 34,005 rejected samples for $\Theta = 0.95$ - Table 6.4)

- The desired effect of sorting out cases which are deemed *unsure* is achieved by this parameter, compared to the previous results (cf. Table 6.1) as the number of false positives drops in all cases

- Most of the true positives are retained and improvement is visible for the overall classification error which drops to 3.63%

- The disambiguation problem remains for the most difficult cases; as an example, gesture *f* is likely to be held for gesture *a*

When applying a very high threshold of $\Theta = 0.95$ (cf. Table 6.4), the following observations can be made:

- Raising Θ close to 1 results in many samples being excluded from accepting as confident (cf. Table 6.4) as more than a third (34005) of all samples were rejected

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**
a	9343	49	10	3	4	21	8	0	86	77
b	58	9286	90	57	0	15	5	6	14	12
c	117	269	8470	95	13	9	23	21	62	7
d	6	43	27	8818	104	8	63	22	2	6
e	13	8	6	112	8584	5	573	14	5	2
f	359	15	3	29	8	8607	57	14	42	34
g	9	4	4	35	20	11	9683	15	6	5
h	2	5	27	32	88	6	6	9066	39	11
i	198	11	23	10	3	10	3	4	9278	49
j	281	14	7	9	6	29	10	8	45	8880

Table 6.3: The classification results with confidence threshold $\Theta = 0.65$, the overall classification error is 3.63% with 6776 rejected samples

- In most cases the recognition of false positives drops to (or close to) 0; those cases which are difficult to disambiguate are reflected clearly in the results
- Some classes 'suffer' more than others in terms of recognition or rejection rate, e.g. hand poses *c* and *e* are subject to many exclusions which can be interpreted in such a way as that these gestures are probably more ambiguous than others
- Fairly many samples remain as false positives in some cases (300 in the case of gesture f → a). Confidence must be high for several neurons (i.e. hand poses) in cases like this
- The overall classification error is lowered to 1.3% on average at the cost of the fact that some gestures are better recognized than others - this is validated during live tests of the system: The overall performance of the system is stabilized while recognition results fluctuate for a few examples

6.1.3.4 Improvement of generalization to unknown persons

Table 6.5 and Table 6.6 show the results of the system evaluated on data of a person excluded from the training set while employing the confidence measure. Exemplary results for the confidence parameters $\Theta = 0.75$ and $\Theta = 0.85$ are shown (cf. Table 6.5 and Table 6.6 respectively). Analogous to the experiments before, the following observations can be derived:

- The effect of introducing Θ when generalizing on unknown data can be described as similar albeit there exist slight differences, namely the tests are made on a

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**
a	7104	3	0	0	1	4	1	0	8	14
b	13	7655	4	1	0	2	1	0	0	1
c	112	50	4242	9	0	0	2	0	9	1
d	3	4	0	5957	5	0	8	1	0	0
e	9	0	0	12	4120	1	114	2	1	0
f	300	2	0	1	0	5843	4	0	2	7
g	8	1	0	3	0	3	8766	3	1	0
h	0	0	1	3	3	0	1	6336	7	0
i	38	0	0	1	0	0	1	0	7623	9
j	51	0	0	1	1	0	0	0	5	6633

Table 6.4: The classification results with confidence threshold $\Theta = 0.95$, the overall Classification error is 1.31% - 34005 rejected samples

smaller sample set which quickly leads to a significant drop in the recognition of true positives. Overall the system performs well offline - though slightly worse compared to when employing the whole database.

- Error rates close to 0 (or 0) are achieved in most of the cases with a (comparatively) small choice of Θ (cf. Table 6.5)

- A good choice of Θ is crucial for the classification task as too many samples may be ruled out (cf. gesture g in Table 6.6) too early which leads to poor performance results in some cases.

- Equally good classification error rates are difficult to achieve due to the fact that too many samples are rejected (more than 65% in the case of $\Theta = 0.85$) which makes the system too unstable for some cases. Results are averaged over time, hence it remains acceptable to expect some false positives - which are then ruled out by a simple maximizing function - than to increase the confidence in such a manner as that too few samples are included. The latter case results in some hand gestures nearly not being recognized at all.

- Averaging results over time does not help if Θ is chosen too high. Best results are achieved for the system as a whole with Θ ranging in [0.6,0.8]

- An overall error measurement has to be taken cautiously, as the number of true positives varies significantly in between the classes *a-j*

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**
a	429	2	0	0	0	0	0	0	333	0
b	1	1119	17	0	0	0	0	0	3	2
c	1	53	919	2	0	0	0	6	4	0
d	0	0	0	1402	8	0	0	0	0	0
e	0	0	0	261	403	0	3	33	0	0
f	1	0	0	0	0	1084	0	0	0	0
g	0	0	0	1	3	1	470	0	0	0
h	0	0	17	0	2	0	0	1727	12	2
i	0	0	0	0	0	0	0	0	1569	1
j	47	0	0	0	0	1	1	0	4	474

Table 6.5: The classification results with confidence threshold $\Theta = 0.75$, the classification error is 7.89% with 9560 rejected samples

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**
a	115	0	0	0	0	0	0	0	195	0
b	0	759	4	0	0	0	0	0	0	1
c	1	23	143	1	0	0	0	1	1	0
d	0	0	0	951	2	0	0	0	0	0
e	0	0	0	96	216	0	0	10	0	0
f	1	0	0	0	0	940	0	0	0	0
g	0	0	0	1	0	1	12	0	0	0
h	0	0	2	0	1	0	0	1507	4	1
i	0	0	0	0	0	0	0	0	1154	1
j	16	0	0	0	0	0	1	0	4	203

Table 6.6: The classification results with confidence threshold $\Theta = 0.75$, the classification error is 5.78% with 13610 rejected samples

6.1.4 Conclusion

The system was evaluated on a Laptop with an Intel i7-3820 QM CPU running at 2.7GHz on one core with 8GB RAM under Ubuntu. With two cameras, up to 6Hz are achievable for this task, with the main bottleneck being the feature generation depending on the size of the point clouds. Additionally, snapshots have to be taken alternatively from each camera to avoid interferences. The introduced thresholds have negligible influence on the computation time.

Test results are in general comparable to the offline system although an exact measurement is not possible as annotations were not yet created for the real-time in-car scenario. Based on a visual inspection of the results, very few errors are observed if a subject's hand gestures are a part of the training set, otherwise performance suffers but not beyond a certain point. The recognition rate in the latter case is impeded by the fact that many of the chosen poses are difficult to disambiguate (see the used gesture alphabet in Figure 4.2), e.g., hand pose 'two' vs. 'three' or 'L' vs. 'one'.

It is possible to boost the performance of the real-time system by using the "confOfMax" confidence measure as in the offline experiments. Confidence thresholds in the range of $[0.6, 0.9]$ yield satisfactory results, by determining whether the system is 'sure' enough to recognize and classify a sample. Of course, a balance has to be found between the need to have very accurate results, and the need to maintain a sufficiently high frequency of results. As a satisfactory frame rate is achieved which can conceivably be boosted by optimizing the code, it is acceptable to reject half of the incoming samples in exchange for a high recognition accuracy.

Building upon prior results, the hand gesture recognition system presented in this section demonstrates the fusion of multiple sensors is capable of boosting overall classification results while retaining its real-time capability. Tests on an offline database show excellent generalization performance for the set of 10 static poses. In particular, what can be shown is that already a computationally simple confidence measure boosts performance considerably at the cost of ignoring uncertain samples, which is strongly preferable to proposing an incorrect classification. Moreover the system itself is easy to be installed into a scenario such as the car interior without the need for extensive camera calibration. This is a benefit as a test setting can be quickly set up for any kind of intended simulation which is desirable e.g. for the field of HMI as will be seen later on. The only presupposition made is that the cameras should be pointing towards the designated VOI at an angle of about 30° towards each other.

In order to further improve the computational complexity as well as the recognition rate of the system each or all of the following measures can be taken: Reduce the number of sensors, find an improved feature calculation, integrate more sophisticated ML algorithms or improve on the decision making technique. At the time being, the main bottleneck is the feature calculation method which, for this contribution, was already improved in order to approach real-time frame rates. The employed histogram suffers in terms of descriptiveness if the number of subsampled points is chosen too low but is too costly to be calculated if chosen too high thus it remains the means of choice for the remaining part of this thesis as no other descriptor was able to meet the demands for the task at hand.

In order to compensate for reducing the number of sensors, and thus losing a significant amount of information, the following section explains the measures taken by employing the already discussed novel ML fusion approaches. Based on these new features, a simple light-weight method for recognizing dynamic gestures from static poses is introduced.

6.2 Dynamic hand gesture recognition with a single ToF camera

This section describes the improvements made to the system described in the previous section. The main novelty is the realization of a system capable of recognizing static and dynamic gestures from a single ToF camera implementing the ML algorithms developed and tested in Section 5.2. Moreover the visualization of the system's behaviour is now realized on an iPad allowing for improved user feedback and more realistic testing environments.

6.2.1 In-car infotainment application

The framework was connected to a tablet running a simulated infotainment system as can be seen in Figure 6.2. The camera is mounted on the front console and connected to a standard laptop with Ubuntu running the recognition pipeline. The laptop in turn is linked via a wireless interface with the tablet simulating an infotainment system with switchable audio channels, a USB and CD interface as well as simulated incoming phone calls. The user can utilize the standard gesture alphabet to interact with each of the functions being able to control the system with her/his right hand during a car drive.

