Towards Gestural Understanding for Intelligent Robots

Dr.-Ing. Jan Nikolaus Fritsch

Dr.-Ing. Jan Nikolaus Fritsch
Honda Research Institute Europe GmbH
Carl-Legien-Str. 30
63073 Offenbach/Main

email: jannik.fritsch@honda-ri.de

Elektronische Version der von der Technischen Fakultät der Universität Bielefeld zur Feststellung der Lehrbefähigung im Fachgebiet Angewandte Informatik und zur Verleihung des akademischen Grades eines habilitierten Doktors der Ingenieurwissenschaften (Dr.-Ing. habil.) genehmigten schriftlichen Habilitationsleistung.

Vorsitzender des Habilitationsausschusses: Dekan Prof. Dr. Jens Stoye

Gutachter der schriftlichen Habilitationsleistung:

Prof. Dr.-Ing. Franz Kummert, Universität Bielefeld
Prof. Dr. rer. nat. Christian Wöhler, Technische Universität Dortmund
Prof. Dr.-Ing. Rainer Stiefelhagen, Karlsruhe Institute of Technology

Die schriftliche Habilitationsleistung wurde am 28.11.2011 bei der Technischen Fakultät der Universität Bielefeld eingereicht.

Tag der mündlichen Habilitationsleistung: 23.05.2012

Towards Gestural Understanding for Intelligent Robots

Habilitationsschrift

der Technischen Fakultät der Universität Bielefeld
vorgelegt von

Dr.-Ing. Jan Nikolaus Fritsch

Acknowledgments

This monograph is the result of a long process during which I have had the support from a variety of people, to whom I am deeply thankful. Especially the PhD students at the Applied Computer Science group at Bielefeld University with whom I have worked together have contributed substantially to the presented research results. I am deeply grateful to Nils Hofemann, Joachim Schmidt, Zhe Li, Axel Haasch, and Anna-Lisa Vollmer for the productive joint research that has made this monograph possible. Furthermore, I would also like to thank all other members of the Applied Computer Science group that I have worked with during my time in Bielefeld, as well as in later times during my visits to the university. I am especially thankful to Gerhard Sagerer for believing in me since he hired me as a PhD in his group in 1998 and to Franz Kummert who always found time to discuss issues over all these years up to the completion of this monograph that would not have been possible without his encouraging support.

I would like to sincerely thank the Faculty of Technology at Bielefeld University for accepting this monograph for the habilitation process and the two external reviewers Christian Wöhler and Rainer Stiefelhagen for reviewing this work despite their busy schedules.

I am also grateful for the support of the Honda Research Institute Europe, which has allowed me to give yearly lectures at Bielefeld University. I would like to specifically thank my colleague Christian Goerick who has created a very good working atmosphere despite the many challenges that we encountered together over the years.

As this thesis has been written completely in my spare time, I like to express my heartiest thanks to my wife Jomuna Choudhuri who has strongly supported me in this endeavor that has influenced many evenings and weekends over the last years. Finally, I would like to thank my mother Lisabeth van Iersel who continues to provide enthusiastic support as she has done during all my life.

Abstract

A strong driving force of scientific progress in the technical sciences is the quest for systems that assist humans in their daily life and make their life easier and more enjoyable. Nowadays smartphones are probably the most typical instances of such systems. Another class of systems that is getting increasing attention are intelligent robots. Instead of offering a smartphone touch screen to select actions, these systems are intended to offer a more natural human-machine interface to their users. Out of the large range of actions performed by humans, gestures performed with the hands play a very important role especially when humans interact with their direct surrounding like, e.g., pointing to an object or manipulating it. Consequently, a robot has to understand such gestures to offer an intuitive interface. Gestural understanding is, therefore, a key capability on the way to intelligent robots.

This book deals with **vision-based** approaches for gestural understanding. Over the past two decades, this has been an intensive field of research which has resulted in a variety of algorithms to analyze human hand motions. Following a categorization of different gesture types and a review of other sensing techniques, the design of vision systems that achieve hand gesture understanding for intelligent robots is analyzed. For each of the individual algorithmic steps – hand detection, hand tracking, and trajectory-based gesture recognition – a separate Chapter introduces common techniques and algorithms and provides example methods. The resulting recognition algorithms are considering gestures in isolation and are often not sufficient for interacting with a robot who can only understand such gestures when incorporating the context like, e.g., what object was pointed at or manipulated.

Going beyond a purely trajectory-based gesture recognition by incorporating context is an important prerequisite to achieve gesture understanding and is addressed explicitly in a separate Chapter of this book. Two types of context, user-provided context and situational context, are reviewed and existing approaches to incorporate context for gestural understanding are reviewed. Example approaches for both context types provide a deeper algorithmic insight into this field of research. An overview of recent robots capable of gesture recognition and understanding summarizes the currently realized human-robot interaction quality.

The approaches for gesture understanding covered in this book are manually designed while humans learn to recognize gestures automatically during growing up. Promising research targeted at analyzing developmental learning in children in order to mimic this capability in technical systems is highlighted in the last Chapter completing this book as this research direction may be highly influential for creating future gesture understanding systems.

Contents

1 Motivation

A strong driving force of scientific progress in technical sciences is the quest for systems that assist humans in their daily activities and make their life easier and more enjoyable. In robotics research, there is a long tradition in developing robots with human-like functionalities such as grasping, walking, or navigating in order to enable the robot to move around in the human's environment and manipulate it in a meaningful way. There has been tremendous progress in the field in the last 20 years leading to such impressive results as the humanoid robot ASIMO (Hirose et al., 2001), the small humanoid NAO (Aldebaran Robotics, 2011) and many other humanoid robots (see, e.g., Kawada Industries, 2011) that can move in a human-like way and manipulate simple objects.

Besides continuing the development of movement and manipulation capabilities of such robots, research in robotics focuses on the ambitious goal of building robots that exhibit human-like interaction capabilities in order to allow for a more natural communication between such robots and naive human beings. This effort is driven by the desire to design robots that can interact with humans outside the lab in real world scenarios like, e.g., private households. Developing such *robot companions* is a challenging research topic: the interaction interface has to match all requirements for an easy usability, so that even naive users are able to interact with the robot without an extensive training phase.

Obviously, the most natural way to interact with a robot would be to simply treat it like a human. Such robots that are intended to communicate with humans in a socially intuitive way are also called *social robots* (for an overview see Fong et al., 2003). There is a wide variety of aspects that need to be investigated to realize a socially interactive robot like, e.g., the robot's appearance (see, for instance, MacDorman and Ishiguro, 2006; Walters et al., 2008; Hegel et al., 2009), the role of emotions (see, e.g., Sloman and Croucher, 1981; Becker et al., 2005; Saldien et al., 2010), and the whole range of natural human-robot interaction (see, e.g., Arkin et al., 2003; Breazeal, 2004; Shibata et al., 2009). Especially the interaction capabilities determine what information can be transferred between the human and the robot. This has a direct consequence on what functional benefit the robot can provide, i.e., how good the robot 'understands' what the human wants in order to act appropriately, making it appear as an intelligent interaction partner.

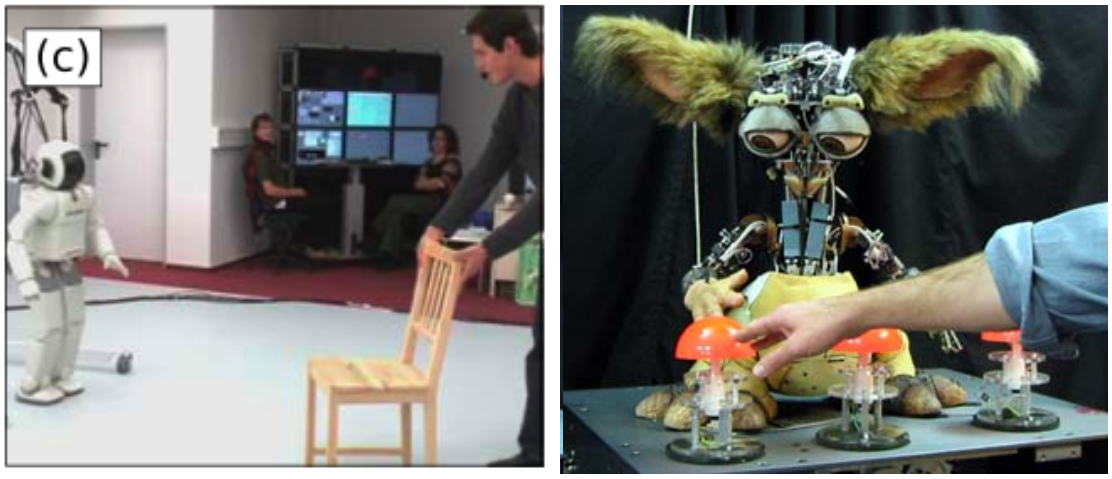
1.1 Background

In the beginnings of human-robot interaction in the 1990's, one of the first investigated communication channels that was going beyond a standard keyboard was the use of natural language. This allows a speech interaction where the human can provide information verbally to the robot (e.g., "My name is John.") or ask the robot for information (e.g., "What is your name?"). In principle, such a communication is not different from a phone conversation with an automatic call center system where only the unimodal speech is analyzed by a technical system for a purely verbal interaction with the human. However, in a setting where both interaction partners share the same environment, humans usually do make use of many other modalities. Consequently, as a robot is embodied and, therefore, shares the environment with the human, it should be able to take advantage of the modalities a human naturally employs in interaction with other humans in order to make the human-robot interaction more natural.

For example, enabling a limited form of a natural interaction with the robot ASIMO has been proposed by Schmüdderich et al. (2008) and is depicted in Fig. 1.1(a)). The human teaches objects to the robot by showing the objects to the robot and giving verbal information about object properties (e.g., "large"). Although the learning looks like a quite natural interaction, all of this is achieved in an object-centered way, i.e., the robot only learns object properties if the object is presented in its field of view. Therefore the human instructor has to move each object in front of the robot before providing additional verbal information.

The example in Fig. 1.1(a) would represent a more natural interaction if the human could also point to objects instead of bringing them into the field of view of the robot. In human-human interaction gestures are a highly important information source because humans intensively use their hands. Gestures can be used in a communication for many different aspects like, e.g., waving 'Hello', pointing to an object, or manipulating an object. Recognizing gestures that are independent of the environment and that are solely defined by the hand motion has been researched already in the 90s (for early reviews see Wu and Huang, 1999; Gavrila, 1999; Aggarwal and Cai, 1999). In contrast to such gestures recognizable in isolation, pointing and manipulating gestures interact with objects in the environment. For approaches incorporating such context information the term *gesture understanding* is used in this book to distinguish them from *gesture recognition* approaches based only on motion data.

A robot capable of understanding gestures allows a human to actually engage in an interaction with the robot in a more natural way without having to adhere to



(a) (b)

Figure 1.1: Teaching robots in interaction. a) The robot ASIMO learns the size of an object presented to him (image from Schmüdderich et al., 2008). b) The robot Leonardo learns the order of pressing buttons (image from Breazeal et al., 2004).

restrictions like a special manner of presenting objects as depicted in Fig. 1.1(a). One famous example where the gestural interaction is already looking more like an interaction with a social creature than a programmed robot is the robot Leonard (see Fig. 1.1(b)) developed by Breazeal et al. (2004). However, this robot is not mobile and the gesture recognition is realized using cameras mounted in the ceiling and behind the robot. Also, the objects that the human interacts with are simple buttons which are known to the robot beforehand.

In order to specify more clearly what is needed to achieve gesture understanding for a mobile robot, let us consider as example the ability to understand multi-modal pointing gestures which is highly relevant for achieving a natural human-robot interaction. Pointing to an object has already been intensively studied in experiments (see, e.g., Kranstedt et al., 2006; Pfeiffer et al., 2008) and virtual reality environments (see, e.g., Kranstedt and Wachsmuth, 2005). Virtual reality environments allow to investigate advanced computational models for understanding human gestures as the human gesturer interacts with objects in the virtual world and the perception of the gesture itself is usually simplified as the human has to wear special sensing equipment. Consequently, the complete context information is readily available from the virtual world for realizing gesture understanding. This allows to put the emphasis on researching a wide range of issues related to more advanced

interaction capabilities and cognitive models while the perception challenges encountered by a mobile robot are mostly avoided. Different from such more complex models for gesture understanding, in the setting of this book the term 'understanding' is restricted to integrating all perceptually observable information for an interpretation of the current gesture.

An important aspect of virtual agents as embodiments of the human counterpart is the opportunity to also investigate the generation of appropriate responses of the virtual agent like, e.g., the synchronous generation of speech and gesture (Kopp and Wachsmuth, 2004). Through creating a computational model that couples gesture recognition with the generation of speech and gesture it becomes even possible to realize more subtle processes like, e.g., imitating and mimicking of observed human hand motions by a virtual character (Sadeghipour and Kopp, 2011). This research stream will be of high relevance for making the interaction with future robots more natural. However, currently even the perception of complex hand motions and associated context information in real-world settings is still an unsolved research topic. In order to make current mobile robots more intelligent, the first step is the adequate processing of the environment information from the robot's sensors for going beyond gesture recognition and achieving gesture understanding while aspects related to generating the robot's response will not be covered in this book.

Coming back to the example of a pointing gesture, in an interaction with the robot the human can simply point to objects to teach them to the robot. In case the object is difficult to identify by gesture alone and the human did not specify additional object properties verbally, the robot can start a dialog and ask for more detailed information in order to resolve the object reference. In this way, the human can support the robot as he/she would do it with a human communication partner. All of the provided information can be stored and used in later interactions, improving the overall interaction quality.