6.2.2 Dynamic hand gestures

Hand gestures are defined as being dynamic, i.e. changing in state over time, and they can be contrasted against static hand poses which in turn do not change in state. Therefore, in an in-car infotainment system, a static hand pose as the *one* pose (cf.Figure 4.2, hand pose 'a') could be connected to selecting the first audio channel while a dynamic zooming in/out gesture could be applied in a typical maps application. This approach makes use of the simple fact that a dynamic hand gesture must have a clearly distinguishable starting pose and a clearly distinguishable ending pose. Consequently a 'grabbing' movement can be defined by starting as hand pose 'h' and ending as hand pose 'f' in the given hand pose database. This is a clearly defined feature and serves as a universal definition in that any kind of dynamic gesture can be captured in this sense. The number of theoretically definable gestures therefore sums

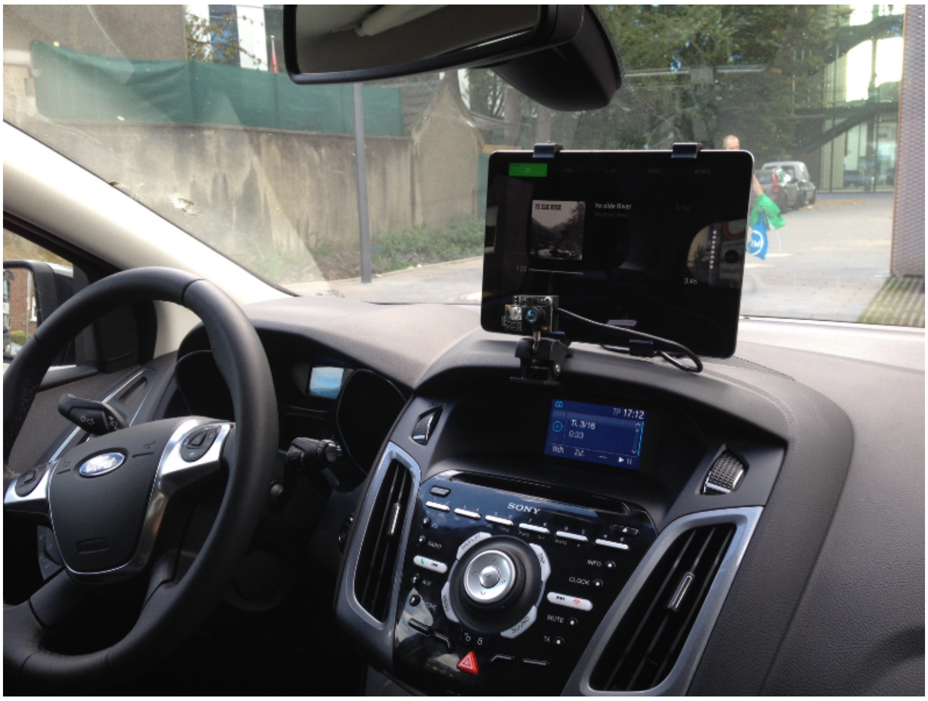


Figure 6.2: Complete system: sensor, laptop and tablet mounted to the front console visualizing the infotainment system. The main improvement for this system lies in the reduction to one camera only and, on the algorithmic side, in the novel ML methods introduced.

up to $n\dot{(}n-1) = 90$ gestures for the given static database, as any case is bidirectional i.e. a gesture from 'a' to 'b' can be performed vice versa.

A static hand pose is denoted as a state s at any given point in time t: s_t. A sequence of n occurrences of a certain hand pose is defined as $< s_{t=0}, ..., s_{t=n} >$. During the interaction phase the gesture recognition module takes consecutive snapshots which are interpreted by the system via a voting scheme. For a series of 10 consecutive snapshots, a static hand pose is recognized by the most frequent occurrence within this series if the occurrence is above a certain threshold. In order to take into account that the frame rate can vary between 5-20Hz, a threshold of 7 yields satisfactory performance in terms of recognition rate and user acceptance as the feedback has to be provided to the user and in order to suppress too frequent changes[2].

This is used as a basis for defining dynamic hand gestures within this time series as follows: For a dynamic gesture any occurrence of the starting state at any given point in time s_t^{st} followed by any occurrence of the ending state s_{t+m}^{en} with $m \geq 1$ within the observed time series, corresponds to the classification of the sequence as containing the

[2] This value was retrieved empirically through a number of experiments. The main improvement stems from the reduction to one camera only.

dynamic gesture: $< s_{t=0}, ..., s_t^{st}, ..., .s_{t+m}^{en}, ..., s_{t=n} >$. This the most simplified notation which allows for the fact that misclassifications may occur in between the detection of the starting state and the detection of the ending state. The only condition being made here is that both classifications must occur within a certain time frame and that the starting state must be detected before the ending state.

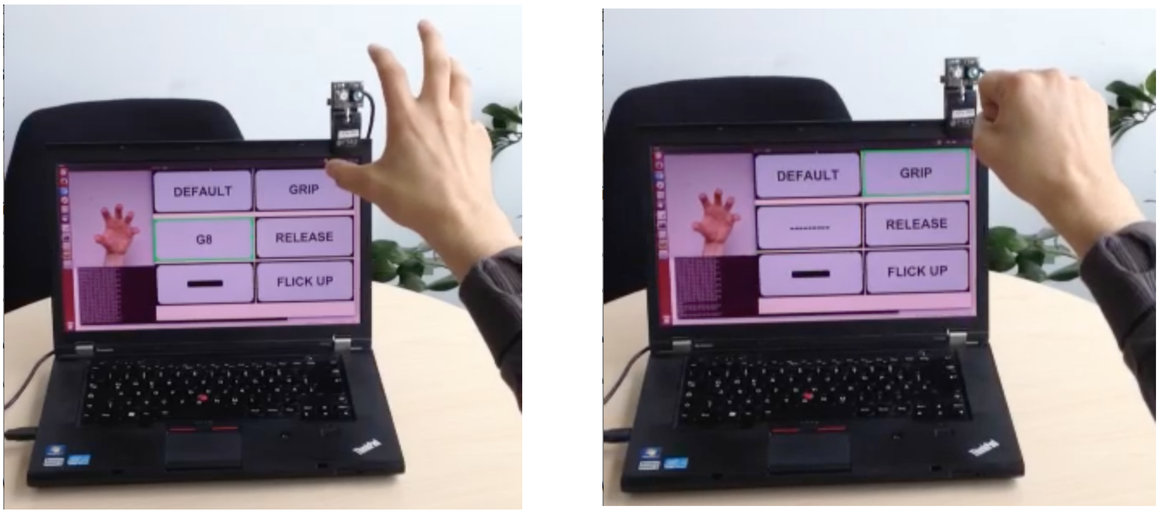


Figure 6.3: Demo setup: The user performs the grabbing gesture defined by the starting pose 'h' (left) and the ending pose 'f' (right) (cf. Figure 4.2 for notations)

In order to stabilize recognition results, a simple extension of the definition above can be made. As soon as one occurrence of a starting state is determined, this starting point of a gesture is only taken as valid if it is immediately followed by one or multiple occurrences of the same state, i.e. $s_t^{st} = s_{t+1}^{st}$. The same rule can be applied for the the ending state of a dynamic gesture. The restriction that these consecutive occurrences of states must form uninterrupted subchains within the observed time frame suffices to define a robust dynamic gesture recognition pipeline, recognizing dynamic hand gestures well in real-time as one can see in Section 6.2.3. Of course, a proper choice of parameters is immanent and strongly depends on factors such as frame rate, classification rate and user feedback. The choice of the length of the observed time frame restricts other parameters as the length of the subchains for starting and ending sequences. As a system reaction time of 150-200ms is already perceived as a delay, this poses an additional hurdle for the integration into the whole in-vehicle system. The benefit of this simple definition is the fact that one allows for uncertain states or even misclassifications to occur in between starting and ending sequences of a dynamic gesture as well as within the time frame as a whole. Additionally, as was found out during the testing phase, this approach provides extra flexibility since every user has a different way of performing a gesture and thus an otherwise more restrictive definition of a dynamic gesture would be too obstructive.

6.2.3 Experiments and results

The first problem to define is the sensible area or volume of interest (VOI). As opposed to systems working with 2D gestures the user has no way of knowing whether she/he has entered the sensitive area or not. To this end the approximate VOI was described to each user and they were to interact freely. For each recording the user was asked to enter the designated VOI and perform the corresponding gesture.

A series of tests were conducted with 10 different persons whose data is not contained in the database, i.e., the results given in Table 6.7 show the generalization performance of the system to previously unseen persons. To each person the functionality of the system was explained and a GUI was realized containing the system's response whether a gesture was recognized or not. For the experiments, a time frame of length $n = 10$ is defined, meaning that from the moment user input is generated the last 10 consecutive snapshots are observed along with the corresponding classifications in order to determine whether a dynamic gesture is contained or not. It was determined that for a time frame of this size, two identical consecutive starting poses and two identical consecutive ending poses suffice to efficiently detect a dynamic gesture.

Four gestures were defined to this end: Grab, release, zoom in and zoom out. Each gesture can be defined via an unambiguous static state from the database. The grabbing motion (shown in Figure 6.3) is defined as starting with hand pose 'h' and ending in hand pose 'f'. The corresponding release gesture is the exact inverse starting with hand pose 'f' and ending with hand pose 'h' (cf. Figure 4.2). The pinching/zooming gesture is defined analogously and can be seen as the same gesture known from pinching/zooming in 2D in e.g. a typical maps application. To this end, zooming out is defined as starting with hand pose 'i' and ending with hand pose 'f'. Consequently the inverse movement from 'f' to 'i' defines the zooming in gesture (cf. Figure 4.2). All users found the concept easy to grasp and interacted with the system by this means naturally.

The overall classification rate is 82.25% averaged over all persons and gestures. There is a 100% recognition rate for zooming in, followed by 90% for zooming out, 80% for release and 59% for grabbing. This shows that with a robust detection mechanism for static hand poses this approach resembles a viable solution. However, misclassifications still occur as the data recordings show in Table 6.8 for all hand

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
grab	10	6	3	5	5	8	6	7	5	4
release	7	6	9	8	9	8	8	9	9	7
zoom in	10	10	10	10	10	10	10	10	10	10
zoom out	9	10	7	10	9	9	10	8	9	9

Table 6.7: Generalisation performance of the system for all ten persons.

gestures, although this amounts to only a few cases as the statistics show. In the case of zooming in/out, the misclassifications sum up to 16 and 3 cases respectively, which in turn makes up for 1% or 0.1% of all the cases. For grab/release the numbers are higher, namely 151 and 78 misclassifications respectively which in turn makes up for 10% and less than 5% of all classifications. Comparing these number to the figures in Table 6.7 helps explaining why the individual gestures perform more poorly as it seems evident that more misclassifications of static hand poses impair the performance of the system. However it also shows that misclassifications are allowed to happen while a gesture is still recognized correctly, which shows the flexibility of the presented approach. An in-depth analysis of the recordings reveals that misclassifications occur in 153 cases of all the correctly recognized gestures performed by the participants within the time frame and between starting and ending sequence. Nevertheless the approach helps to remain robust by dismissing these samples. This shows that such a simple definition of a dynamic gesture is able to provide a satisfactory and stable performance under challenging conditions in real-time. As these statistics also indicate, users tend to remain longer in the final positions of a gesture, nearly 3-4 times longer in some cases (cf. Table 6.8). Hence, e.g. in the case of 'grabbing', the number of detected ending states (state 'f' - 1160 samples) is more than 3 times higher than the number of detected starting states (state 'h' - 338 samples). Why that is the case is subject to further analysis but it helps to provide further stability mechanisms for the problem at hand.

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**sum**
grab	24	0	3	4	0	1160	13	338	107	0	1649
release	0	0	4	11	0	489	4	1126	59		1693
zoom in	5	0	0	0	0	875	0	11	669	0	1560
zoom out	1	1	0	0	0	308	1	0	1361	0	1361

Table 6.8: All hand pose classifications summed up over all persons and sequences.

The individual results per person differ strongly. Person 1 seemed to be at ease with the system as only 4/40 gestures were not recognized while the result for person

3 is the weakest with 11 misclassifications in total. What seems interesting is the fact that, although the gestures are defined inversely, the results differ significantly. In the case of zooming in/out there is a 10% drop in classification performance and for the release/grab gesture this sums up to about 20% in misclassification rate. The former result can be interpreted in such a way as that misclassifications influence the overall results and may interrupt a sequence leading to mostly no classification at all by the system. The latter is more difficult to explain, but the data recorded suggests that for most persons it is easier to grasp the notion of a releasing gesture than that of a grabbing gesture as evidently most users did not finish the movement to the final state and thus the system was unable to determine whether the motion was finished or not. Most users left their hand half open and in their mind had already finished the movement. This problem can be easily fixed by a relaxed restriction of the underlying definition or by simple user guidance or training.

The average sequence length for performing a gesture is 16.34 which shows that defining the time window of length n=10 absolutely suffices for the intended purposes. One has to take into account that each input cloud above a certain threshold (w.r.t. the point cloud size) is taken as input. This on the one hand suppresses noise and unwanted behaviour and on the other hand stabilizes overall results as 'meaningful' input is favoured. The average time the users needed to perform a gesture was about 1.5-1.7 seconds averaged over all gestures and persons.

However, there still needs to be conducted more research on why misclassifications occur and the figures differ for individual cases although the gestures defined are closely related.

6.2.4 Discussion

The experiments show that results may vary for different persons as gestures are not easily definable in a general way but the parametrization presented in this section builds a good starting point for further research. An optimal configuration with respect to e.g. the duration of a gesture very much depends on the given task, the setting and not least on the personal preferences of the user which may explain the partly large differences in classification rate for similarly defined gestures. Improving on the frequency of the system is the main goal as it is desirable to have an HGR framework appearing 'delay-free' to the user. This would require achieving frame rates of 50-100Hz for the current parameter setting, leaving mainly the feature calculation process for adjustment with the time window for the dynamic HGR recognition becoming a subordinate role. The system described so far was completed by designing an interface on a tablet PC and set up as a whole for an in-vehicle user study to research the HMI-related question of how users would interact with an infotainment system when left with as little information as possible. The next chapter deals with this question and the results achieved.

Chapter 7

Building a gesture-controlled Natural User Interface for in-vehicle infotainment control

To what extent are the expectations of users met when interacting via freehand gestures in an automotive environment? This is the main idea researched in Section 7.1 and it revolves around the notion of intuitiveness as this work also intends to shed some light onto the HMI-related question whether intuitiveness can in any way be analyzed with the scope of hand gesture-based interfaces.
Afterwards, a summary of the results presented in each section rounds up the most interesting findings, analyzes the advantages and disadvantages of the underlying approach and gives an outlook on future work building upon these results.

7.1 A multimodal freehand gesture recognition system for natural HMI

Touchscreen systems with smartphone-like gesture control are currently getting more and more popular for in-car infotainment systems, though their position in the cockpit is only a compromise between the optimum in respect of the field of vision while driving and a comfortable place in the reach zone of the driver. In this chapter it is investigated, if the introduction of freehand gestures in combination with a touchscreen control for system features allows to remain operable while driving without the need to physically contact the screen. This allows positioning the screen close to the line of sight. The whole system is embedded into an automotive environment and used as a test setting to verify the user acceptance of gestures of this multimodal approach via an infotainment system realized on a tablet PC. In a three-staged user interaction and observation phase, high user acceptance rates are noticable once the gesture set is known to the individual user.

Designing interfaces for HMI in a natural way is a highly desirable goal as interacting with a system in an intuitive manner results in positive feedback and high user acceptance. Especially when driving a car the intuitiveness of a system, meaning a minimum effort for cogitating about the user interface, helps to reduce the cognitive load for the secondary task and allows to concentrate on the primary one, the driving task. Therefore in this field of application intuitiveness is one key element for safety and needs to be considered in every HMI concept.

While gesture-based user interfaces are already well-established for touchscreen devices, mainly in the mobile field, the rise of the even more natural contactless gestures has just begun with entertainment devices like Smart TVs and the Microsoft Kinect (see e.g. [112] for latest results from Microsoft). It is on the one hand the UI design which has to be considered carefully, on the other hand the interaction concept itself which may innately carry a large part of intuitiveness. This section presents a self-developed gesture recognition system for static hand poses and dynamic hand gestures, for which the performance in an automotive environment is evaluated. This permits to lay the groundwork for the investigation of user acceptance of such a system while being able to carefully choose the influential surrounding parameters.

The hand gesture recognition framework developed in this thesis is based on a single ToF sensor allowing to embed the system into any kind of environment regardless of the lighting conditions. The recognition task is done by sophisticated high-dimensional feature transformation and ML algorithms allowing for defining static hand poses as well as dynamic gestures. User feedback is presented on an infotainment system realized on iPad where simple channel selections as well as accepting/declining incoming phone calls is possible.

In a three-staged study the question of how such a system is accepted by various kinds of users is evaluated. The tablet itself is touch enabled with the well known interaction concepts, but the participants in the study were asked to interact via freehand gestures. This allows for a validation of the acceptance of the new interaction concept which extends the well known user interface to a multimodal approach and gathering some interesting observations regarding the question of how much a freehand gesture-based system can contribute to the feeling of naturalness in HMI.

Modern vehicles are equipped with a multitude of infotainment features, starting with classic radio, digital radio (terrestrial or satellite), media from CD, USB, Bluetooth connected devices, cloud based audio streaming, sat navigation systems, web based information services, communication via phone, text messaging, social networks and many more. These features are typically combined in a central entertainment system controlled either over touchscreen (approx. 8-10 inch display) or a controller-based system combined with a high mounted (approx. 8 inch) display.

Multimodal interaction concepts in vehicles is mainly focused on adding voice control and text-to-speech output to these systems. Nearly all major manufacturers offer these options. The goal is to reduce driver distraction and increase safety, which for example Ford boils down in several advertisement campaigns to the essence "Eyes on the road. Hands on the wheel". The manufacturers base their statements on in-house studies measuring the time, drivers were not watching the driving scene, but the infotainment display. In addition they state a reduced reaction time, as critical objects were seen faster.

Also several extensive neutral scientific studies were performed to evaluate the quality of driving (the primary task) when executing a secondary or tertiary task, like interacting with the entertainment system [90]. The results are somehow in-line with the OEM studies mentioned before, but also make a clear statement that the cognitive load is not reduced when replacing button and screen-based interaction with voice commands. Therefore performing secondary and tertiary tasks is still a safety issue. The approach of presenting a system controllable via freehand gestures brings an additional interaction modality, having in mind that human-machine-interaction is highly multimodal and getting closer to a natural way of interaction has the potential to reduce cognitive load when controlling infotainment systems. But therefore, a high level of intuitiveness of the gestures is required.

Jef Reskin [91] made the strong statement of intuition being exchangeable with the term familiarity and gave examples of how seemingly intuitive devices as the mouse were not intuitive at the time of invention to a user having not encountered this device before. However there is a plethora of research being carried out in the field of HMI connected to the use of hand gestures which calls out for intuitive gestures without really taking into consideration what this term means. Especially the choice of a certain gesture set is questionable when developing a new HMI system. Soutschek et al. [113] develop a system for freehand interaction within the medical field without stating the reason for the choice of gestures. Nielsen et al. [71] present two different approaches to developing gesture interfaces, claiming that intuitive is one of the core points of the human based approach without actually putting more emphasis into the definition of this term.

Pfleging et al. have developed a similar approach [85]. In their work classical controls are combined with voice control and gesture-enabled touch areas. The perceived usability and perceived task load of their approach are similar to the base line with classical controls only. With this in mind this work is focused on starting with observing the users and learning the freehand gestures they expect for certain functionalities in a multimodal approach.

Reeves et al. state general rules for an overall multimodal system design [93]. They will be an additional input for future work when designing a complete system based on the results presented in this paper. Emphasis is put on consistency, here between touch and contactless gestures, as this will also be seen in the results.

7.1.1 System setup

The benefit of choosing the car environment is that a user can easily relate to a variety of use cases. Some of the most frequent scenarios in the car include changing the radio channels or addressing a certain audio source as in this case from USB or the CD drive. Therefore these cases were included into the setup and designed the top row in such a way as that these functions are quickly and directly recognizable by the user. Thus the top row icons resemble a USB device, a possible CD inserted and radio channels from one to three.

User feedback is provided by highlighting the active icon in green as opposed to the inactive icons in black. Further user feedback is provided as the currently chosen audio channel is depicted by a large icon in the main display containing a dummy progress bar and the title being played. When one of the channels is chosen, the audio file starts playing in order to give even more user feedback. This is considered as a feature nice-to-have, although not must-have, as the tests are performed in a non-time-critical scenario - the car resting in a parking lot - in order to focus more on the primary goal, namely the question of intuitive interfaces. Therefore questions such as driver distraction are postponed to forthcoming research analysis. These features are considered as most relevant to a general application during a real test drive scenario - both audio and clearly visible user feedback is important.