Obviously, for a truly human-like interaction with a robot, the robot should be capable of understanding not only pointing gestures but all types of human gestures. This is not only important for understanding commands given by a human in a multi-modal way, but also for a pro-active behavior of the robot, or as Nehaniv (2005) puts it: "If the robot can recognize *what* humans are doing and *why* they are doing it, the robot may act appropriately". For example, a cooking support robot can monitor what a human is doing by tracking the manipulated objects that are equipped with radio frequency markers (Fukuda et al., 2005). If the robot understands the individual actions, i.e., how they relate to the overall target, it can give recommendations by speech or gesture to help in the kitchen as depicted in Fig. 1.2. Achieving such a gestural understanding without attaching markers to

Figure 1.2: The robot Robovie supporting a human in the kitchen by giving verbal and gestural advice on the presumed next action (image from Fukuda et al., 2005).

every possible object is an important step for the applicability of robots in different scenarios. A suitable alternative to markers is the use of non-intrusive computer vision techniques that can be used not only to recognize and track objects but also at the same time to observe the human hand motions.

1.2 Aim

Focussing on vision-based sensors that are carried onboard mobile robots, the aim of this book is to provide the reader with a deeper insight into the current research on gestural understanding for intelligent robots. The descriptions of methods and algorithms for vision-based gesture understanding are centered around the application domain of mobile robots and their specific requirements. Only when the recognition algorithms are developed with this specific application in mind, they are applicable for improving the robot's multi-modal interaction capabilities. Towards this end, typical vision-based methods for feature extraction and hand / body tracking are reviewed in the first half of this book. With this background, the second half of this book focuses in more detail on the broad variety of recognition approaches and especially the incorporation of context information into the gesture recognition process in order to understand gestures. Examples for relevant processing stages as well as for integrated systems provide an insight into the concrete realization of gesture understanding systems.

Having realized advanced gesture understanding capabilities, it becomes possible to build robots that have human-like interaction capabilities with respect to gesture

understanding within a given context. They are, therefore, getting closer to the goal of building truly social robots. However, a drawback of current recognition approaches is the fact that they are either manually programmed or require carefully prepared training data. In other words, the gesture understanding system has to be carefully designed by experienced researchers in order to enable the robot to understand gestures. This is in contrast to how young children learn to understand gestures, which is basically by observing other humans performing gestures in interaction. This learning of gesture understanding is, therefore, a fundamentally different process that is closely linked to the social interaction between the caregiver and the child. Research on developmental learning in children paves a highly interesting way for constructing open-ended gesture understanding systems and, consequently, may play a dominant role in this field in the future. At the end of this book, this topic is touched on by highlighting recent research results towards exploring aspects of developmental learning in children for constructing gesture understanding algorithms that enable human-like learning in social robots.

The remainder of this introductory Chapter is structured as follows: In the next Section the different skills a robot needs to have to support gesture understanding algorithms will be shortly reviewed. Based on these information sources, Section 1.4 will outline what processing functionalities have to be realized for gesture understanding and are detailed in later Chapters of this book. The terminology used throughout this work is summarized in Section 1.5 and the Chapter concludes in Section 1.6 with an overview of the organization of the subsequent Chapters.

1.3 Robot Skills Needed for Gestural Understanding

In order to build an intelligent social robot, several robot skills are necessary to support the algorithms performing gestural understanding. Throughout this book we will take as test domain the so-called *home tour* scenario that was introduced in the European project 'The Cognitive Robot Companion' - COGNIRON (2004). The core idea of this scenario is that a user buys a robot from a store and unpacks it at home for the first time. In this home scenario, the user has to show the robot all the relevant objects and places in his/her home that are needed for later interaction. For example, the user can go with the robot to the kitchen and say "This is my favorite blue coffee mug." while pointing to a mug. This is a relevant information if the user later in the living room asks the robot to "Get me my coffee mug.". The home tour scenario requires the robot to be capable of an interactive and dynamic learning process where it continuously extends its knowledge through

interaction with humans.

Performing gestural understanding in this scenario requires several other capabilities to support the gesture recognition algorithms and provide additional context information:

Finding a partner who wants to engage in a communication

The very first requirement for the robot is to know that there is an interaction partner. In static installations this is trivial, because the human will be required to position himself in front of the system. For mobile robots, the situation is a little different as the human may expect the robot to orient himself towards the human, similar as this is an implicit behavioral rule in human-human interaction. This in turn requires the robot to have omnidirectional sensors that allow him to detect a human anywhere in its surroundings. Humans usually use sound information for detecting other humans outside their current field of view, providing a coarse location of the interaction partner. After a robot has obtained such a coarse information, it also needs to be able to move towards this location and then find the interaction partner in order to align the specific sensors for gesture understanding towards this interaction partner.

Focussing the robot's sensors

Since both the user and the robot are mobile, the robot must be able to continuously track and focus on its communication partner to satisfy the precondition of an interaction. This is an important non-trivial skill as typical vision sensors have a limited field of view and in a natural interaction the human is usually not maintaining a fixed position and posture. Another aspect making this more challenging is the fact that often several potential interaction partners are in the vicinity of the robot. In such situations, the fusion of different information sources is needed for detecting and tracking the correct interaction partner (see, e.g., Fritsch et al., 2003). In the following, we will assume that at least the complete upper body of the interaction partner is always in the range of the robot's gesture recognition sensors and that the robot moves appropriately to avoid loosing the interaction partner if the human starts to move out of the sensor range.

Besides the ability to keep the gesturing human in the sensor range, the robot must also be able to sense the objects the human interacts with. If there is a direct physical manipulation of the object by the human's hand, the same sensor that is used for gesture recognition can be used for sensing the object. However, if the interaction is more distant, like in pointing gestures to another location or

(a) (reprinted from Reiser et al., 2009)

(b) (reprinted from Richarz et al., 2007)

(c)

Figure 1.3: Different situations where a robot interacts with a human partner.

a remote object, the field of view of the gesture recognition sensor might be not sufficient. For such situations, a steerable sensor might be needed that can be directed to the location the human is pointing at in order to provide good sensor data for the relevant object or location. In the following, we will assume that the robot possesses the skill for an adequate focussing of an object sensor to pointing locations or objects. Whether this requires the robot to actually have two distinct sensors for gesture recognition and object recognition depends on the field of view of the gesture sensor and the type of additional location/object information to be recognized. Figure 1.3 shows some exemplary situations where a robot and a human are engaged in an interaction.

Object recognition

The skill of object recognition is typically performed based on visual information obtained by the robot's object sensor and provides symbolic information about objects in the scene. While there is a huge variety of algorithmic realizations, a common aspect is that the object recognizer needs to be trained with example images of all possible objects of interest beforehand. Object recognition is needed if the type of the object plays a role for the gesture understanding process. Taking the example of a pointing gesture with an associated user utterance "This is my blue coffee mug", the object recognition should be able to recognize this object from a set of objects in the pointing direction in order to support the resolution of this multi-

modal object reference. Another example is a manipulative gesture, where the type of object serves as a-priori information about the possible manipulations that can be done with the object. These two examples show that object recognition can often provide context information which is complementary to the gesture-centered analysis of the visual information.

Speech recognition

As pointed out above, users should be able to use their natural interaction modalities when interacting with a social robot. Speech is a very powerful modality as it allows to specify object properties that can be used to restrict the search space and facilitate identification. Commercial all-purpose speech recognition systems use a huge vocabulary implicitly containing all words potentially used for specifying object properties. However, this requires the human to use close-talking microphones. Instead, if speech recognition is expected to work with microphones onboard the robot, the achievable sound quality is drastically reduced. This in turn requires also a drastic reduction of the vocabulary in order to achieve sufficient recognition quality while still including those words required for supporting gestural understanding. Independent of the chosen speech recognition approach, it should provide a textual transcription of the verbally specified user utterances for the next processing step.

Speech understanding & Dialog control

The task of speech understanding is the parsing of the user utterance into meaningful semantic parts. In other words, the textual transcription has to be converted into a semantic representation of the user input. This representation contains additional information like, e.g., expectations for potential gestures if the sentence starts with "This is ...". The individual semantic representations are processed by the dialog that has to collect all information provided by the user throughout the whole interaction, i.e., over several utterances and over several input modalities. For example, if speech understanding provides a representation containing the expectation for a gesture, the dialog can forward all relevant information (i.e., gesture expectation and additional verbally specified object details like 'blue' and 'coffee mug') to the subsystem for gesture understanding in order to resolve this object reference. From the perspective of the human user, the dialog is the main human-machine interface as it forwards user instructions to the internal robot control system and provides appropriate verbal feedback about their execution.

Memory

If the gesture understanding successfully resolves the object reference, it can extract object information from the visual scene (location, size, ...) that enhances the verbally specified information. The multi-modal information that is collected during such an interaction is not only relevant for the current interaction but needs to be stored for future retrieval. For example, a later instruction to "Get me my coffee mug." can be understood by the robot if it has access to some kind of memory to retrieve the last location of the mug. Therefore, such a memory needs to be accessible from different system components for using the stored information and updating it.

The different capabilities outlined above have to be present for realizing gestural understanding on a mobile robot. In the following, we assume all of these subsystems are performing as outlined and turn to the individual algorithmic steps required to perform gesture understanding itself.

1.4 Functionalities for Realizing Gesture Understanding

In order to approach the individual steps required to realize gesture understanding let us take the example of the home tour scenario where a human points out objects to a robot in a natural interaction style. This usually results in the human pointing to an object and giving - at the same time - verbal information like, e.g., "This is my favorite blue coffee mug". Resolving such multi-modal object references requires the robot skills outlined in the previous section as well as:

- detection of the hand / whole body
- tracking of the hand / whole body
- recognition of the gesture
- resolution of the multi-modal object reference by linking the pointing gesture to the verbally specified information and the visually perceived object data

The individual algorithmic steps that need to be implemented to resolve object references are depicted in Fig. 1.4. The conceptual architecture does not differentiate in the feature extraction stage whether the hand or the whole body is detected and tracked as long as information about the moving hand is obtained. While many

approaches to gesture recognition have focussed on detection and tracking of the hand only, an additional source of information for hand gesture recognition is the fact that the hand is linked to the human body. From the overall configuration of the human body, rough information about the hand position can be inferred. For some types of dynamic gestures that are independent of the exact hand shape, the movement of the lower arm may even be sufficient to recognize the gesture. We will therefore consider both cases in the more detailed description of the requirements in the following Subsections.

1.4.1 Detection of the Hand

Due to the embodiment of the robot and the goal of a natural interaction, the perception of the human and his gestures needs to be performed non-intrusively, i.e., without attaching measurement devices to the human gesturer. In this way, any human can directly engage in an interaction without any preconditions, like, e.g., wearing markers. Different sensing techniques are available for observing humans and their gestures, like, e.g., infrared cameras, time-of-flight cameras, or color cameras. As humans use their eyes to perceive the gestures of other humans, the application of computer vision techniques for gesture recognition by a robot seems most appropriate. An important additional benefit of using a color camera as sensor is that the manipulated environment can be analyzed in a way similar as the human does it. For example, this allows the social robot to 'understand' human descriptions of visual object properties like shapes ("round") or colors ("blue"). Obviously, objects can be equipped with technical means like radio frequency markers on the objects to identify and track them (Fukuda et al., 2005), but a social robot should

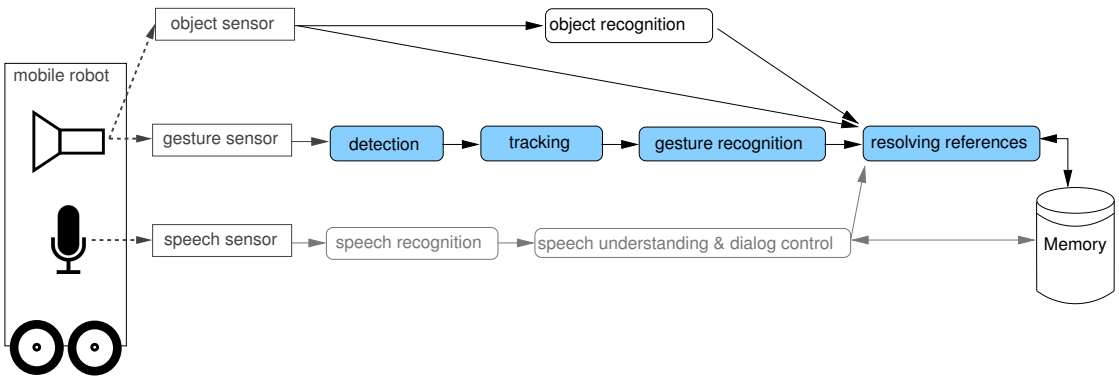


Figure 1.4: Conceptual system architecture for resolving multi-modal object references.

allow a human to introduce any object without preparing the object for interaction. Therefore, the focus is here on vision-based approaches for gestural understanding.