In the given scenario in an automotive environment, a ToF sensor was mounted onto the front console. It is recording the nearby user environment (see Fig.7.1) and is connected to a standard laptop running the Ubuntu OS. This setup is necessary to record the users hand input only. It is responsible for recording the VOI acquired by the ToF camera and cropping it accordingly so that the user input is reduced to the relevant body part. The resulting sensor data, i.e. the point clouds, are processed

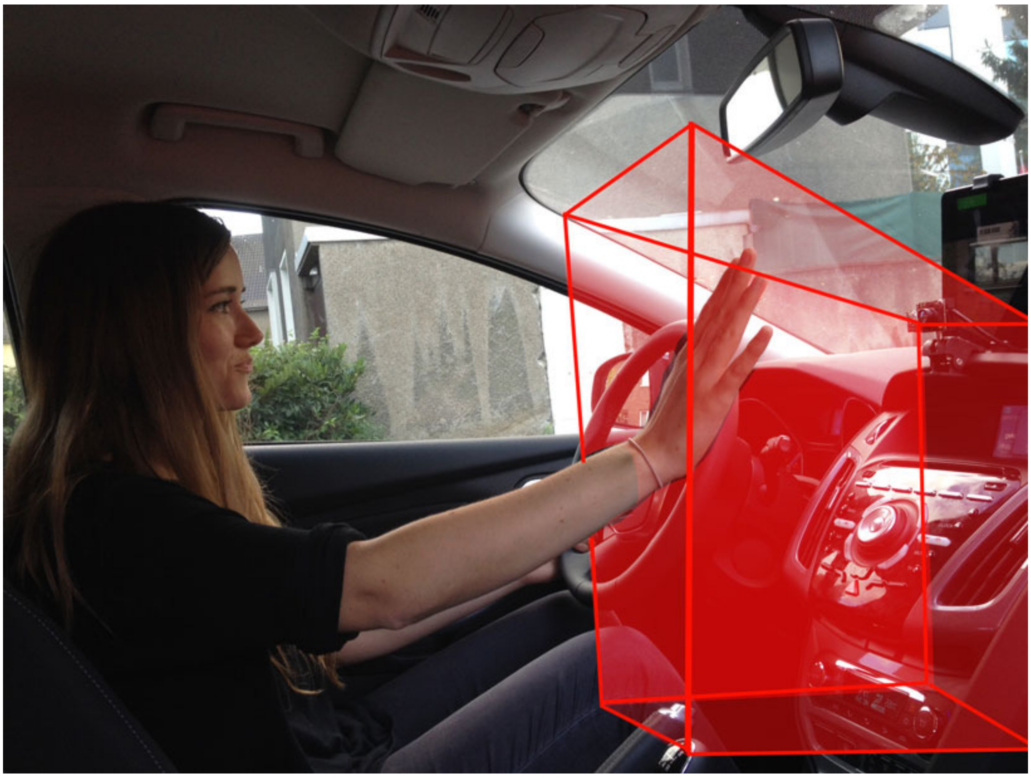


Figure 7.1: The driver interacts with the infotainment system, the sensitive VOI is marked in red.

with a frame rate of up to 20Hz. Unnecessary parts of hand and forearm is cropped via PCA. The input is transformed into a histogram capturing the geometry of the hand by the point cloud normals (see Section 2.3 for more details) and a cascade of MLPs is used for training and classification of the sample. The classification module makes several assumptions and uses a binning technique to interpret the most likely candidate for a given hand pose. Dynamic gestures are deduced from these bins via a state sequence definition. Any hand pose detected is transferred to the mobile device which interprets the user input as e.g. switching the channel or declining an incoming call. The infotainment App was realized on an iPad air under iOS and is responsible for the visualization and the user feedback. Data transfer is done by an ad-hoc network via a local hotspot on the laptop. The system described as such in this paper is easily transferable into the automotive environment for e.g. realizing quick test scenarios.

The iPad with the designed UI was mounted to the front window and placed slightly above the front console. This raises the question of what might be the most frequently used functions in such a car and reduced the complexity of the system accordingly, since a typical modern car environment can comprise up to 40-50 different functions or more, most of which are not used during the average drive with a car. Moreover the positioning of the iPad reflects a compromise of visibility and functionality as it does not occlude the vision for the driver too much during a hypothetical drive while still remaining in the driver's field of view just enough to properly display the user feedback.

Additionally, to the features mentioned above, a simulated call coming in was implemented, during the which the user was able to react to by either accepting or declining it depending on the given task. Finally the user was also able to dismiss the call after being done with the interaction process. All the described functions have been linked to an individual hand pose.

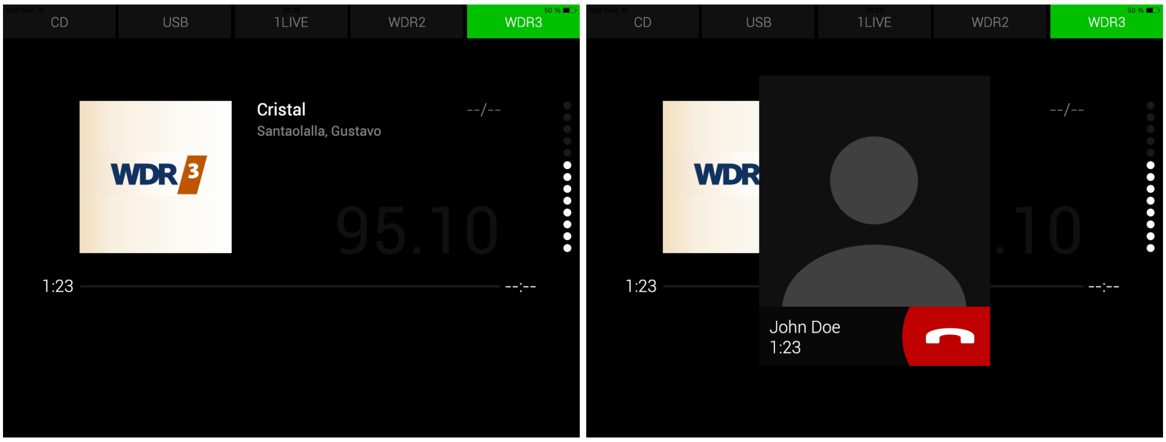


Figure 7.2: Infotainment system: Multi-channel interface (left) and the same interface during an incoming call (right)

7.1.2 Key question - what is intuitive?

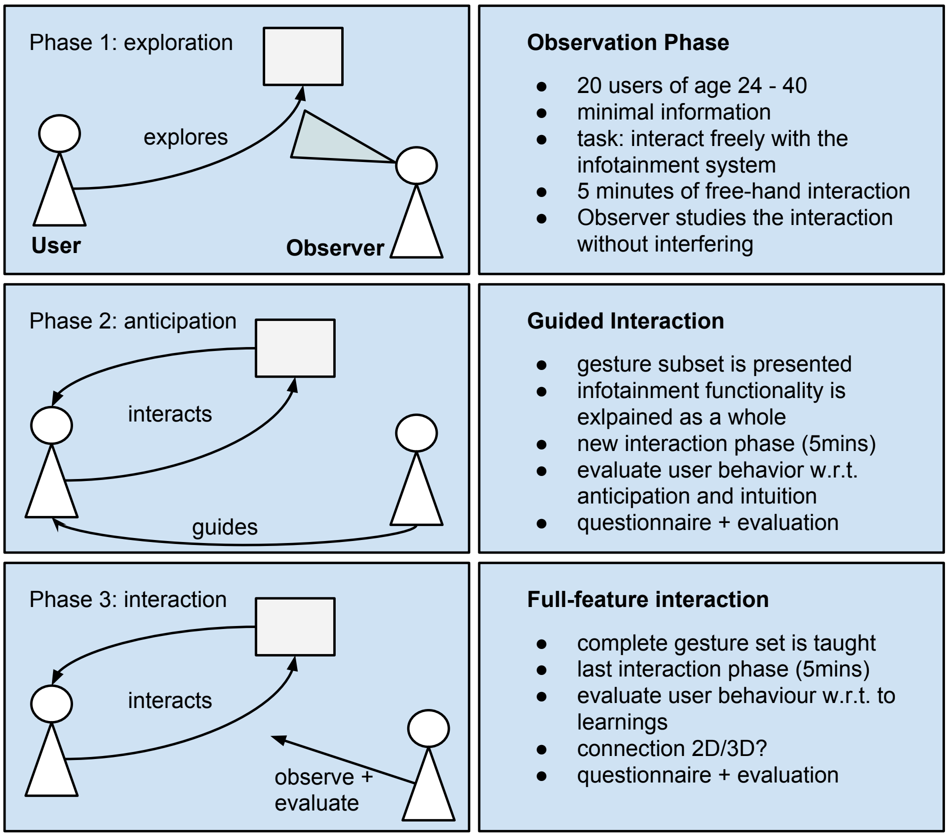
The use of the term *intuitive* has become somewhat haphazard over the course of the years. There is no proper objective measurement for it as one traverses highly subjective notions when trying to put it into any kind of context. Many definitions make use of the term *instinctive* in order to circumscribe it which is neither objective nor accurate in any way. Sometimes the 'gut feeling' is taken into account when trying to explain intuition. In terms of gestures this can be understood as some innate feature which feels right when coming to use in any kind of context of the large field of HMI. The author also believes that there is some notion of intuition strongly connected to gestures as a means of control, therefore the aim is to find any cues related to this question.

Many systems are designed with the desire of being highly intuitive to interact with for the *average* user. Such a feature becomes ever more important in an interaction scenario as e.g. a driving scene which is highly safety critical. Intuitive interaction with e.g. the infotainment system not only reduces driver distraction but also provides increased feeling of safety. In this work, intuitive actually means the user has full and most of all easy accessibility to the controls of the system, with only little overhead, to achieve a rewarding *transfer effort*. It became observable, this goal cannot be achieved by the way this infotainment system is set up - controllable by freehand gestures - but, at the same time, a simple teaching phase was able to lead to satisfying results.

7.1.3 User study

The study comprised 20 users of age between 24 - 40 years, 12 persons being male, 8 female and all persons being right-handed. All participants are frequent smartphone users for at least 5 years and none has had experience with freehand gesture interaction. The study was conducted in three steps which are described in Figure 7.3. In Phase 1 - the exploration phase - none of the features to the participants were explained, except that the visible system is a replacement for the infotainment system controllable by freehand gestures. During this phase which took about 5 minutes the observer watched the behaviour closely while taking notes and not commenting on any of the interaction steps. In Phase 2 system functionality was explained couple with a demonstration of a single hand pose. It was explained that the present system is a reduced version reacting to static poses only. The user was left to interact with the system for another 5 minutes. User acceptance was evaluated via the INTUI questionnaire (cf. [121]). In phase 3 the full gesture set is presented to the user followed by another interaction and observation phase of 5 minutes. Again, user acceptance is evaluated via another INTUI questionnaire.

Within the given context, the transfer of learning is defined as an autonomous performance made by the participant during which she or he is able to successfully transfer learned knowledge onto a previously unseen task or action. This happens by assumptions made by the user based on the knowledge acquired so far. As an example, it is assumed that a participant who is taught the 'one' gesture which is associated with the leftmost button in a row of five buttons will be able to deduct that as there are more elements aligned in the same row there is a connection between the numbering on the interface and the numbering by gestures - i.e. from one to five. Thus in this case, after learning the 'one' gesture and receiving the user feedback - namely the highlighting of button one - the transfer of learning would be making the



Observation Phase

- 20 users of age 24 - 40
- minimal information
- task: interact freely with the infotainment system
- 5 minutes of free-hand interaction
- Observer studies the interaction without interfering

Guided Interaction

- gesture subset is presented
- infotainment functionality is exlpained as a whole
- new interaction phase (5mins)
- evaluate user behavior w.r.t. anticipation and intuition
- questionnaire + evaluation

Full-feature interaction

- complete gesture set is taught
- last interaction phase (5mins)
- evaluate user behaviour w.r.t. to learnings
- connection 2D/3D?
- questionnaire + evaluation

Figure 7.3: Interaction scheme in three phases

’two’ gesture to access button two. The claim is that there is a connection between the transfer of learning and the intuitiveness of a system. More specifically, the lower the effort for the user to make such a transfer of learning the more likely a certain mechanic becomes perceivable as ’intuitive’. This assertion is corroborated by the experiments.

The experiments were evaluated with the INTUI questionnaire (cf.[121]) in which users have to rank the answers to questions mainly concerning usability and intuitiveness on a Likert scale from 1-7. The questionnaire focuses on the situation during the experiment and in retrospect. Furthermore, the questions are divided into subcategories as e.g. ’intuitive use’ or ’use guided by reason’. Selected results of the statistical analysis are presented Tab.7.1 as scores averaged over all participants. It is noteworthy that low scores for ’gut feeling’ and ’intuition’ are good results while high scores for the remaining represent are good results. This depends on the way the questions are formulated to the participant.

Since none of the participants have had prior experience with interaction by freehand

	Mean	Std dev
Effortlessness	3.98	0.93
Gut feeling	2.65	1.15
Verbalization	6.967	0.01
Magic experience	4.85	0.79
Intuition	3.2	1.9

Table 7.1: The average scores for selected INTUI questions

gestures the score for the interaction was perceived as 'magical'. It took all participants some time to become familiar with the way of interaction which is reflected in a slightly above-average score in effortlessness. Nevertheless, after some time all participants felt at ease with the new way of interaction which can be corroborated by the fact that the gut feeling and intuition scores show definite positive tendencies. All participants were able to verbalize what they were doing in retrospect and were aware of how to interact with the system and how to conduct the test runs. All participants said that after some time getting used to the system they had no problem interacting with static gestures at all. At this point, most of the users were able to perform a certain transfer of learning as they established a connection between touch gestures and freehand gestures which is detailed and explained in the observations.

7.1.4 Results

All subjects started using dynamic gestures in order to interact with the system. It was assumed beforehand that people might be tempted to 'intuitively' use gestures they were trained to use every day, when interacting with their smartphones or tablets and thus make the smallest effort to achieve a rewarding transfer of learning. Naturally all subjects were confused and partially disappointed as their expectations were not met at all.

However, in the second phase of the experiments, after being taught and explained the gesture alphabet, all subjects interacted with the system naturally. Most interestingly, when being explained that the channels were accessible by counting, all subjects succeeded in deducing the fact that if one channel is accessible by employing one finger, raising a certain amount of fingers might lead to interaction with other functions of the system. During this 'experimentation' phase of each user, highly rewarding moments were noticeable when a certain transfer effort led to a success in interaction. However, sometimes confusion arouse in this phase as e.g. some users tend to mimic the 'three gesture' with the index, middle and ring finger (and not as in

this database - partly inhibited due to their cultural background). This was quickly correctable and also accepted in the according cases.
What is moreover striking is the fact, that a certain gesture was rapidly learned by all the users being able to employ it whenever necessary or asked by the supervisor of the experiment. It is of extra importance to note that any of the gestures being initially transferred from the 2D tablet/phone scenario into the 3D scenario (e.g. touching to pointing) was completely discarded by the users as not feasible. So what was deemed as initially logic or intuitive - and still is for the development of other interfaces and related work mentioned in this paper - is not at all what the subjects were employing towards the end of the experiments. The presented system with its static hand poses was - towards the end - preferred by all the subjects, contrasted to the initial phase of each experiment, where it was observed that each of the subjects starting with waving or pointing in order to interact with the system. The main reasons for this, as mentioned by the participants, were the fact that a static gesture is more easily performed than a dynamic one.
Furthermore any introduction of a dynamic gesture would have demanded from the user to divert attention from the road towards the interaction with the system which is non-desirable in this situation. One may claim, any gesture demanding as little attention from the user to prove whether it was processed correctly is to be preferred over any other one. This is the case for this gesture alphabet as it was observable that all the subjects caught up the static gestures effortlessly and employed them for their use. Any corresponding gesture as 'pointing' (being the equivalent of a touch gesture on a display) to highlight e.g. a channel, demands the user to at least divert some attention from their primary task - something actually to be avoided in this given scenario.

7.1.5 Discussion

The user acceptance study conducted in the automotive environment shows that hand gestures can provide an intuitive means of interaction if realized appropriately. Static hand poses e.g. surpass dynamic hand gestures in terms of ease of use and lastly also user acceptance as they require less cognitive load to be performed. The fact that the UI of the tablet PC helps derive further hand poses by the simple alignment of elements shows how important it is to setup the interaction scenario as a whole even in such a minimal setting with few interaction features. The user acceptance scored weakly in the initial research phase but gained once the users fell comfortable and more specifically due to the fact that they could easily deduce certain poses without assistance. In a very short time the means of interaction felt natural and intuitive, as the evaluation of the questionnaire demonstrates.

Chapter 8

Discussion and Outlook

The work presented in this thesis demonstrates the development of a real-time applicable hand gesture recognition system for in-vehicle use with contributions and improvements made to several aspects:

- setup of a large reference database for 3D hand gesture recognition
- multiple ToF-based data fusion techniques
- development of various novel ML approaches applicable to different problems
- realization of a real-time applicable HGR system, quickly transferable to different application scenarios
- robust hand gesture recognition based only on depth data
- development of a robust dynamic hand gesture recognition framework based on static poses
- HMI-related research approaches to define the naturalness of human hand gestures

Additionally, work which has been conducted towards the ending phase on this thesis and hasn't found its way to be integrated, sheds some light on the following questions:

- development and testing of novel approaches to integrate depth data into convolutional neural networks
- integration of the whole HGR system onto mobile devices opening a whole new range of application scenarios for research and development

Some of the contributions which haven't made their way into this thesis are going to be presented on upcoming conferences but shall nevertheless be discussed here in a short form, as they are relevant to the topic of HGR overall. The following sections present the most important findings along with a critical discussion as well as an outlook on further research.

8.1 Discussion

This thesis outlines the possibility of setting up a hand gesture recognition system based on data retrieved from a single sensor. The contribution of this work is divisible into three fields: Machine Learning, Computer Vision and Human-Machine Interaction.

From the Machine Learning perspective several fusion algorithms were presented, fusing either data streams from multiple sensors or information obtained from a trained classifier. Fusing data streams either early, i.e. on the feature level, or late on the decision level was shown to perform equally well. To this end, various point cloud descriptors were tested and evaluated on a large, complex data base. It is not surprising that information, obtained from another angle improves the overall results as more knowledge about the problem is available. Moreover, various angle positions for fusion have been tested, showing that the exact angle position itself is not as relevant as the fact that there are multiple cameras viewing the scene. This provides more flexibility from a product designer's point of view. Adding another camera increases the overall cost of a multi-sensor setup thus it must be clear that the added benefit surpasses the expenses when considering an installation in this manner.

However, when a lot of data samples are available, the ML fusion approaches presented in this work demonstrate how classification accuracy can greatly be increased simply by training multiple models and exploiting information contained therein. Various approaches are addressed in this thesis, but the core idea of fusing features, obtained from a sample by nonlinear transformation, with neuron activities from a trained NN model remains the same in essence. An extensive number of experiments tested the viability of this concept extended to different domains, all proving its basic practicability. While the general performance of the system could be improved, also along with particular individual cases where previous models were struggling, the limits of this fusion technique also became clear very quickly. Not only does it require a lot of data samples but in a setting, in which an already well-performing model is present, significant improvements do not occur though it can be noted that it neither degrades the performance. Experiments conducted on the large database and on the well-known MNIST database validated this on multiple occurrences.

The main advantage of this concept remains the fact that *useful* information can be generated and utilized from the nature of the multiclass problem itself. Additional care must be taken when extending this idea beyond the hierarchy level utilized within this thesis as the model generated quickly tends to overfit. On the other hand, the effort undertaken for an optimal parameter search for generating optimal models, especially for the task of HGR, remains in a reasonable scope demonstrating the flexibility and adaptability to new problem domains. Performance improved the most in situations where the existing model performed worst, i.e. the gestures most difficult to disambiguate to to their resemblance, hence proving the practical applicability of the fusion techniques introduced in this work.

From a practical point of view, the underlying thesis shows that HGR can be done in real-time under difficult illumination settings with a low-cost sensor setup being quickly transferable to different domains due to its low space requirements. The HGR system provided high recognition results with frame rates up to 20 Hz on a Laptop with an Intel i7-3820 QM CPU running at 2.7GHz on one core with 8GB RAM.

HGR is a complex problem and hence the number of approaches differ significantly, each demonstrating satisfactory performance in its domain. This thesis shows that it is possible to achieve satisfactory results based on appearance only. Formalizing a complex hand model capturing the hand's 27 degrees of freedom naturally allows to capture more fine-grained movement. However, depending on the task aimed at, a data-driven approach is already more than sufficient as demonstrated in this thesis. Disambiguating between 10 static and 4 dynamic hand gestures can be done efficiently via an appearance-based system with either one or two depth sensors, allowing for creating a quick testing environment to research important HMI-related questions. Naturally, fusing information from multiple sensors boosts the performance, but aside of the additional space requirements the additional costs have to be taken into account. In an automotive setting, as presented in this thesis, the employment of ToF cameras maintains robust results but the question of positioning the camera is not easy to answer as space tends to be allocated very efficiently within an in-car setting from a designer's perspective.