Research on vision-based gesture recognition has a long history and many different approaches have been developed. A variety of reviews (Moni and Ali, 2009; Mitra and Acharya, 2007; Poppe, 2007; Erol et al., 2007; Moeslund et al., 2006; Wang et al., 2003) have attempted to assess the current state of the field, but each review took a different perspective on the considered gesture types and sensing modalities. Most application examples of gesture recognition technology have concentrated on solving a specific recognition task that was defined for a very limited application domain like, e.g., recognizing sign language (Vogler and Metaxas, 2001) or Tai Chi gestures (Campbell et al., 1996). Many of these approaches are characterized by a static camera setup. Different from such typical experimental setups, a social robot moves in an environment, i.e., its camera setup is not static. This results in the image background being unpredictable, posing hard challenges to the analysis methods for detecting the gesturing hands/body parts:

Unknown background: One common detection technique is to learn the differences between the interesting foreground objects (i.e., the hands) and the background. In an interaction setting, the background in an ordinary living environment can contain anything - think about the different rooms you live in - which makes it impossible to provide a background model a priori. For every interaction situation, the robot would need to adapt itself to the current appearance of the background in order to separate the hands from this background.

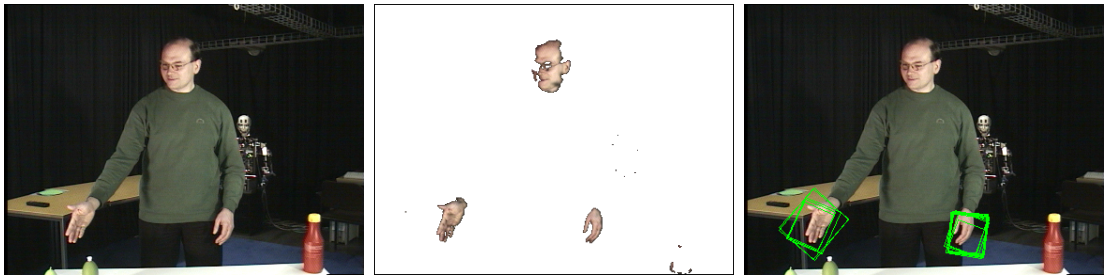
Dynamic changes in the background itself: While the adaptation to a static or slowly varying background can be performed (see, e.g., Wren et al., 1997), such an adaption becomes challenging in more dynamic backgrounds. While moving entities like other humans or pets may be explicitly detected and filtered out, this cannot be done easily with image areas containing unspecific motion like, e.g., a TV screen in the field of view.

'Moving' background due to robot motion: Another influence especially present with social robots are the dynamics of the robot itself. If during the interaction the human moves, the robot also has to move to keep the human in the field of view, resulting in a considerable change of the background.

From the many variations in the background that occur in a human-robot interaction setting it follows that any adaptation of the detection algorithms to the

specific background is a highly challenging task. Instead of trying to model the varying background, it is therefore advisable to put more effort in modeling the interesting foreground, i.e., the gesturing human.

The most basic feature for detecting hands is the skin color - even a single image pixel contains this information. An example result of applying a skin color detection algorithm on the image shown in Fig. 1.5(a) is depicted in Fig. 1.5(b). However, the skin color is not as discriminative as it may seem: skin color varies across the body (compare hand inside and hand outside!), lighting conditions influence the appearance of the skin, and different people have different skin colors. Moreover, an image of a gesturing human may contain also his face, being also skin-colored and, therefore, providing an additional false hand detection. All of these issues have to be handled when aiming to detect the hands, therefore color alone is often not appropriate.



(a) (b) (c)

Figure 1.5: Different approaches for detecting the gesturing hand: a) input image; b) detection of skin color; c) detection of hand appearance.

As a more complex image feature the overall appearance of the hand can be incorporated into the hand detection. The hand has many degrees of freedom and, consequently, the appearance varies substantially, but if only certain hand postures are to be detected, such an approach is a suitable way to capture the overall hand and not just individual pixels (see Fig. 1.5(c) for detection results). However, if the human hand takes on postures that are not covered by the appearance-based hand model, the hand detection will fail. In order to avoid relying only on the features of the hand itself for detecting it, the remaining human body can be a target of detection as well, imposing strong constraints on where to expect a hand.

While for the hand itself a certain appearance is expectable, this does not hold for the whole body. The appearance depends on clothing, and also the shape is influenced by the clothing. Therefore, a reliable detection of individual body parts

from a single viewpoint is quite challenging (see Sigal et al. (2004) for a multi-camera approach). A more promising direction for the ultimate goal of gesture recognition is to take a probabilistic approach to detection, i.e., provide in the detection stage only weak classifiers (Shotton et al., 2011) and leave the final decision on where the individual body parts are located in the image to the tracking stage, integrating this information over time. For example, such a weak classifier can find elongated structures (arms, legs, torso) that differ from the background. Whether this structure is actually a body part and which body part it is, is decided by the tracking algorithm combining all body part detections.

1.4.2 Tracking of the Hand

During an interaction with a social robot, a human may perform a variety of gestures. Some of these gestures (the static ones) may be completely described by a characteristic hand appearance so that the detection of the hand as outlined in the previous Section would be enough (e.g., *thumb up* for 'everything is ok') . Other gestures require to capture the overall hand movement (e.g., *pouring milk into the coffee*) and require the ability to track the hand positions over time. The standard way of tracking a single hand is to apply a Kalman filter to the results of the detection stage in order to integrate the hand positions over time and cope with occasional outliers (see, e.g., Fritsch et al., 2000). Tracking becomes more complicated, however, if both hands of the human act in the scene and if the hands can become occluded by each other or by objects in the scene. All of these aspects require a more complex tracking system, handling the appearance and disappearance of the individual hands appropriately. Obviously, limiting the detection to hands only and tracking them in isolation is more difficult than considering the whole body, allowing to implicitly track occluded hands by tracking the arms.

For tracking the whole body some detection of body parts is needed. Assuming that a reliable detection of non-characteristic body parts is hardly possible (see previous Section), the body tracking has to cope with unreliable detection results and should, therefore, be realized in a probabilistic way. The probabilistic modeling should not be restricted to the detection results, also the tracking itself can benefit by preferring 'typical' body configurations, i.e., assigning an increased likelihood to such configurations. Besides such soft influences, the modeling of the whole body also allows to incorporate hard constraints between body parts (e.g., physical joint constraints).

Combining all of these issues in a probabilistic body tracking framework provides the body configuration over time. This is a much richer representation than only

the hand trajectories. Nevertheless, if only the hand trajectories are needed for gesture recognition, they can be derived from the body configuration.

1.4.3 Recognition of the Gesture

The detection and tracking provides the hand's trajectory, which needs to be analyzed to recognize a certain gesture. The algorithmic approach for the recognition of a gesture depends on the type of gesture to be recognized. Staying with the example of a multi-modal object reference, a pointing gesture has to be recognized. Theoretically, also a static posture of a human pointing with the hand could be considered a pointing gesture. However, as this requires a different kind of analysis and the recognition algorithms would be strongly limited - there are only a few relevant postures in natural human-robot interaction - we focus here on the more general task of recognizing dynamic gestures.

If the tracking provides the trajectory data of the hand, then the task of the recognition algorithm is to analyze the trajectory data and detect a typical pointing trajectory. This can be achieved by applying a pattern matching algorithm to the trajectory data, assuming 'pointing' has a characteristic trajectory. While in the 2D image plane typically used for hand detection the trajectory is not very characteristic, the 3D hand trajectory is already more discriminative. If the 3D hand position is obtained by an approach tracking the whole body, then the information about the body configuration can be used as additional information. For example, the 3D trajectory data can be represented in a coordinate system relative to the human body. In this way, the fact that a human usually points to an object 'away' from his body is retained in the 3D data used for gesture recognition. There are several factors influencing how a referenced object can be identified based on a pointing gesture (Pfeiffer et al., 2008). Taking a simple heuristic, some implementations have assumed that the human looks at the object while pointing at it, resulting in a virtual eye-hand-object line (Nickel and Stiefelhagen, 2007; Droeschel et al., 2011) as depicted in Fig. 1.6.

1.4.4 Incorporating Context for Understanding the Gesture

Following the recognition of a pointing gesture, the object that was referenced with this gesture has to be identified by the social robot to allow for a dialog with the user about this object. For large objects or locations, the pointing gesture alone is sufficient to identify the object. However, if in a home setting a household object (e.g., a mug) is referenced on a table with several other objects, the pointing gesture

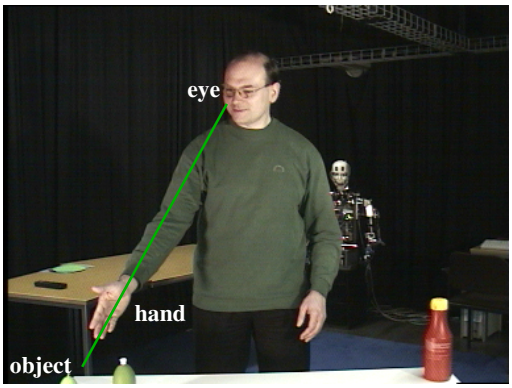


Figure 1.6: Pointing gesture with visualization of virtual eye-hand-object line.

alone will often be too imprecise to identify the object. Therefore, humans often give in such situations additional verbal information about the referenced object like its type, color, size, or shape.

For a natural interaction with a social robot, the algorithm for resolution of object references needs to be capable of taking advantage of such multi-modal context information provided by the human communication partner. Note that the term 'multi-modal' is used here not only for the speech modality but also to denote information that is extracted from the image data but is of a different type than the actual hand motion.

If the dialog receives the semantic representation of a user utterance from speech understanding containing information that a gesture is to be expected (see Section 1.3), it hands the relevant information to the object resolution. For the example "This is my favorite blue coffee mug" this would be

```
<Gesture expected = "yes">
<Object type = "coffee mug">
<Object color = "blue">
```

On receiving this information, the object resolution system queries the gesture recognition for a pointing gesture. If a pointing gesture has occurred, the rough pointing direction is provided by gesture recognition. This direction is in the 3D representation relative to the gesturing human and has to be transformed into world coordinates so that the robot can analyze the corresponding scene location. The verbally specified information is then used as an additional filter for analyzing the scene. In the example, the object color 'blue' can be used as low-level visual filter. Furthermore, if the robot already has an understanding of the object type 'coffee mug' and an appropriate object detector, this could be used for filtering as well.

However, as the appearance of objects varies strongly - think about the different mugs you have in your cupboard - this verbal information should be used only if the associated filtering will not prohibit finding the referenced object.

1.4.5 Beyond Communication: Recognizing Manipulative Actions

The resolution of object references as outlined above is needed for a successful communicative interaction between the human and the robot where the human intends to show the robot a new object. A different aspect in gestural understanding is the recognition of gestures that are intended to manipulate objects in the world instead of only pointing at them. For a variety of object manipulations the hand trajectory may look similar, but depending on the intention of the human, the object differs and there is a different effect. For example, pouring water into a plant or milk into your coffee mug may have a similar trajectory, but the interpretation is different. It is therefore necessary to associate the hand motion with the object manipulated in order to understand such manipulative gestures. Following the gesture categorization introduced by Bobick and Ivanov (1998) we also use the term *action* for this combination of gesture and object.

Recognizing manipulative actions therefore requires information of the hand trajectory and the object being manipulated. The necessary extension of the conceptual system architecture introduced before for realizing manipulative action recognition is depicted in Fig. 1.7. A component for object recognition is added and action recognition is achieved by combining trajectory and object data in a single integrated algorithm.

This incorporation of object information into the action recognition process has

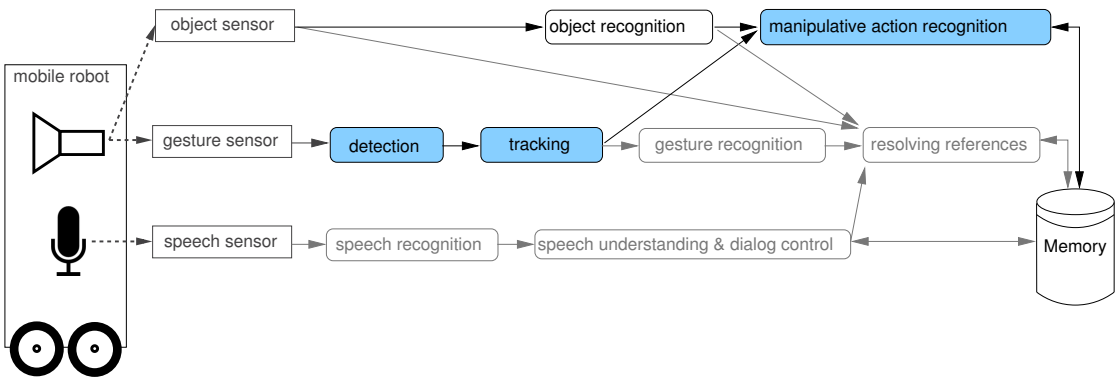


Figure 1.7: Conceptual system architecture for recognizing manipulative actions.

the additional benefit that also action sequences can be modeled in this approach. For example, if a human pouring coffee into a mug is observed, there is a high probability that he/she may add next sugar or milk to his/her coffee. In this action sequence, every next action is closely linked to the object currently manipulated, and the recognition process should be capable of taking advantage of this context information in order to recognize the individual actions and the overall action sequence.

1.5 Terminology

Before outlining the organization of this book, the central terms used for describing the different aspects of approaches for gesture understanding are reviewed in the following.

Posture describes a position or pose of the human hand or body. This is therefore some kind of static configuration information that can be identified based on a single image depicting a human without any temporal information. Here, we will focus on postures closely related to the hand itself or in conjunction with the arm and upper body. For example, when pointing at something this is indicated by the index finger of the hand, but without more information like, e.g., the overall body configuration it may be impossible to identify what specific object the human pointed at.