Another benefit is the self-calibrating property of the system, due to the fact that only a depth data is used thus omitting the need for tedious calibration procedures with e.g. RGB cameras. The employed 3D data descriptor captures a large amount of variance in the data thus being optimal for the task of HGR with ToF cameras where noise is one of the greatest hurdles to deal with. As for the multi-sensor setup, the angle of alignment of the depth cameras is incorporated into the weights during training, further promoting the use of MLPs for the classification task, and consequently the sensors need only be positioned at an approximate angle towards each other.

There are many questions opening up when installing an HGR system into the vehicle in order to improve the user experience, such as where to place the sensor, how to define the interaction space of the system, what kind of gestures would be accepted by a large user base, which gesture should be assigned to which task and furthermore how to provide optimal user feedback. It was found out, that the setup, as demonstrated in this thesis, is sufficient with respect to being able to conduct proper user studies. The user-based study conducted within the frame of this thesis demonstrated that freehand gestures are generally accepted as a means of interaction. The gesture set chosen for this thesis is not too complex and the way of interacting with hands innately carries intuitiveness as on the one hand the users interacted seamlessly while being able to deduct unseen functions and interaction mechanics, simply from knowledge acquired during the task itself.

8.2 Outlook

Neural networks have shown to be beneficial for the core ideas of this thesis as they allow to easily research and evaluate the main algorithms introduced in Section 5.2. The algorithms, as mentioned, are intended to be useful in practical applications, which was demonstrated by the fact that a quick adaptation to certain problematic classes was possible within a reasonable amount of parameter search time and without the need to change the network's topology too much. Furthermore there indeed is information carried by the class output neurons which can be exploited opening various possibilities for further research.

One might ask, what the upper boundary of this approach is in terms of hierarchy levels. Initial results hint at possibly rapid overfitting problems but without extensive testing, both on various parameter settings and on difference problems, this remains difficult to assess. It seems, as proved by the experiments, that the information contained in the output layer of an MLP does not suffice to improve on the task in the same way as by the fusion approach, hence further research, directed at the question of what makes the fusion approach superior to the simple cascade approach, will be conducted. Moreover, fusing information from the output layer of an MLP seems to be extendable in an n-fold way for the standard approach as well as for the temporal domain, thus further work will be redirected to this topic.

This thesis has spawned a number of interesting research fields some of which have not been mentioned so far. On the more theoretical side, CNNs have show to perform extraordinarily, hence a number of tests has been run to assess their applicability to this domain. As only depth data was recorded for this task, the database was transformed and 11-bit encoded (cf. Figure 8.1) and fed into the Theano Python Library [4] [6] for training and classification.

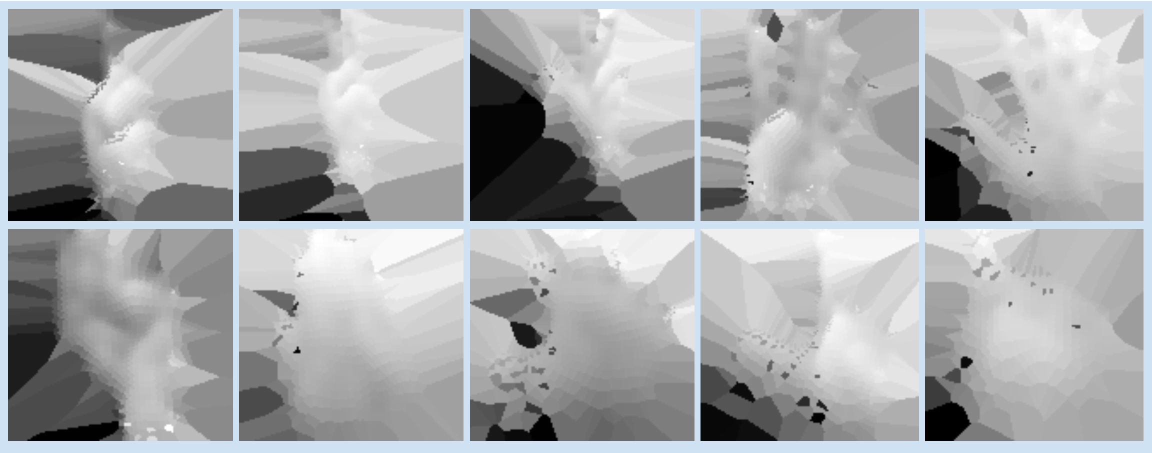


Figure 8.1: All ten gesture classes with the depth encoded.
Top row: Counting 1-5. Bottom row: ’Fist’, ’flat’, ’grip’, ’L’, ’point’.

Parameter optimization required some effort, however it was possible to achieve satisfactory results (cf. Table 8.1), approximating the findings from this thesis. It seems promising that with further effort expended in the direction of parameter optimization, equal or improved results can be achieved. The greatest drawback is the training time, as it can take several hours up to several days to train a single net on a PC with a CUDA-optimized graphics card, depending on the size of the training set and the parameter choice.

From the application point of view, this system has, with the exception of the ToF module due to interface problems, been transferred to a mobile device, achieving frame rates up to 20Hz, thus opening a whole range of new application scenarios. This is of uttermost interest as mobile devices equipped with ToF cameras are soon to appear on the mass market. All results are to appear on upcoming conferences related to this topic. Lastly, further research from the perspective of HMI is going to lead into two directions: How can we measure the effects of an HGR system towards the goal set prior, namely driver distraction and reduction of cognitive load and what

Learn. Rate	Feature Maps	Pre M.Pool	L1 kernel (mp 2x2)	L2 kernel (mp 2x2)	#pers.	#smpl.	val err	test err
.1	15, 30	4x4	9x9	5x5	5	300	17.0	23.0
.1	20, 50	4x4	9x9	5x5	20	300	36.0	38.0
.1	10, 20	4x4	9x9	5x5	20	300	9.46	14.81
.1	20, 20	4x4	9x9	5x5	20	300	9.86	15.8
.1	10, 20	2x2	5x5	9x9	20	300	61.0	60.0
.1	10, 20	2x2	9x9	5x5	20	300	64.0	62.0
.1	10, 20	2x2	17x17	9x9	20	300	69.0	68.0
.25	10, 20	2x2	9x9	5x5	20	300	7.45	11.55
.01	20, 20	4x4	9x9	5x5	20	300	45.0	44.0
.25	20, 20	4x4	9x9	5x5	20	300	11.5	16.7
.05	20, 20	4x4	9x9	5x5	20	300	10.0	16.0
.25	20, 20	2x2	9x9	5x5	20	300	7.33	11.46
.05	20, 20	2x2	9x9	5x5	20	300	7.43	10.98

Table 8.1: The best results along with the parameters for the net layers and kernels, with learning rate, feature maps, maxpool kernels, number of samples per persons and per train/test set, number of persons and the best error achieved on the test and validation sets.

in how far intuitiveness be a measurable factor for hand gesture interaction? A setup with a driving simulator and subjects performing the lane-change task [63] is going to be the scenario, in which these issues will be addressed.

Moreover, advanced concepts with respect to user guidance during interaction with hand gestures are being developed with the intention to further research the question of intuitiveness of hand gesture-controlled interafaces within the vehicle but also transferred to other interaction scenarios.

Bibliography

[1] Aitor Aldoma, Markus Vincze, Nico Blodow, David Gossow, Suat Gedikli, Radu Bogdan Rusu, and Gary Bradski. Cad-model recognition and 6dof pose estimation using 3d cues. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 585–592. IEEE, 2011.

[2] Luıs A Alexandre. 3d descriptors for object and category recognition: a comparative evaluation. In *Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Portugal*, volume 1. Citeseer, 2012.

[3] Frank Althoff, Rudi Lindl, Leonhard Walchshausl, and S Hoch. Robust multimodal hand-and head gesture recognition for controlling automotive infotainment systems. *VDI BERICHTE*, 1919:187, 2005.

[4] Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[5] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

[6] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.

[7] Alberto Del Bimbo and Pietro Pala. Content-based retrieval of 3d models. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 2(1):20–43, 2006.

[8] Kanad K Biswas and Saurav Kumar Basu. Gesture recognition using microsoft kinect®. In *Automation, Robotics and Applications (ICARA), 2011 5th International Conference on*, pages 100–103. IEEE, 2011.

[9] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library.* " O'Reilly Media, Inc.", 2008.

[10] Ludovic Brethes, Paulo Menezes, E Lerasle, and J Hayet. Face tracking and hand gesture recognition for human-robot interaction. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 2, pages 1901–1906. IEEE, 2004.

[11] Ilana Bromberg, Qian Qian, Jun Hou, Jinyu Li, Chengyuan Ma, Brett Matthews, Antonio Moreno-Daniel, Jeremy Morris, Sabato Marco Siniscalchi, Yu Tsao, et al. Detection-based asr in the automatic speech attribute transcription project. In *INTERSPEECH*, pages 1829–1832, 2007.

[12] Volkert Buchmann, Stephen Violich, Mark Billinghurst, and Andy Cockburn. Fingartips: gesture based direct manipulation in augmented reality. In *Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 212–221. ACM, 2004.

[13] Gilles Burel and Hugues Hénocq. Three-dimensional invariants and their application to object recognition. *Signal Processing*, 45(1):1–22, 1995.

[14] Manuel Caputo, Klaus Denker, Benjamin Dums, and Georg Umlauf. 3d hand gesture recognition based on sensor fusion of commodity hardware. In *mensch & Computer*, volume 2012, pages 293–302, 2012.

[15] Louis-Charles Caron, David Filliat, and Alexander Gepperth. Neural network fusion of color, depth and location for object instance recognition on a mobile robot. In *Second Workshop on Assistive Computer Vision and Robotics (ACVR), in conjunction with European Conference on Computer Vision*, 2014.

[16] Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

[17] Ankit Chaudhary, Jagdish Lal Raheja, Karen Das, and Sonia Raheja. Intelligent approaches to interact with machines using hand gesture recognition in natural way: a survey. *arXiv preprint arXiv:1303.2292*, 2013.

[18] Chia-Ping Chen, Yu-Ting Chen, Ping-Han Lee, Yu-Pao Tsai, and Shawmin Lei. Real-time hand tracking on depth images. In *Visual Communications and Image Processing (VCIP), 2011 IEEE*, pages 1–4. IEEE, 2011.

[19] Wongun Choi, Caroline Pantofaru, and Silvio Savarese. A general framework for tracking multiple people from a moving camera. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(7):1577–1591, 2013.

[20] Sheran Corera and Naomi Krishnarajah. Capturing hand gesture movement: a survey on tools, techniques and logical considerations. *Proceedings of chi sparks*, 2011.

[21] Koby Crammer and Yoram Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *The Journal of Machine Learning Research*, 2:265–292, 2002.

[22] Li Deng and Dong Yu. Deep learning: methods and applications. *Foundations and Trends in Signal Processing*, 7(3–4):197–387, 2014.

[23] Paul Doliotis, Alexandra Stefan, Christopher McMurrough, David Eckhard, and Vassilis Athitsos. Comparing gesture recognition accuracy using color and depth information. In *Proceedings of the 4th international conference on PErvasive technologies related to assistive environments*, page 20. ACM, 2011.

[24] André Elisseeff and Jason Weston. A kernel method for multi-labelled classification. In *Advances in neural information processing systems*, pages 681–687, 2001.

[25] Gregory M Fitch, Susan A Soccolich, Feng Guo, Julie McClafferty, Youjia Fang, Rebecca L Olson, Miguel A Perez, Richard J Hanowski, Jonathan M Hankey, and Thomas A Dingus. The impact of hand-held and hands-free cell phone use on driving performance and safety-critical event risk. Technical report, 2013.

[26] Sergi Foix, Guillem Alenya, and Carme Torras. Lock-in time-of-flight (tof) cameras: a survey. *Sensors Journal, IEEE*, 11(9):1917–1926, 2011.

[27] Giles M Foody and Ajay Mathur. A relative evaluation of multiclass image classification by support vector machines. *Geoscience and Remote Sensing, IEEE Transactions on*, 42(6):1335–1343, 2004.

[28] Barak Freedman, Alexander Shpunt, Meir Machline, and Yoel Arieli. Depth mapping using projected patterns, April 3 2012. US Patent 8,150,142.

[29] William T Freeman, Ken-ichi Tanaka, Jun Ohta, and Kazuo Kyuma. Computer vision for computer games. In *Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on*, pages 100–105. IEEE, 1996.

[30] J Friedman. Another approach to polychotomous classification. *Dept. Statist., Stanford Univ., Stanford, CA, USA, Tech. Rep*, 1996.

[31] Luigi Gallo, Alessio Pierluigi Placitelli, and Mario Ciampi. Controller-free exploration of medical image data: Experiencing the kinect. In *Computer-based medical systems (CBMS), 2011 24th international symposium on*, pages 1–6. IEEE, 2011.

[32] Timothy Gatzke, Cindy Grimm, Michael Garland, and Steve Zelinka. Curvature maps for local shape comparison. In *Shape Modeling and Applications, 2005 International Conference*, pages 244–253. IEEE, 2005.

[33] Jungong Han, Ling Shao, Dong Xu, and Jamie Shotton. Enhanced computer vision with microsoft kinect sensor: A review. *Cybernetics, IEEE Transactions on*, 43(5):1318–1334, 2013.

[34] S. Haykin. *Neural networks: a comprehensive foundation.* Prentice Hall, 1999.

[35] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[36] M Hossny, D Filippidis, W Abdelrahman, H Zhou, M Fielding, J Mullins, L Wei, D Creighton, V Puri, and S Nahavandi. Low cost multimodal facial recognition via kinect sensors. In *Proceedings of the land warfare conference (LWC): potent land force for a joint maritime strategy. Commonwealth of Australia*, pages 77–86, 2012.

[37] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2):415–425, 2002.

[38] Kota Irie, Naobiro Wakamura, and Kazunori Umeda. Construction of an intelligent room based on gesture recognition: operation of electric appliances with hand gestures. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 1, pages 193–198. IEEE, 2004.

[39] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(5):433–449, 1999.

[40] Ian Jolliffe. *Principal component analysis.* Wiley Online Library, 2005.

[41] Mohamed-Bécha Kaâniche. *Gesture recognition from video sequences.* PhD thesis, Université Nice Sophia Antipolis, 2009.

[42] Sanna Kallio, Juha Kela, and Jani Mäntyjärvi. Online gesture recognition system for mobile interaction. In *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, volume 3, pages 2070–2076. IEEE, 2003.

[43] Ben Kehoe, Akihiro Matsukawa, Sal Candido, James Kuffner, and Ken Goldberg. Cloud-based robot grasping with the google object recognition engine. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4263–4270. IEEE, 2013.

[44] Sheila G Klauer, Thomas A Dingus, Vicki L Neale, Jeremy D Sudweeks, and David J Ramsey. The impact of driver inattention on near-crash/crash risk: An analysis using the 100-car naturalistic driving study data. Technical report, 2006.

[45] Eva Kollorz, Jochen Penne, Joachim Hornegger, and Alexander Barke. Gesture recognition with a time-of-flight camera. *International Journal of Intelligent Systems Technologies and Applications*, 5(3-4):334–343, 2008.

[46] Thomas Kopinski, Stefan Geisler, Louis-Charles Caron, Alexander Gepperth, and Uwe Handmann. A real-time applicable 3d gesture recognition system for automobile hmi. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pages 2616–2622. IEEE, 2014.

[47] Thomas Kopinski, Alexander Gepperth, Stefan Geisler, and Uwe Handmann. Neural network based data fusion for hand pose recognition with multiple tof sensors. *ICANN*, 2014.

[48] Thomas Kopinski, Alexander Gepperth, and Uwe Handmann. A light-weight real-time applicable hand gesture recognition system for automotive applications. In *Intelligent Vehicles Symposium, 2015. Proceedings. IEEE*, page in press. IEEE, 2015.

[49] Thomas Kopinski, Alexander Gepperth, and Uwe Handmann. A simple technique for improving multi-class classification with neural networks. *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2015.

[50] Thomas Kopinski, Stephane Magand, Alexander Gepperth, and Uwe Handmann. A pragmatic approach to multi-class classification. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, page to appear. IEEE, 2015.

[51] Thomas Kopinski, Darius Malysiak, Alexander Gepperth, and Uwe Handmann. Time-of-flight based multi-sensor fusion strategies for hand gesture recognition. In *Computational Intelligence and Informatics (CINTI), 2014 IEEE 15th International Symposium on*, pages 243–248. IEEE, 2014.

[52] Alexey Kurakin, Zhengyou Zhang, and Zicheng Liu. A real time system for dynamic hand gesture recognition with a depth sensor. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pages 1975–1979. IEEE, 2012.

[53] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Intelligent Signal Processing*, pages 306–351. IEEE Press, 2001.

[54] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[55] Damien Lefloch, Rahul Nair, Frank Lenzen, Henrik Schäfer, Lee Streeter, Michael J Cree, Reinhard Koch, and Andreas Kolb. Technical foundation and calibration methods for time-of-flight cameras. In *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications*, pages 3–24. Springer, 2013.

[56] Tao Li, Chengliang Zhang, and Mitsunori Ogihara. A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. *Bioinformatics*, 20(15):2429–2437, 2004.

[57] Yi Li. Hand gesture recognition using kinect. In *Software Engineering and Service Science (ICSESS), 2012 IEEE 3rd International Conference on*, pages 196–199. IEEE, 2012.

[58] Zhi Li and Ray Jarvis. Real time hand gesture recognition using a range camera. In *Australasian Conference on Robotics and Automation*, pages 21–27, 2009.

[59] Rung-Huei Liang and Ming Ouhyoung. A real-time continuous gesture recognition system for sign language. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pages 558–567. IEEE, 1998.

[60] Jiayang Liu, Lin Zhong, Jehan Wickramasuriya, and Venu Vasudevan. uwave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive and Mobile Computing*, 5(6):657–675, 2009.

[61] Damian M Lyons. System and method for permitting three-dimensional navigation through a virtual reality environment using camera-based gesture inputs, January 30 2001. US Patent 6,181,343.

[62] Asanterabi Malima, Erol Özgür, and Müjdat Çetin. A fast algorithm for vision-based hand gesture recognition for robot control. In *Signal Processing and Communications Applications, 2006 IEEE 14th*, pages 1–4. IEEE, 2006.

[63] Stefan Mattes. The lane-change-task as a tool for driver distraction evaluation. *Quality of work and products in enterprises of the future*, pages 57–60, 2003.

[64] Sushmita Mitra and Tinku Acharya. Gesture recognition: A survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(3):311–324, 2007.

[65] Thomas B Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *Computer vision and image understanding*, 81(3):231–268, 2001.

[66] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, and Kari Pulli. Multi-sensor system for drivers hand-gesture recognition. In *IEEE Conference on Automatic Face and Gesture Recognition*, pages 1–8, 2015.

[67] Matteo Munaro, Filippo Basso, and Emanuele Menegatti. Tracking people within groups with rgb-d data. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 2101–2107. IEEE, 2012.

[68] Mathieu Nancel, Julie Wagner, Emmanuel Pietriga, Olivier Chapuis, and Wendy Mackay. Mid-air pan-and-zoom on wall-sized displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 177–186. ACM, 2011.

[69] James W Neuliep. *Intercultural communication: A contextual approach.* Sage Publications, 2014.

[70] Jakob Nielsen. *Usability engineering.* Elsevier, 1994.

[71] Michael Nielsen, Moritz Störring, Thomas B Moeslund, and Erik Granum. A procedure for developing intuitive and ergonomic gesture interfaces for hci. In *Gesture-Based Communication in Human-Computer Interaction*, pages 409–420. Springer, 2004.

[72] Steffen Nissen. Implementation of a fast artificial neural network library (fann). *Report, Department of Computer Science University of Copenhagen (DIKU)*, 31, 2003.

[73] Eshed Ohn-Bar and Mohan Trivedi. In-vehicle hand activity recognition using integration of regions. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pages 1034–1039. IEEE, 2013.

[74] Eshed Ohn-Bar and Mohan Manubhai Trivedi. The power is in your hands: 3d analysis of hand gestures in naturalistic video. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 912–917. IEEE, 2013.

[75] Eshed Ohn-Bar and Mohan Manubhai Trivedi. Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations. *Intelligent Transportation Systems, IEEE Transactions on*, 15(6):2368–2377, 2014.

[76] Kenji Oka, Yoichi Sato, and Hideki Koike. Real-time fingertip tracking and gesture recognition. *Computer Graphics and Applications, IEEE*, 22(6):64–71, 2002.