Gesture describes a dynamic hand movement, differentiating a dynamic hand gesture from a static hand posture. Obviously, techniques used for recognizing a hand posture from a single image can also be applied iteratively on a sequence of images. The information that a human holds his hand for 2 seconds in a pointing posture can be considered sufficient to say that he/she performs a pointing gesture. However, just the temporal extent of a posture is usually not informative enough to allow further inferences as much information is contained in the dynamics of the gesturing hand. Continuing with the pointing example, even when knowing the hand is in 'pointing' posture since two seconds, only the analysis of the dynamics of the hand motion may tell us the exact object the human is pointing at. As the majority of gestures to be expected in a natural human-robot interaction are characterized by the dynamic hand motion, the focus of this book is on the recognition of such gestures.

Detection denotes the successful verification that a certain object is present. For the task of gesture understanding, this means on the most basic level to verify that a hand is present. This verification can be done easily for elementary features like, e.g., the hand color. However, for the detection of the hand using more abstract representations like, e.g., the hand shape, this already involves specifying in more detail the hand to be detected. One can construct a detector for a 'pointing' hand posture, but this can be seen at the same time as actually recognizing a 'pointing' posture. There is, therefore, a smooth transition between detection and posture recognition.

Recognition denotes the identification of a certain object, i.e., not only its presence but also more details like, e.g., its current state or a specific subtype of an object class. As outlined for *Detection*, the task of detecting a specific hand posture can be considered a recognition of the posture. The difference becomes clearer for gestures: Detection of a gesture tells us there is a gesture performed, but not what gesture it is. Recognizing the gesture means also knowing what specific gesture was performed.

Understanding goes beyond *Recognition* by incorporating additional information. We use the term here to denote that recognizing only the hand motion does often not provide enough information for a robot to react appropriately. For the example of a human pointing at an object, when the robot has recognized a pointing gesture, this does not mean the robot has understood what object the human is pointing at. For the actual understanding, additional context information has to be analyzed. Incorporating this additional information differentiates gesture understanding approaches from classical gesture recognition methods operating only on the hand motion data.

Context denotes in this book the additional information sources available for enriching gesture recognition so as to achieve gesture understanding. Types of context considered here are user-provided context (e.g., verbally specified object properties) and situational context (e.g., what objects are present in the scene). Note that temporal context (e.g., what gestures have been preceding the gesture in question) is usually already incorporated in the recognition algorithms.

The terminology described above will be useful for reviewing the many different approaches that have been proposed for equipping robots with gesture recognition and understanding capabilities. It should be noted, however, that the proposed

terminology is just one way of differentiating between related approaches and that other distinctions could be made. This terminology, therefore, serves to guide the reader through this book, but other publications may use the terms introduced above in a different way.

1.6 Organization of the Book

The book is organized as follows: Chapter 2 starts by giving an introduction to general categorizations of gestures and to the more specific topic of gesture understanding for human-robot interaction. For the technical task of observing a gesturing human, the Chapter includes a comparison of different sensing devices to capture the human hand motion and motivates the use of standard vision cameras for gesture understanding to be performed by an embodied robot. The different design decisions that have to be considered when building a vision-based gesture understanding system are reviewed at the end of this Chapter.

The first step for any gesture understanding system is to detect the human hand. A variety of algorithmic solutions to vision-based hand detection is reviewed in Chapter 3. After a successful hand detection, the hand has to be tracked over several images to build up the temporal information of the hand gesture. Chapter 4 provides an overview of the different algorithmic approaches that have been developed to perform tracking of the hand. Two dominant approaches are the adaptation to changing visual features like, e.g., skin color and the model-based tracking of hands by incorporating constraints like, e.g., the overall body configuration. For each type there is a detailed example covered in this Chapter.

Chapter 5 describes probabilistic methods that are based on the hand trajectory and perform the recognition of gestures. While the methods outlined in this Chapter only deal with the isolated task of analyzing a hand trajectory, Chapter 6 introduces approaches that incorporate context into the recognition process to realize gesture understanding. Interesting context examples for gesture understanding are verbally specified object properties and objects that are manipulated by the gesturing human. For both types, algorithmic approaches are outlined in more detail.

Intelligent interactive robots that are equipped with some of the gesture recognition and gesture understanding techniques outlined in the previous Chapters are depicted in Chapter 7.

Going beyond manually designed gesture understanding approaches, Chapter 8 introduces research results obtained from studies where humans demonstrate ges-

tures to a child or a robot. Although research in this direction is just beginning, it offers the potential to replace engineered gesture understanding approaches by more advanced algorithms that incorporate learning techniques and take advantage of the way that humans employ for teaching gestures.

The book concludes with Chapter 9, reviewing the presented techniques and applications as well as possible future trends in making the gestural interaction with social robots more natural.

2 Gestures in Human-Robot Interaction

In this Chapter a more closer look will be taken at the type of hand gestures that are performed in human-robot interaction and the technical prerequisites for understanding these gestures. To start with, Section 2.1 first reviews a general categorization of gestures whose roots date back to the beginning of the last century. Back then, the target of the categorization was to describe the different kinds of gestural behavior a human being can make use of. For the aim of enabling robots to understand gestures, this section then reviews other categorizations targeted stronger at gestures relevant in human-robot interaction and finally identifies those groups of gestures most beneficial for creating an intelligent robot.

In order to approach this goal, a gesturing human has to be observed by the robot. Section 2.2 outlines different sensing techniques that have been applied for this task together with their respective strengths and weaknesses. Out of the different sensing techniques, a color camera is chosen as most appropriate sensing technique for recognizing the range of different gesture types performed in typical interactions of humans with social robots. Before going into the details of realized algorithms for this task, Section 2.3 provides the reader with an overview of the different design decisions that influence the performance and characteristics of a gesture understanding system. The Chapter is summarized in Section 2.4 before turning to the algorithmic details in subsequent Chapters.

2.1 Categorizations of Gestures

The term *gesture* is used in many different disciplines to describe aspects related to human movement. Although there exists the journal *Gesture* (Kendon and Mueller, 2005) that is explicitly focussed on gesture-related research, a clear definition of a 'gesture' applicable for different research communities is difficult. In the context of this book, the term gesture focusses on the interpretation of dynamic hand motions (cf. Section 1.5). A wider definition could incorporate movements of the whole body, facial mimics, or the dialog interaction (see, e.g., Rieser, 2004; Rieser and Poesio, 2009). Concentrating in this Chapter primarily on the analysis of hand movements, next a short look at general categorizations of hand movements will be

taken before focussing stronger on gestures in human-robot interaction.

2.1.1 General Categorizations

Due to the variety of different gestural expressions that are being performed by humans in everyday life, many categorizations can be found in the literature. While these differ in the targeted interaction situations and underlying research questions, there are similarities between many of the proposed categorizations. These common aspects are related to early gesture categorizations (Wundt, 1900/1973; Efron, 1941/1972). Focussing on the semiotic aspects of gesture we can outline four general types of gesture (McNeill, 1996, chapter 3):

Iconic and Mimetic gestures are used to depict an object or concept through a movement that resembles relevant aspects of it. For an object the gesture can indicate properties like, e.g., size or shape. Concepts are much broader in nature and, consequently, a variety of gestural expressions may indicate such things as directions of thought or metaphors. For example, drawing circles in the air with the stretched out index finger may indicate 'continue with your thoughts'.

Deictic gestures are closely related to the environment as they reference a specific object. Note that this object can be either concrete or only virtually present, i.e., introduced in the interaction preceding the gesture. For example, a stretched out index finger pointing at an empty location may indicate 'put it here'.

Symbolic gestures are simply gestures that have a specific and conventionalized meaning. Consequently, symbolic gestures are valid only in a specific social group and may have different meanings in different cultures. For example, connecting the thumb and index finger in a circle and holding the other fingers straight may indicate the word/meaning 'okay'.

Baton-like gestures and Beat gestures are not directly related to semiotic aspects as they do not convey meaning on their own. Instead, they occur in synchrony with other communicative expressions, e.g., with speech. Through beat gestures, specific parts of communicative expressions are highlighted to underline their importance or to help in the understanding process. For example, the importance of the verbally uttered sentence "Be here at four!" can be emphasized by knocking with the index finger on the table while saying "four".

The listed categorization is only a rough separation of human hand movements in different classes and many other categorizations have been proposed. For example, other early categorizations have been proposed by (Ekman and Friesen, 1969; Kendon, 1972). However, as also pointed out by (Kendon, 2004, Ch. 6), a single gesture category capturing all different aspects and viewpoints is difficult to find and it is more appropriate to see the range of different categorizations as tools for performing research in different directions. From the perspective of gestural understanding for intelligent robots, a general categorization does not incorporate the relevance of the considered gesture classes for interaction of a human with a technical system or the technical challenges when aiming to recognize relevant hand gestures.

2.1.2 Gestures in Human-Computer Interaction

The above categorization considered the complete range of gestures and the findings were based on observing humans interacting with each other. More recently, however, humans start to gesturally interact also with technical systems. In such settings, the computer can only react appropriately if the human gesture is recognized correctly. Consequently, for the task of gesture understanding in a technical system, the classification scheme should also consider how to tackle the recognition problem. One such attempt focussing more stronger on human-computer interaction was proposed by Pavlovic et al. (1997), who put forward the categorization hierarchy depicted in Fig. 2.1.

While the semiotic categorization introduced in the previous subsection focusses

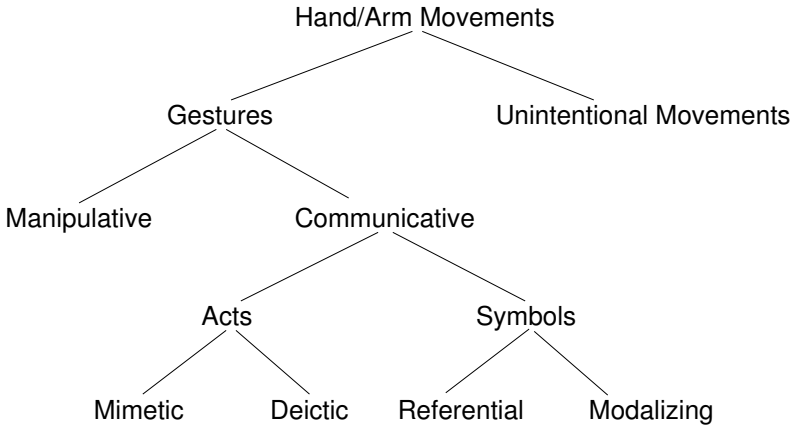


Figure 2.1: Gesture categorization for human-computer interaction proposed by Pavlovic et al. (1997).

mainly on aspects outlined in Pavlovic's subgroup of communicative gestures, Pavlovic's hierarchical view is already targeted stronger at the computational analysis of gestures. We find on the top-level a separation into unintentional movements and gestures. Unintentional movements do make humans appear 'human-like' and are very important in creating virtual agents that should exhibit a human-like behavior (Becker et al., 2004), but such human movements are not part of any meaningful gesture. However, a technical system for gesture recognition has to be able to separate such movements from meaningful gestures in order to avoid the confusion of unintentional movements with actual gestures, i.e., producing a large number of false positive results. Gestures themselves are separated into manipulative and communicative gestures, where the large subgroup of communicative gestures is split up into several categories highlighting the different kinds of gestures that can be used in an interaction. Here we can see similar terms as in the general categorization from the previous subsection. While Pavlovic's categorization is more suited for describing gestures occurring in interaction with a technical system, it is still quite general when aiming to equip a robot with gesture understanding capabilities.

2.1.3 A Gesture Categorization for Human-Robot Interaction

Another categorization of gestures that is especially targeted at the recognition of gestures in human-robot interaction contexts has been proposed by Nehaniv (2005) and is depicted in Fig. 2.2.

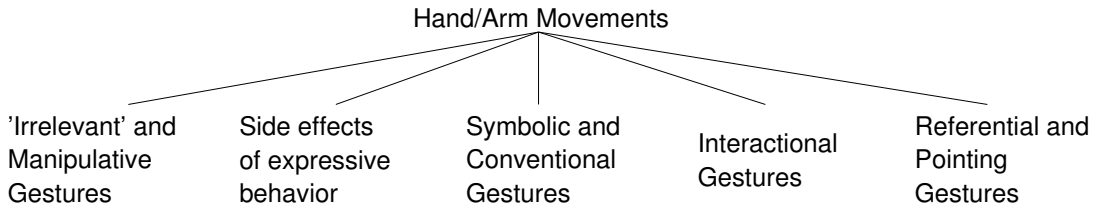


Figure 2.2: Gesture categorization targeted at human-robot interaction proposed by Nehaniv (2005).

This categorization is not hierarchically structured but provides a separation between different gestures that play different roles when it comes to the underlying intent of the gesturing human. Each class in Nehaniv's categorization subsumes several specialized classes capturing important differences within a parent class:

'Irrelevant' and Manipulative Gestures

The first class includes all body motions that are intended to change the human's relation to the environment or the environment itself. Importantly, these gestures are not intended to communicate or to interact with another partner. For example, the motion of arms and hands during walking are gestures that are simply *side effects of motor behavior*. Changing the environment is performed with *irrelevant gestures* like, e.g., playing with a paper clip or with *actions on objects*. The latter subtype includes the broad range of actions that are in the focus of research on imitation learning (Billard and Siegwart, 2004) and includes such actions as grasping a cup in order to drink.

The description of these subtypes clearly shows that there is a varying degree of context relevance in the individual gestures. For example, *side effects of motor behavior* are directly linked to the current activity of the overall body (i.e., the behavior) while there is no physical context when observing someone playing with a paper clip. It should be noted that for recognizing irrelevant and manipulative gestures acting on objects, these objects need to be known in order to understand the gesture and, therefore, the current intent of the acting human.

Side Effects of Expressive Behavior

This class is separated from the irrelevant gestures described before as it contains all gestures that are performed during an interaction with an interlocutor. Different from gestures with an explicit role in the communication process that are the content of the following classes, this class contains all those gestures without any specific interactive, communicative, symbolic, or referential role. It should be noted, however, that they may play a relevant role in improving the overall interaction quality as they may be used for emphasizing verbally uttered statements.

Symbolic and Conventional Gestures

As pointed out in Section 2.1.1, symbolic gestures depend on social groups in which such gestures have a conventionalized meaning. These gestures explicitly convey content through a specific hand configuration and are therefore comparatively simple to recognize using trained models. However, as these gestures are heavily dependent on knowing the context in which a gesture is performed, the correct interpretation of a recognized gesture may be challenging. For example, holding up two fingers can be interpreted in different ways (Morris et al., 1979):

1. A person that enters a restaurant and - after visually contacting the waiter -

performs this gesture may want to communicate that a table for 'two' persons is needed.

2. In a demonstration of anti-war protesters this gesture is likely to have the meaning 'peace'.

3. Politicians and others commonly use it as a sign for 'victory'.

This short listing is by no means complete but already gives an impression of the challenges that a gesture interpretation system faces after having identified the symbolic gesture itself.

Interactional Gestures

This class represents all gestures that are used to regulate the interaction with a partner. This regulation of a cooperative activity includes initiating, maintaining, synchronizing, organizing, or terminating the interaction. A dominant cooperative activity is communication itself, where interactional gestures do not convey any content, but are simply supporting the exchange of information by regulating the information flow. Interactional gestures are targeted at influencing other people's behavior as opposed to *'irrelevant'* and *manipulative gestures* (Nehaniv's first class listed above) that are acting on inanimate objects. Note, however, that offering an object to a partner by holding the object is a gesture that would belong to both classes.

Referential and Pointing Gestures

Compared to the other classes listed before, this last class is rather narrow. It contains those gestures that are used to refer to objects, locations, persons, or directions. Especially objects and locations are not restricted to physically present entities, but include locations in space that are used as 'virtual' objects or locations during a discourse. For example, when describing a room layout on a table area between two interlocutors, the position of the door may be a location on the table introduced at the beginning of the interaction. Later, both just point to this 'empty' position on the table when referring to the door of the room they are talking about.

2.1.4 Selected Categories for Creating Intelligent Robots

For a robot exhibiting truly human-like gestural interaction capabilities, all of Nehaniv's categories outlined in the previous Section would have to be recognized by

a robot. However, the current state of the art in gesture understanding is far from coping with this broad variety of gestures. As the different categories pose different requirements on the sensors and algorithms for recognition and understanding, research projects usually pick a single gesture category for developing algorithms.

For prioritizing what gestures to focus on for making robots more intelligent, it should be noted that enabling the robot to recognize *Side Effects of Expressive Behavior* will not raise the quality of the information flow between human and robot. A similar argument holds for *Interactional Gestures* as the interaction between human and robot is not very fluent anyways. However, it is very important to be aware that such types of gestures occur in human-robot interaction. Any gesture recognition algorithm aiming at the recognition of the other categories has to be robust against these types to avoid producing false recognitions.

In contrast to the previous two categories, *Symbolic and Conventional Gestures* explicitly support information exchange. However, the concepts that can be transmitted by the human with these gestures are rather abstract and can also be expressed verbally in an advanced human-robot interaction interface. Therefore, this category is also not a first priority for realizing gesture understanding capabilities on a robot.

Referential and Pointing Gestures are gestures performed to transfer information to a communication partner. As these gestures refer to objects or virtual places in the physical space, this kind of information provided by a gesture cannot be replaced easily by a verbal command. Therefore, this category will be one focus in the description of gesture understanding algorithms in this book.

Manipulative Gestures are another category that is highly relevant for making robots more intelligent. Different from the referential gestures, the manipulative gestures are not part of any explicit communication. However, they are very important for more advanced understanding capabilities of the robot, i.e., the robot can know what the human is doing without being told explicitly. Consequently, this is the second category that is in the focus of this book.

2.1.5 The Influence of Context on Gesture Understanding

The previous Subsection has focussed on *which* gestures to recognize. The main emphasis in this book will be on algorithmic methods on *how* to recognize these gestures. The two questions of *which* and *how* to recognize are central questions for any scientific paper on gesture recognition and are usually already answered in the abstract of any paper.

However, a third question that is often overlooked in scientific papers on gesture

recognition but is highly important for providing a robot with a good gesture understanding performance is *when* to recognize a gesture. Many papers assume that the human performs meaningful gestures while being observed by the robot which is the case in a lab environment where the focus of the interacting humans is on testing the robot's gesture understanding capabilities. However, this is not true in a natural interaction setting as the categorization in Section 2.1.3 has shown. Consequently, when focussing on a subset of the categories proposed in the previous Subsection, the fact that many 'unmodelled' gestures will be observed in the data has to be considered. If the gesture understanding algorithm has knowledge of *when* to expect a relevant gesture, it can use this context information to restrict the processing to such situations. This not only saves computational resources but also reduces the likelihood of wrong recognition results.

Coming back to the example of a pointing gesture as described in the Motivation, a possible context for activating the recognition of such a referential gesture is the information verbally given by the user. If the user's utterance contains the keyword 'this', it is likely that he/she is currently pointing at an object that he/she is describing. Another type of context that is more relevant for manipulative gestures is the situational context in the form of objects. Only if the hand is close to, e.g., a coffee mug, a manipulative gesture 'lifting mug' can occur while this gesture should not be recognized if there is no mug in the vicinity. Including such different kinds of context into the gesture recognition algorithms in order to achieve gesture understanding is a crucial step for making robots more intelligent. Consequently, different algorithmic approaches aiming at this target will be covered in detail in Chapter 6 of this book.

2.2 Sensing Devices for Observing Gesturing Humans

In order to understand gestures, a technical system has to acquire information about the gesturing human. Clearly, the most important features are the motion of the hand and its configuration. But besides these obvious aspects, the current configuration of the overall body and the objects in the environment can be of high relevance, too (see, e.g., Kjellström et al., 2008). In this Section we will review different sensing devices that can be used for measuring hand gestures and partly also the overall body configuration. We will point out their advantages and disadvantages, respectively, to motivate the choice of a camera as sensing device for gesture understanding.

2.2.1 Intrusive Sensing Methods

Probably the first method to provide a technical system with data from a gesturing human was to attach sensors directly to the body. Depending on the application domain, several different intrusive sensing techniques are in use today:

Marker-based Sensing of the Overall Body Motion

A standard technique used, e.g., for creating characters in animation movies or computer games is to attach markers to the human body. These markers can be active in the sense that each marker can calculate its position in some world coordinate system through, e.g., measuring a magnetic field generated externally. As this requires the markers to dispose of local processing and energy, the resulting devices can become rather large.

An alternative are passive markers that are sensed by external sensing devices. The most common technique is to use reflective markers that can be detected very reliably in images from far-infrared cameras (see Fig. 2.3).

Figure 2.3: The highlighted dots are infrared reflections from markers attached to the human body and golf club.

The benefit of using the infrared spectrum instead of the visible spectrum is the robustness against other scene contents. In the visible spectrum the lighting conditions heavily influence the observed scene and are usually determined by the application domain. In contrast, the far-infrared cameras measure reflections of infrared light sources that are used to illuminate the scene without disturbing the human impression. This is especially important in virtual reality environments

like, e.g., caves where the visible light is dimmed to allow for better visibility of the screens representing the virtual environment.

An important disadvantage of markers is the necessity to attach them somehow to the body. While this can be done for the limbs by attaching markers to the clothes, the equipping of hands with markers is less straightforward. By wearing some kind of wristlet, the rough hand position can be obtained, but more detailed information like, e.g., the finger positions is beyond the reach of this sensing method as it will disturb the human while gesturing.

There have also been attempts to use far-infrared lighting directly without any markers to cope with low-light situations like, e.g., military command rooms (Chu and Nevatia, 2008). Due to the lack of additional markers, only a coarse visual image is obtained and a sophisticated post-processing similar to 2D vision approaches (see later) is needed to extract the gesturing human with this technique.

Instrumented Clothing: Motion Capture Suits

Another way of measuring human body motions is known as motion capturing and is usually realized by letting a human wear a so-called motion capture suit as depicted in Fig. 2.4(a).

A motion capture suit is essentially a tight fitting garment that has various

(a) (b)

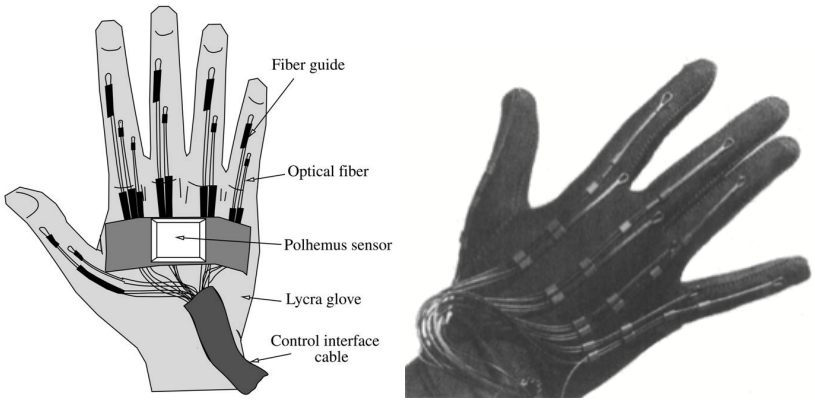
Figure 2.4: Xsens® motion capture suits with position sensors to directly measure the human body configuration. a) classical motion capture suit; b) advanced version to wear over normal clothing (images from http://www.xsens.com).

sensors to directly measure the position of the human limbs relative to each other without requiring any external setup. While the marker-based approach provides the relative information only implicitly, a motion capture suit has sensors to extract this information directly at the spot, providing a much higher quality. However, as a classical motion capture suit is somewhat difficult to put on, it is generally not used for interactive applications but rather in domains like, e.g., creating animation movies. More recent versions as depicted in Fig. 2.4(b) aim to enable the use of this technique also for other applications.

If a user engaged in an interactive application wears a motion capture suit, the suit could also be equipped with devices to enable a direct sensory feedback to the user's body. For example, lifting an object in a virtual reality environment could be made more realistic by force feedback. Up to now this possibility has only been explored in few applications (see, e.g., Miaw, 2010), but it may become more important in the future.

Instrumented Clothing: Glove-based Sensing of the Hand

While motion capture suits are primarily targeted at measuring the overall human body configuration, so-called datagloves are instrumented gloves for measuring the detailed configuration of the hand. Figure 2.5 depicts one such device where the sensor wiring is easy to see. Such gloves are used intensively in virtual reality environments where the hand configuration itself, i.e., the position of all fingers, has to be measured.



(a) (b)

Figure 2.5: A dataglove measures the hand configuration directly. a) technical sketch of the measurement instrumentation; b) picture of the original VPL DataGlove from 1987.

As it is challenging to use attached markers for extracting this information due to visual occlusion of the individual fingers, wired gloves are a quasi standard in this application domain. For similar reasons, these gloves are used in research on robot grasping (see, e.g., Pardowitz et al., 2007; Maycock et al., 2011), but they are not well-suited for natural gesturing in everyday life.

2.2.2 Active Sensors

The intrusive techniques outlined before directly provide measurements of the position of human limbs and/or bending of their joints. In the case of special markers for infrared light, this is already a combination of intrusive sensing with an active sensing approach where infrared light is used to illuminate the scene. Alternatively, one can use an active sensor that does not require any physical markers on the human body, i.e., there is no need for any preparation by the user. Two measuring techniques are in common use to obtain a 3D image of a gesturing human wearing no additional equipment: time-of-flight measurements and structured light. It has to be noted that both sensor types provide only a depth image and possibly a grey level image. Attempting to recognize a human hand from such data is basically impossible, as its shape is not discriminative enough. Instead, such active sensors are applied to obtain the configuration of the complete body pose and derive the hand position from this. Additionally, if color information is needed to analyze the scene, a separate color camera (see next section) has to be used in a calibrated setup with the depth sensor to correctly associate depth and color pixels to each other. This is needed if the human interacts with its physical environment, but not if he/she interacts with a virtual environment or a computer game on a screen.

Time-of-Flight Measurement

One type of active sensors uses modulated light and measures the time of flight of the emitted light to calculate the distance to the surface reflecting the light. Figure 2.6 shows an example of a depth image obtained by a time of flight sensor.

Some active sensors able to capture scenes in 3D have become commercially available, but they are rather expensive and usually have only low image resolutions (MESA Imaging, 2011; PMDTec, 2011). For example, the depth image shown in Fig. 2.6 has only 160×124 pixels. While we as humans can easily infer the scene content from looking at the depth image, this data has to be processed by special algorithms to infer the hand configuration. Especially in the case of today's time-of-flight sensors, the low resolution makes this task very hard.