[77] Rebecca L Olson, Richard J Hanowski, Jeffrey S Hickman, and Joseph L Bocanegra. Driver distraction in commercial vehicle operations. Technical report, 2009.

[78] S. Oprisescu, C. Rasche, and B. Su. Automatic static hand gesture recognition using tof cameras. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pages 2748–2751. IEEE, 2012.

[79] Guobin Ou and Yi Lu Murphey. Multi-class pattern classification using neural networks. *Pattern Recognition*, 40(1):4–18, 2007.

[80] Francisco Parada-Loira, Elisardo González-Agulla, and Jose Luis Alba-Castro. Hand gestures to control infotainment equipment in cars. In *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*, pages 1–6. IEEE, 2014.

[81] Orasa Patsadu, Chakarida Nukoolkit, and Bunthit Watanapa. Human gesture recognition using kinect camera. In *Computer Science and Software Engineering (JCSSE), 2012 International Joint Conference on*, pages 28–32. IEEE, 2012.

[82] George V Paul, Glenn J Beach, Charles J Cohen, and Charles J Jacobus. Tracking and gesture recognition system particularly suited to vehicular control applications, May 23 2006. US Patent 7,050,606.

[83] Vladimir Pavlovic, Rajeev Sharma, Thomas S Huang, et al. Visual interpretation of hand gestures for human-computer interaction: A review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):677–695, 1997.

[84] Bastian Pfleging, Albrecht Schmidt, Tanja Döring, and Martin Knobel. Autonui: a workshop on automotive natural user interfaces. In *Proceedings of the 3rd International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, pages 207–209. ACM, 2011.

[85] Bastian Pfleging, Stefan Schneegass, and Albrecht Schmidt. Multimodal interaction in the car: combining speech and gestures on the steering wheel. In *Proceedings of the 4th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, pages 155–162. ACM, 2012.

[86] CA Pickering. The search for a safer driver interface: a review of gesture recognition human machine interface. *Computing and Control Engineering*, 16(1):34–40, 2005.

[87] Carl A Pickering, Keith J Burnham, and Michael J Richardson. A research study of hand gesture recognition technologies and applications for human vehicle interaction. In *3rd Conf. on Automotive Electronics*. Citeseer, 2007.

[88] John C Platt, Nello Cristianini, and John Shawe-Taylor. Large margin dags for multiclass classification. In *nips*, volume 12, pages 547–553, 1999.

[89] Jagdish L Raheja, Ankit Chaudhary, and Kunal Singal. Tracking of fingertips and centers of palm using kinect. In *Computational intelligence, modelling and simulation (CIMSiM), 2011 third international conference on*, pages 248–252. IEEE, 2011.

[90] Thomas A Ranney, Joanne L Harbluk, and Y Ian Noy. Effects of voice technology on test track driving performance: Implications for driver distraction. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 47(2):439–454, 2005.

[91] Jef Raskin. Viewpoint: Intuitive equals familiar. *Commun. ACM*, 37(9):17–18, 1994.

[92] Siddharth S Rautaray and Anupam Agrawal. Vision based hand gesture recognition for human computer interaction: a survey. *Artificial Intelligence Review*, 43(1):1–54, 2015.

[93] Leah M. Reeves, Jennifer Lai, James A. Larson, Sharon Oviatt, T. S. Balaji, Stéphanie Buisine, Penny Collings, Phil Cohen, Ben Kraal, Jean-Claude Martin, Michael McTear, TV Raman, Kay M. Stanney, Hui Su, and Qian Ying Wang. Guidelines for multimodal user interface design. *Commun. ACM*, 47(1):57–59, January 2004.

[94] Michael A Regan, Charlene Hallett, and Craig P Gordon. Driver distraction and driver inattention: Definition, relationship and taxonomy. *Accident Analysis & Prevention*, 43(5):1771–1781, 2011.

[95] Stefan Reifinger, Frank Wallhoff, Markus Ablassmeier, Tony Poitschke, and Gerhard Rigoll. Static and dynamic hand-gesture recognition for augmented reality applications. In *Human-Computer Interaction. HCI Intelligent Multimodal Interaction Environments*, pages 728–737. Springer, 2007.

[96] Zhou Ren, Jingjing Meng, and Junsong Yuan. Depth camera based hand gesture recognition and its applications in human-computer-interaction. In *Information, Communications and Signal Processing (ICICS) 2011 8th International Conference on*, pages 1–5. IEEE, 2011.

[97] Zhou Ren, Junsong Yuan, Jingjing Meng, and Zhengyou Zhang. Robust part-based hand gesture recognition using kinect sensor. *Multimedia, IEEE Transactions on*, 15(5):1110–1120, 2013.

[98] Zhou Ren, Junsong Yuan, and Zhengyou Zhang. Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 1093–1096. ACM, 2011.

[99] Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *The Journal of Machine Learning Research*, 5:101–141, 2004.

[100] Brian D. Ripley. *Pattern recognition and neural networks*. Cambridge university press, 1996.

[101] Linda Ann Roberts, Hong Thi Nguyen, and Edward Michael Silver. Gesture activated home appliance, August 30 2005. US Patent 6,937,742.

[102] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5:3, 1988.

[103] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 3212–3217. IEEE, 2009.

[104] Radu B. Rusu. The point cloud library, March 2011.

[105] Radu Bogdan Rusu, Nico Blodow, Zoltan Marton, Alina Soos, and Michael Beetz. Towards 3d object maps for autonomous household robots. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 3191–3198. IEEE, 2007.

[106] Radu Bogdan Rusu, Gary Bradski, Romain Thibaux, and John Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2155–2162. IEEE, 2010.

[107] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[108] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Dolha, and Michael Beetz. Towards 3d point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941, 2008.

[109] Yoichi Sato, Makiko Saito, and Hideki Koike. Real-time input of 3d pose and gestures of a user's hand and its applications for hci. In *Virtual Reality, 2001. Proceedings. IEEE*, pages 79–86. IEEE, 2001.

[110] Thomas Schlömer, Benjamin Poppinga, Niels Henze, and Susanne Boll. Gesture recognition with a wii controller. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*, pages 11–14. ACM, 2008.

[111] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond.* MIT press, 2002.

[112] Toby Sharp, Cem Keskin, Duncan Robertson, Jonathan Taylor, Jamie Shotton, David Kim, Christoph Rhemann, Ido Leichter, Alon Vinnikov, Yichen Wei, Daniel Freedman, Pushmeet Kohli, Eyal Krupka, Andrew Fitzgibbon, and Shahram Izadi. Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 3633–3642, New York, NY, USA, 2015. ACM.

[113] S. Soutschek, J. Penne, Jo. 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 Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*, pages 1–6. IEEE, 2008.

[114] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 1453–1460. IEEE, 2011.

[115] Thad Starner and Alex Pentland. Real-time american sign language recognition from video using hidden markov models. In *Motion-Based Recognition*, pages 227–243. Springer, 1997.

[116] Bastian Steder, Radu Bogdan Rusu, Kurt Konolige, and Wolfram Burgard. Point feature extraction on 3d range scans taking into account object boundaries. In *Robotics and automation (icra), 2011 ieee international conference on*, pages 2601–2608. IEEE, 2011.

[117] Jesus Suarez and Robin R Murphy. Hand gesture recognition with depth images: A review. In *RO-MAN, 2012 IEEE*, pages 411–417. IEEE, 2012.

[118] Jie Tang, Stephen Miller, Arjun Singh, and Pieter Abbeel. A textured object recognition pipeline for color and depth image data. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3467–3474. IEEE, 2012.

[119] Matthew Tang. Recognizing hand gestures with Microsoft's kinect. *Web Site: http://www.stanford.edu/ class/ee368/Project_11/ Reports/Tang_Hand_Gesture_Recognition. pdf*, 2011.

[120] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In *Computer Vision–ECCV 2010*, pages 356–369. Springer, 2010.

[121] Daniel Ullrich and Sarah Diefenbach. The intui questionnaire. http://intuitiveinteraction.net/, 2014.

[122] Michael Van den Bergh and Luc Van Gool. Combining rgb and tof cameras for real-time 3d hand gesture interaction. In *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, pages 66–72. IEEE, 2011.

[123] Juan Wachs, Helman Stern, Yael Edan, Michael Gillam, Craig Feied, Mark Smith, and Jon Handler. A real-time hand gesture interface for medical visualization applications. In *Applications of Soft Computing*, pages 153–162. Springer, 2006.

[124] Robert Walter, Gilles Bailly, and Jörg Müller. Strikeapose: revealing mid-air gestures on public displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 841–850. ACM, 2013.

[125] John Weissmann and Fblf Salomon. Gesture recognition for virtual reality applications using data gloves and neural networks. In *Neural Networks, 1999. IJCNN'99. International Joint Conference on*, volume 3, pages 2043–2046. IEEE, 1999.

[126] Paul Werbos. Beyond regression: New tools for prediction and analysis in the behavioral sciences. 1974.

[127] Daniel Wigdor and Dennis Wixon. *Brave NUI World*. Morgan Kaufmann, 2011.

[128] Frank R. Wilson. *The hand*. Vintage Books, 1999.

[129] Terry Windeatt and Reza Ghaderi. Coding and decoding strategies for multi-class learning problems. *Information Fusion*, 4(1):11–21, 2003.

[130] W. Wohlkinger and M. Vincze. Ensemble of shape functions for 3d object classification. In *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*, pages 2987–2992. IEEE, 2011.

[131] Ying Wu and Thomas S Huang. Vision-based gesture recognition: A review. In *Gesture-based communication in human-computer interaction*, pages 103–115. Springer, 1999.

[132] Yuan Yao and Yun Fu. Contour model-based hand-gesture recognition using the kinect sensor. *Circuits and Systems for Video Technology, IEEE Transactions on*, 24(11):1935–1944, 2014.

[133] Kristie Young, Michael Regan, and M Hammer. Driver distraction: A review of the literature. *Distracted driving*, pages 379–405, 2007.

[134] Xu Zhang, Xiang Chen, Wen-hui Wang, Ji-hai Yang, Vuokko Lantz, and Kongqiao Wang. Hand gesture recognition and virtual game control based on 3d accelerometer and emg sensors. In *Proceedings of the 14th international conference on Intelligent user interfaces*, pages 401–406. ACM, 2009.

[135] Zhengyou Zhang. Microsoft kinect sensor and its effect. *MultiMedia, IEEE*, 19(2):4–10, 2012.