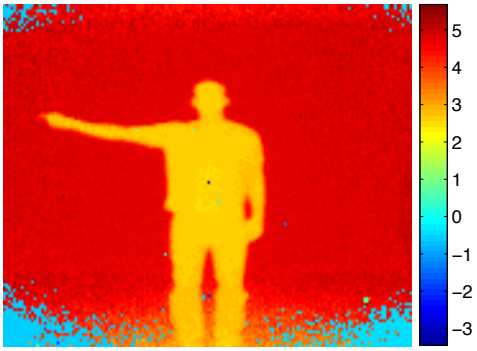


Figure 2.6: Depth image obtained by the active laser-based SwissRanger® SR2 time of flight sensor with the depth values represented by pixel color (image from Holte et al., 2008).

Structured Infrared Light - the Kinect Sensor

Instead of explicitly measuring the depth based on time of flight, it is also possible to use structured light for illuminating a scene and calculating the depth values from the deformation of the projected pattern. This technology has appeared in a famous consumer product in November 2010, the Kinect® system (Microsoft, 2011) for the Microsoft® XBox® game console. With its market introduction, the remarkable quality of the depth measurements of this novel sensing device has inspired many researchers to also use this device for other applications than just interacting with computer games. The sensing equipment of the Kinect® system intended for gesture recognition consists of three parts:

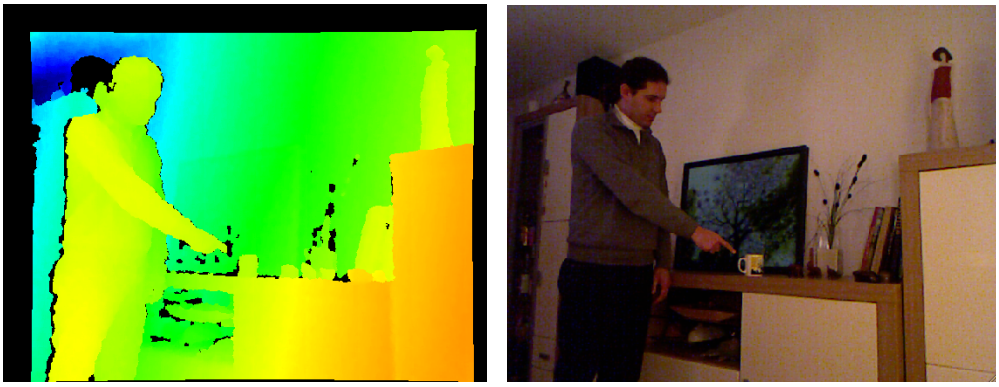
- An infrared light source to project a specific pattern in the scene.
- A 640×480 infrared CMOS camera to record the infrared pattern.
- A 640×480 RGB CMOS camera to record the visible scene (cf. next Section).

Figure 2.7 depicts an image from the infrared camera visualizing the structured pattern projected by the active infrared light. The pattern is differently deformed based on the shape of the human body it is projected on. With this it is possible to calculate from the deformations of the infrared pattern image the depth of each infrared point, providing a depth image for the complete scene as depicted in Fig. 2.8(left).

In addition to this depth image, the Kinect® also contains a standard camera to obtain an RGB image of the scene. The two outputs of the sensor that serve as basis for gesture recognition are depicted in Fig. 2.8. By knowing the geometrical

Figure 2.7: Image from Kinect sensor exhibiting pattern of structured infrared light.

relation between the depth image and the RGB image, both images can be mapped to each other and for each color pixel value also its depth value becomes available. This results in a so-called RGBD image.



(a) (b)

Figure 2.8: The data provided by the Kinect sensor after preprocessing are (a) the depth image and (b) the associated RGB color image.

The additional depth information is an import cue used in the Kinect® application software for generating for each image several body pose hypotheses (Shotton et al., 2011) that are then pruned in a tracking framework based on temporal information. This can be assumed to be done with the help of color information, but no publication detailing this is available, yet. Nevertheless, the methods used by the Kinect application software for recognizing body poses and gestures are similar

in nature to the gesture recognition methods based only on vision-based sensing, which is covered in the next Subsection.

2.2.3 Vision-based Sensing Methods

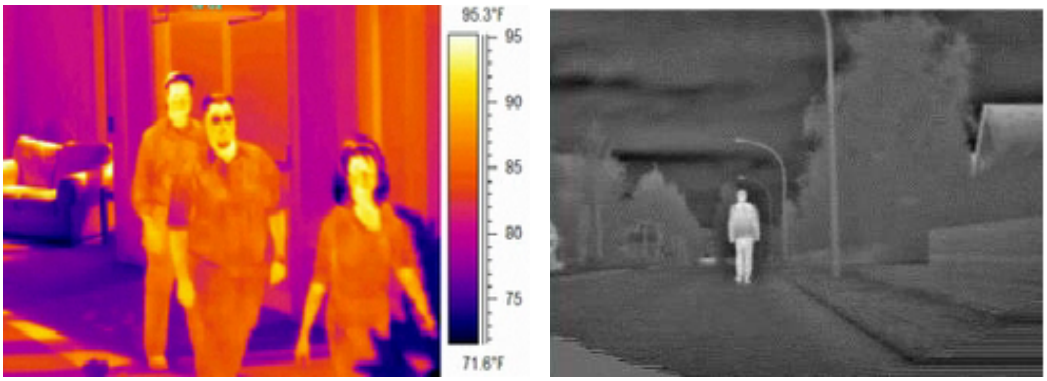
The methods outlined in the previous Subsections aim at providing the 3D position of body parts. While markers and gloves deliver this data directly, the active sensing methods require a preprocessing of the acquired data like, e.g., the Kinect body pose recognition based on the generated depth image. Similarly, vision-based sensing methods require a preprocessing to extract the relevant information. In this Subsection, different passive vision sensors providing image data that can serve as basis for body pose recognition are outlined.

Near-Infrared Cameras

Different from the application of far-infrared cameras together with infrared light sources and reflective markers for obtaining the overall body motion, near-infrared cameras can be used without any active light source for directly extracting the image regions representing warm body parts (Sato et al., 2000). Obviously, this is only possible if the gesturing human is acting in an environment that has a lower temperature (see Fig. 2.9(a)). Similar to other vision-based sensing methods, the images from infrared cameras allow the extraction of the 2D hand position and possibly its shape if the image resolution is high enough. However, as the infrared image pixels only represent relative temperature information, infrared images are not suited for using any appearance-based hand recognition methods as it is common for processing color images (see below). Consequently, the position of the hand can be extracted reliably, but already the details of the hand configuration are difficult to classify based only on the shape of the hand. When it comes to understanding hand actions in context, the limited information from an infrared camera is even more restricting, as it does not provide any detailed information about objects in the environment having normal temperatures. Therefore, and because of the cost of such cameras, they are nowadays less often used for hand gesture recognition, but they are intensively used for the task of human detection in surveillance and automotive applications (see Fig. 2.9(b)).

Color Cameras

Performing vision-based sensing of hand gestures using color cameras is nowadays a common computer vision application. An important aspect boosting vision re-



(a) (b)

Figure 2.9: Images obtained from infrared cameras. a) indoor image with several humans; b) outdoor image showing capabilities for pedestrian detection.

search was the availability of cheap digital cameras for easy interfacing with typical personal computer hardware.

A simplification that has been applied mainly in the beginning of vision-based gesture recognition (see, e.g., Starner and Pentland, 1995) and that makes the extraction of visual features much easier is the use of specially colored gloves (see Fig. 2.10(a)). However, as this requires the gesturer to wear special equipment it is nowadays only rarely used in gesture recognition research.

(a) (b)

Figure 2.10: Gesture recognition using color images. a) setup using a colored glove (image from Wang and Popović, 2009); b) setup where a robot observes a gesturing human.

Nowadays color cameras are typically used to monitor the gesturing human from a distance without requiring him to wear specialized equipment (see Fig. 2.10(b)). Consequently, this non-intrusive sensing method is applicable in a wide range of interaction scenarios. Different from all the other sensing methods outlined above, the image-based analysis of hand gestures seems, at first glance, to resemble how a human observes another human performing gestures. Realizing a human-like processing scheme is not very dominant in today's algorithmic approaches to vision-based gesture recognition but is of greater importance when it comes to analyzing the context of gestures, i.e., incorporating information from other objects in the environment.

Another sensing method being very closely related to color cameras is the use of stereo setups consisting of two color cameras. In such a setup the depth information can be acquired from the disparity between the images of the two cameras while at the same time the color information is directly available. This setup is most closest to how the human perceives its environment. However, the quality of the depth information depends on finding matching structures in both images. This can only be done for textured scenes, while for homogenous image areas no good depth information can be extracted with today's stereo algorithms. Therefore the depth image is usually sparse, i.e., not all image pixels have an associated depth. As the Kinect sensor has shown (see page 35), it is preferable to use a structured light setup if the reliable acquisition of depth data is intended.

Multi-Camera Setups

Instead of sensing the gesturing human only from a single perspective, multiple cameras are used in surveillance and 'intelligent room' setups to observe the gesturer from different positions as depicted in Fig. 2.11.

While the images from a single color camera are directly analyzed, images from multiple cameras can be processed in two fundamentally different ways:

Early fusion: All images are preprocessed to explore the relationship between them. For example, through calculating in textured image areas the image disparities based on multiple images, a representation in the form of 3D points on the human body can be obtained. This information can be used in addition to the color image data from a single camera to infer the body configuration.

Late fusion: Instead of combining the raw data from multiple cameras, in the late fusion approach each camera image is analyzed individually to find, e.g., body parts (Sigal et al., 2004). This eases the detection task as different viewing

Figure 2.11: Images showing gesturing human from different perspectives taken with multi-camera setup (images from Kehl et al., 2005).

angles result in a higher likelihood to detect each body part from a defined perspective. Only after this generation of intermediate results for each camera is the fusion performed to obtain the final processing result.

Obviously, a multi-camera setup providing different viewing perspectives cannot be used with a mobile robot unless an existing stationary multi-camera infrastructure can be accessed by the robot. With the progress in gesture understanding algorithms sensing the gesturing human from a single perspective only, multi-camera setups are, therefore, primarily used today in surveillance applications.

2.2.4 Choosing the Right Sensor

Based on the descriptions of the different sensing methods given above, the sensing method most appropriate for interaction with a social robot can be selected. For a natural interaction a non-intrusive approach is most suited as it allows any user to start interacting with the robot directly without first preparing himself by attaching markers or wearing gloves. While active sensors deliver depth images that provide 3D information about the scene, this data typically lacks color information. However, color images are likely to be needed for recognizing objects in the environment that are an important source of information for the automatic analysis of manipulative and referential gestures.

Nevertheless, depth data from stereo cameras or an active sensor can support the gesture recognition task as exemplified by the Kinect system. Whether or not to use an additional active sensor depends also on the focus of the research: Humans do not have such a sensing capability and only rely on their two eyes and a lot of visual experience to infer depth. Image processing approaches extracting depth

based on image data from a stereo camera are still far from reaching human-like performance, so the use of special depth sensing in addition to a color camera circumvents this issue. At the same time, through using not only depth data but also color image data, a social robot becomes more comparable in terms of its capabilities and is likely to be capable of building up human-like representations that can be communicated in the interaction between the robot and a human. As will be demonstrated in Chapter 6, color image data also allows the extraction of the scene context for understanding manipulative and referential gestures. Before turning in the next Chapters to the details on how to realize vision-based gesture understanding and context incorporation, the next Section will give an overview of design decisions relevant for realizing systems performing the analysis of sensor data.

2.3 Design Decisions for Gesture Understanding Systems

Having outlined the types of gestures to be recognized and the choice of a color camera as sensor, this Section will cover general design decisions for realizing a vision-based gesture understanding system. The vision-based recognition of human body motions has attracted a lot of interest in the image processing community over the years. This has resulted in a huge number of publications each dealing with different aspects of the problem. A number of reviews have tried to summarize the state of the art with respect to motion of the overall body (see, e.g., Moeslund et al., 2006; Wang et al., 2003; Moeslund and Granum, 2001; Gavrila, 1999; Aggarwal and Cai, 1999) as well as with a focus on hands and hand gestures (see, e.g., Erol et al., 2007; Wu and Huang, 1999; Pavlovic et al., 1997). Notably, in both fields some authors of reviews concluded a decade ago that the field of vision-based motion/gesture recognition was still in its infancy (Wang et al., 2003; Moeslund and Granum, 2001; Pavlovic et al., 1997). However, with the increase in computational power and the progress on the sensor side (especially the Kinect sensor outlined in Section 2.2.2), the technological basis for gesture understanding has advanced substantially in recent years. At the same time, the demand for more intuitive user interfaces has driven research efforts. Both aspects have resulted in first commercially available gesture interfaces, most notably the Kinect system. Different algorithmic approaches have been developed for the recognition of static and dynamic gestures and are outlined in subsequent Chapters of this book, while this Section introduces general design decisions for such gesture understanding systems.

2.3.1 Graylevel vs. Color Images

An obvious design decision is the choice of the input data used for hand gesture recognition. In the early days of image processing, all algorithms relied on gray level images for analysis. Color images were rarely used due to the costs of video cameras and, more importantly, the increased computational effort associated with processing three color values instead of one gray value. Nowadays, the computational effort is less relevant in the design decision and the intended application can guide the choice of input data. This choice can be guided by looking at the way in which the human visual system is organized: it has different receptors for luminance and color information and the processing of these two receptor types also differs: luminance is used to perceive motion and to see in three dimensions using both eyes, while the color information is used primarily for object recognition (Encyclopedia Britannica, 2011).

Consequently, when aiming at the detection of hands in single images, it seems obvious that color images are the better choice. While today indeed many approaches for static hand posture recognition use color, there is one major drawback to this in algorithmic solutions: The human visual system has the capability of automatically correcting the perceived color based on the color distribution of the light sources illuminating the scene. This so-called *color constancy* (Funt et al., 1998) is a topic of active research and algorithmic solutions to realize this feature for image processing tasks still have requirements that cannot easily be fulfilled in standard applications (Morel et al., 2009). Consequently, if color image data is used in recognition approaches it is often associated with the implicit assumption that the lighting conditions are fixed, i.e., variations during the interaction have to be excluded. This allows to use offline-trained color representations or to initialize the color model at the beginning of the interaction.

However, often lighting conditions cannot be fixed. For example, even if there seems to be normal lighting from the subjective experience of a human observer, a typical indoor scene is usually not as homogeneously illuminated as it is done for taking professional pictures. As a result, the hand undergoes different lighting conditions while moving in the scene. As a consequence of these challenges many earlier approaches that had to perform robustly in environments where the lighting could not be fully controlled (e.g., rooms with windows) often used gray level hand representations, even if the recording camera was a color camera, and focussed on using shape or motion information. In more recent approaches, in order to cope with the resulting changes of hand appearance, the hand color representations are adapted online to changing lighting conditions.

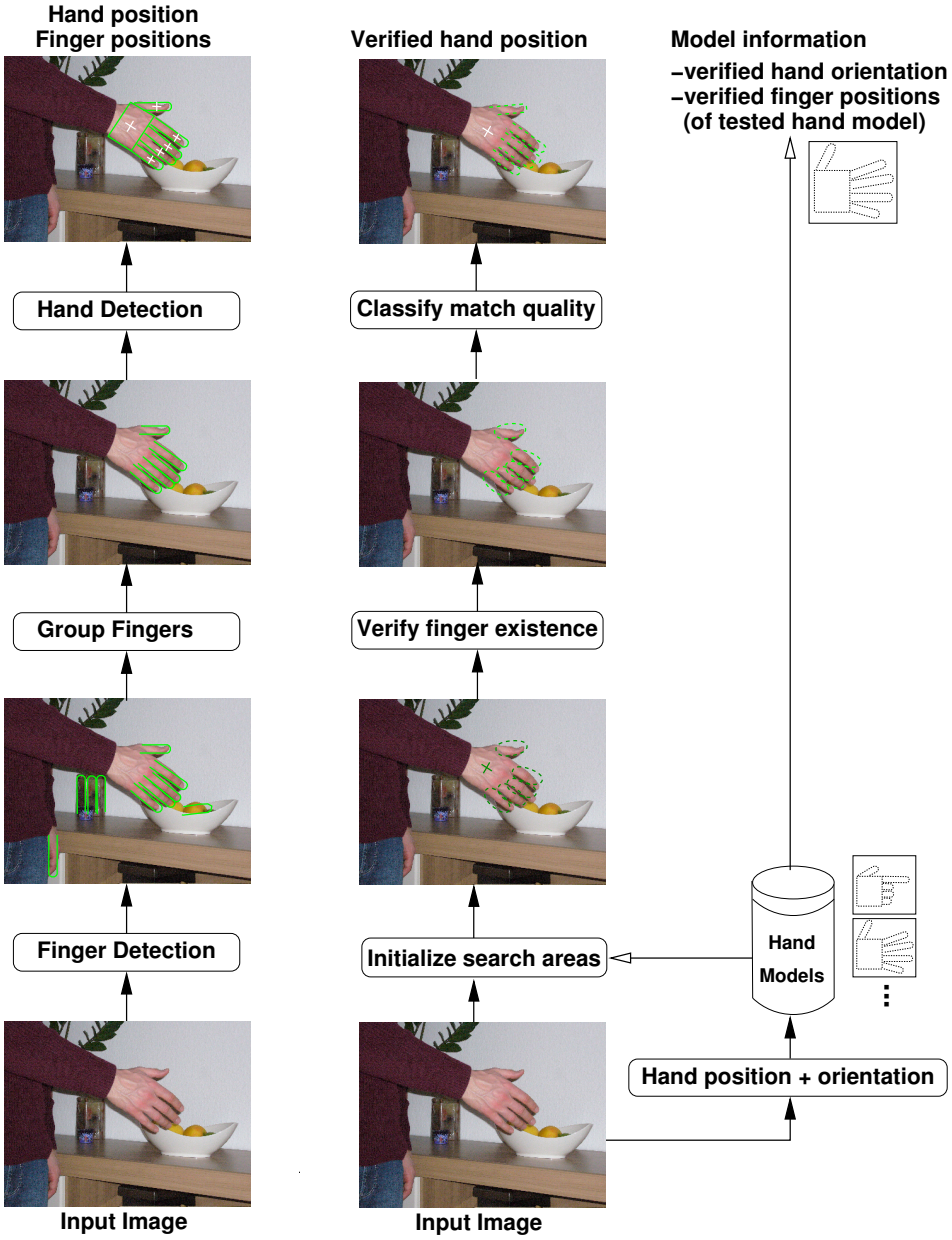
2.3.2 Data-driven vs. Model-driven Processing (Bottom-up vs. Top-down)

Besides the choice of input data, one key characteristic of a recognition algorithm is whether it is data-driven or model-driven. If an algorithm first performs its computations on the input data and constructs out of the results more complex analysis results, this is called a bottom-up processing scheme (Dewey, 2011, Ch. 7). In contrast, if an abstract model is used as representation based on which the algorithm tries to fit the input data to this model, it is called top-down processing. For recognizing hands both methods are in use. For example, in a bottom-up scheme (see Fig. 2.12(a)) generic detectors for individual fingers would be applied and a hand position could be hypothesized based on all finger detection results. In a top-down scheme (see Fig. 2.12(b)) one would select the hand model to be recognized from a database and run four finger and one thumb detector at the estimated hand position to verify or falsify the existence of the fingers. Subsequently, based on the matches of the individual finger detections the decision is made whether the hand model could be matched well enough to the image. If the match is good enough, the hand model from the database is considered to be present in the image.

Both types of processing schemes have different advantages when it comes to building recognition systems:

Data-driven Processing: The bottom-up scheme does not need sophisticated models and allows to start a recognition algorithm on the input data without any explicit a priori knowledge of the gestures to be recognized. Instead, the rules describing how to process the data and how to compose elementary parts together to more complex recognition results have to be defined. Potential drawbacks are the large computational load and the potentially large amount of false positives when running the recognition on any input data. The latter is due to the fact that visual scenes are often ambiguous, i.e., the coded rules have to be general enough to detect all gestures, resulting in misdetections if a visually similar scene is encountered. This problem can be alleviated by incorporating context knowledge (see Section 2.1.5 on page 29) restricting the recognition results to those that are meaningful in the current situation.

Model-driven Processing: In the top-down scheme the recognition process is rather a verification, enabling to concentrate the computational effort on a small part of the input data if the system has expectations of the gestures to be recognized. Consequently, such a scheme is less likely to produce false positives. Through relying on a priori defined models, however, this scheme



(a) Data-driven approach (b) Model-driven approach

Figure 2.12: Exemplary visualization of the different processing schemes. (a) Data-driven (bottom-up) detection of a hand based on detecting its fingers; (b) Model-driven (top-down) search for image structures that match the expected hand model.

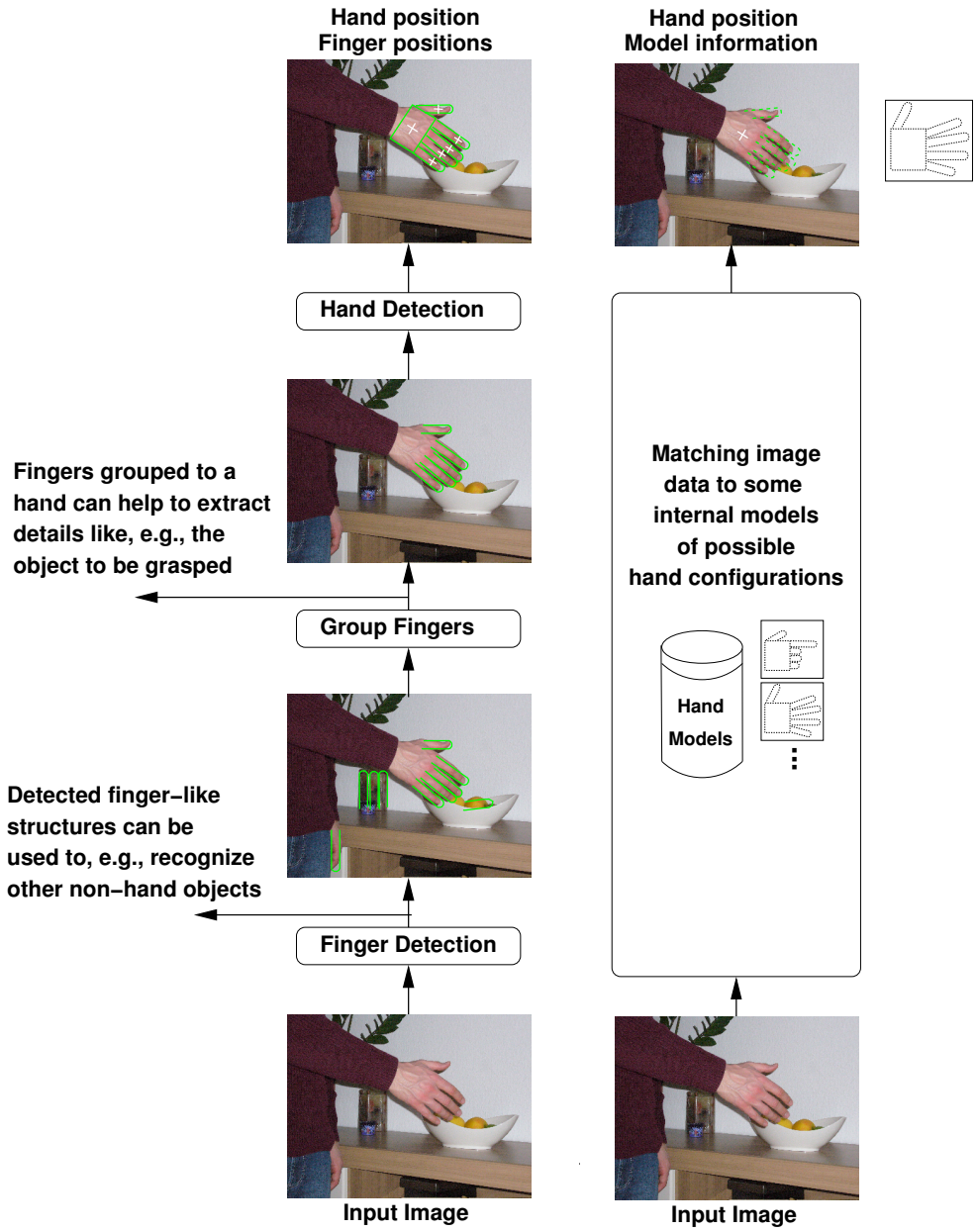
has difficulties with situations that are not captured in the predefined models, i.e., it can result in a larger number of false negatives. For example, if an unknown hand configuration is encountered, none of the hand models can be matched correctly to the image data. Usually the best match will be delivered as recognition result, but in this situation the best match can be a completely different gesture. Another crucial point is the necessity to have a priori knowledge about what model is applicable and where in the image the model is expected. Therefore, the effort of creating appropriate models for top-down processing is a major challenge. Furthermore, depending on the application, the smaller robustness towards unexpected hand configurations can be a major disadvantage.

The description given has tried to point out the differences in both processing schemes. Nevertheless, such a clear separation is usually difficult for categorizing algorithmic approaches: many approaches follow one scheme for the overall organization but incorporate aspects of the other scheme in the individual steps. Consequently, a complete system often incorporates both concepts to a varying degree in its algorithmic realization.

2.3.3 Modular vs. Holistic Approaches

While the previous Subsection covered the direction of the processing flow, another design difference in recognition algorithms concerns the architectural granularity for organizing the recognition process. At the two extremes of this system design decision are A) a modular architecture with several building blocks and B) a holistic approach with very few building blocks.

Modular Architecture: In a modular architecture, the recognition procedure consists of a set of processing steps that have to be carried out to obtain the result. Consequently, the coordination of the different steps and the exchange of data between them has to be engineered appropriately. While this requires additional effort, it has the benefit that the different processing steps can be individually adapted or improved, as intermediate results can be evaluated. Additionally, the results can be used in parallel for a variety of other tasks. For example, in a modular bottom-up approach (see Fig. 2.13(a)) the task of detecting a hand in an image can consist of steps to find skin-colored fingers, group fingers together that can belong to a single hand, and finally classify the grouped fingers with a potential palm area to be a hand.



(a) Modular approach (b) Holistic approach

Figure 2.13: Different architecture designs for hand detection. (a) Modular approach allowing to use intermediate processing results also in other parts of the overall interaction system. (b) Holistic approach prohibiting access to intermediate results, this is usually due to the use of internal models that are themselves holistic.

Holistic Architecture: The flexibility of modular approaches and the parallel use of intermediate results for other tasks is not available in holistic approaches to hand gesture recognition (see Fig. 2.13(b)). For example, the task of detecting a hand in an image is solved in these approaches by matching some kind of appearance-based hand model that has been trained on large amounts of training data beforehand. This means that no intermediate processing results are available, and also no partial detection is possible, which is an important feature when dealing with partial occlusions. While holistic approaches can be adapted to different tasks simply by running the training again on the corresponding data set, they usually lack the feature of online adaptivity. So they are only applicable to the trained task and for every new task a new training set and a corresponding recognizer are needed. Alternatively, the system can be re-trained to cope with several different tasks at once, but here care has to be taken that the holistic representation can actually model the different tasks without becoming unspecific, i.e., too generic for a good recognition quality.

For actually creating a gesture understanding system, the type of architecture is partly determined by the algorithmic approach and partly by the application. With increasing complexity of gestures to be recognized and context to be incorporated, the choice will likely be a modular approach, as this allows incremental coding and testing as well as an easier collaboration if different people are involved in the implementation.

2.3.4 Gesture Recognition vs. Hand Detection/Recognition

The previous Subsections have covered architectural design decisions of gesture recognition approaches. In this Subsection, we want to provide a broad picture of what information is actually extracted from the individual images and how this information is integrated over time. As outlined in Section 1.5, the detection of a hand provides its position, while the recognition includes information about the hand posture. For the recognition of pointing and manipulative gestures that are considered here (see Section 2.1.4), different ways of modeling a gesture are possible.

On the highest abstraction level, the model for the complete gesture can either use some kind of holistic representation implicitly containing time information or it can be represented by a sequence of hand positions/postures over time. In the latter case, the hand has to be found in each individual image for analyzing the overall sequence. As explained before, this task can be supported by using top-down information about the hand position in the last image. Finding the hand

in a single image can require either a simple detection of the hand position or the recognition of the hand posture.

The difference between hand detection and hand recognition is important for recognizing gestures where not only the hand motion but also the hand posture are important. For example, when waving a hand, the number of stretched out fingers makes a difference: all fingers ('hello') or only the index finger ('no no'). However, many gesture recognition approaches are based only on hand detection and ignore the actual hand configuration while the gesture was performed. For the special case of a pointing gesture it should be noted that intuitively the recognition of the hand posture in a single image should be sufficient. However, the entity or object that was pointed at can only be extracted reliably by including trajectory information.

The different levels of processing results (position vs. posture) and the different ways of representing the temporal information to achieve gesture recognition can be matched to different algorithmic approaches. In order to allow a better understanding of the individual algorithmic steps, different Chapters of this book are devoted to the individual steps. However, as pointed out before, there are a variety of ways to realize gesture understanding, therefore these next Chapters are linked together only loosely. Especially for modular approaches, the overall gesture recognition approach is a combination of the algorithms detailed in the individual Chapters: Methods for hand detection and hand recognition in isolated images are described in detail in Chapter 3 while the temporal association of individual results is covered in Chapter 4. On the highest abstraction level, the recognition of gestures without context can be found in Chapter 5. Combining the information of gesturing hands with context can be done on different levels and is described in Chapter 6.

In each Chapter links to the previous Chapters are made where appropriately. With the exception of holistic approaches, this should allow the reader to keep track of the different abstraction levels for realizing gesture understanding.

2.4 Summary

This Chapter has covered the basic issues relevant for following the methods and techniques for gesture understanding described in the next Chapters. Starting from general gesture categorizations, the restriction to the domain of human-robot interaction as well as practical considerations with respect to the state of art have motivated the focus on referential and manipulative gestures. For understanding these two types of gestures, the incorporation of context information into the recognition algorithms is of great importance.

For performing the actual gesture understanding, sensors observing the human gesturer are required. In this Chapter different sensor types have been reviewed in terms of their advantages and disadvantages for enabling a natural human-robot interaction. While classical gesture recognition approaches only required sensing of the human, additional requirements have to be met for understanding the two gesture types considered here: for referential gestures the objects that are the subject of the reference have to be sensed by the robot and for manipulative gestures the objects that are manipulated have to be recognized. These requirements result in the choice of a color camera as best sensor, as it represents the environment in a similar way as a human perceives it.

In order to realize gesture understanding based on camera images, vision-based algorithms have to be applied to the image data. Although there is a wide variety of algorithmic approaches, there are some general design decisions that influence the characteristics of the resulting approaches and have been reviewed in this Chapter.

With the material covered, the reader now has the background knowledge for following the detailed descriptions in the next Chapters, detailing the different algorithmic steps required for understanding gestures in human-robot interaction.

3 Detection of the Hand

A precondition for any gestural understanding process is the availability of information about the gesturing hand. In this Chapter, methods for the extraction of the gesturing hand from isolated images will be outlined. First, the difference between hand detection and posture recognition will be outlined in Section 3.1. Subsequently, a detailed description of bottom-up methods for hand detection based on visual features is provided in Section 3.2 and the use of actual 2D or 3D models for solving the detection task in a top-down manner is covered in Section 3.3.

3.1 Hand Detection vs. Posture Recognition

Obtaining information of the gesturing hand from a single image requires either the detection of the hand position or the recognition of the hand configuration. Notice that a successful recognition implicitly provides the hand position, i.e., it enhances the hand detection with additional information about the hand configuration. In this Chapter, we assume that some kind of representation for the hand is used to detect the hand position or the hand configuration in different images. An example image sequence of a pointing gesture that will serve as reference for this and later Chapters is depicted in Fig. 3.1.

Figure 3.1: Image frames from an example image sequence depicting a human pointing at a coffee mug.

Figure 3.2 visualizes the result of a successful hand detection as white cross in the example images and the recognition of the complete hand configuration is visualized

by a green polygon around the hand shape. In general, the detection task is much easier, as for the recognition task many detailed hand configurations need to be modeled and recognized. For the highly articulated and flexible human hand this poses a challenging problem as it is capable of generating a huge number of different configurations. Therefore, detection approaches applying general features like, e.g., the hand color are usually easier to implement and more robust.

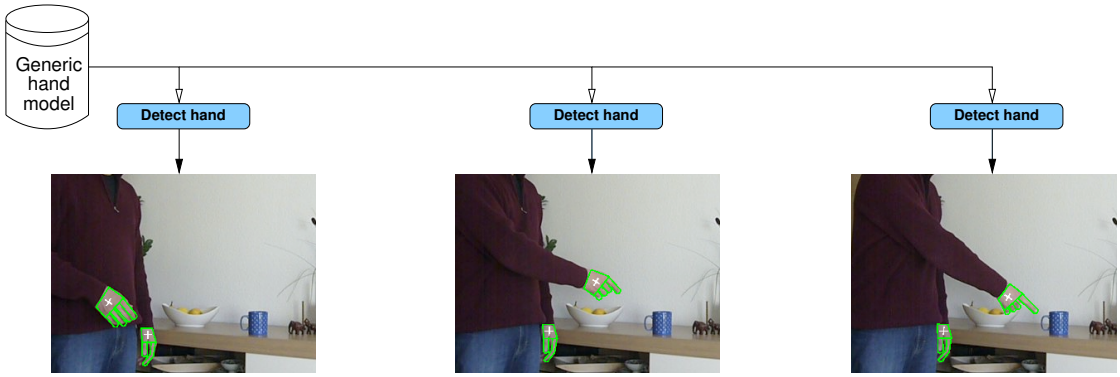


Figure 3.2: Sequence of images depicting the different types of results provided by hand detection algorithms: 1) white cross depicting the detection of the hand position only; 2) green hand and finger shape indicating the detection of specific hand configurations.

Obviously, recognition approaches have to use models for the hand configuration that allow some kind of abstraction, but what is the right level of abstraction? In other words, how many different models are needed to capture all hand configurations relevant to the task? If each gesture to be recognized is characterized by a single specific hand configuration that remains unchanged throughout the gesture, the answer is straightforward. However, the assumption of a fixed hand configuration will not hold for manipulative gestures. The danger of having a limited number of models lies in not successfully recognizing the hand in all images of the sequence. It is therefore advantageous to go beyond the analysis of individual images. The processing of image sequences builds on the methods used for analyzing single images and this Chapter will cover the basic methods for processing individual images while Chapter 4 will then focus on methods for analyzing image sequences.

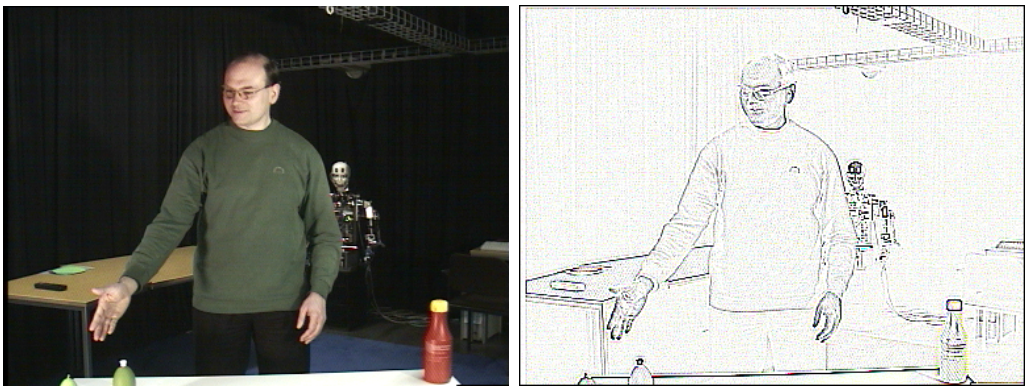
In the following two Sections different approaches aiming at the detection of hands or the recognition of hand postures from single images are reviewed. The approaches are grouped based on the type of modeling they apply. Section 3.2 covers methods directly modeling visual features of the hand while Section 3.3 outlines methods applying some kind of geometric hand model.

3.2 Modeling the Hand's Visual Features

One large body of research is comprised of approaches that model the hand as a whole based on different types of visual features. Given a sufficiently large difference in the visual appearance of the hand to be detected and the background, a straightforward feature of a human hand is its shape. Therefore, approaches using this feature will be reviewed first before dealing with color-based and appearance-based modeling of the human hand.

3.2.1 Hand Shape

The shape of the hand and its outline are easy to detect for humans under nearly every lighting condition. However, extracting the hand contour with computer vision algorithms is much more difficult. This is due to the fact that the decision where the hand is in the image is not obvious when using only basic image features like edges. This is demonstrated in Fig. 3.3 where for an example image the output of an edge detection algorithm is shown.



(a) (b)

Figure 3.3: Example input image and resulting image after edge detection.

Obviously, in a bottom-up approach the amount of detected edges is difficult to handle. Methods performing edge detection directly on the input data are therefore primarily used in top-down approaches applying some kind of model that guides the selection of relevant image areas and edges. Such approaches are the topic of Section 3.3 while we here will concentrate on how to use the hand shape directly in a bottom-up manner.

In order to reduce the amount of features extracted, it is desirable to concentrate on the relevant image area. One way to achieve this is the use of color information,

i.e., to analyze only the skin-colored image areas (see also Section 3.2.2). However, if the image contains not only a hand but also other skin-colored objects like, e.g., a head or wooden parts then color may not be sufficient. A standard approach avoiding an explicit skin color model is to use additional restrictions on the image background.

For example, if only communicative gestures are to be recognized in a non-natural setting, a uniformly colored background like, e.g., a dark wall can be chosen. With such a distinctive background, a preprocessing step in the form of a binarization of the input image results in an intermediate image that only contains the hands and possibly a face. A closely related method is background subtraction where an image of the nearly static scene without the acting hand is learned. This background model can be represented in different ways, for example by modeling each individual pixel as a Gaussian color distribution as shown by Wren et al. (1997). By subtracting from an input image the learned background model, the differences between the learned image and the current image are obtained. If only the gesturing hand is of interest, the background model should contain the human so that the difference is largely the gesturing hand (see Fig. 3.4). This, however, works only if the human does not change his position, otherwise the difference to the original position will cover a large part of the difference image. The image subtraction does not differentiate whether the change comes from a new foreground object or an appearing background object (if this area is covered in the background model by the human in its default position), individual pixels have to be classified as foreground or background in order to obtain a binary image containing the hand area.

Figure 3.4: Result image obtained by performing 'background' subtraction. Note the two left hands resulting from the difference between the background hand position and the current hand position.

Similarly, if hand gestures in front of a projection wall are to be recognized, the projected image can be subtracted from the camera image containing the hand and the projection. In this way, the hand and arm are segmented and can be used to obtain the hand contour (Licsar and Sziranyi, 2005).

Instead of modeling the background or assuming that the hand has a distinctive color, it is also possible to rely on the fact that the human hand is *acting* in the scene, i.e., it is moving. If the background can be assumed to be static, then segmentation algorithms relying on image motion can be used to extract the hand. Extracting image motion can be done by performing image differencing on two subsequent images (Gonzalez and Woods, 2001). The resulting difference image depicts image pixels that have changed as shown in Fig. 3.5. While this method only detects non-constant pixels including background pixels that appear if the hand in the foreground moves away, the more complex optical flow algorithms provide for each individual pixel a motion vector indicating the direction of the motion (see, e.g., Willert et al., 2006). In order to perform contour extraction, such a flow field has to be segmented into 'static' and 'dynamic' areas. Notice that such motion-based approaches can only be applied if the image to be analyzed is part of an image sequence that is captured at a sufficiently high frame rate and if the observing camera is static.

(a) (b)

Figure 3.5: Difference image depicting hand motion. a) binary segmentation result; b) with original pixel data overlaid.

After extraction of a binary segmentation image containing the hand, contour extraction algorithms can be used to get the hand's contour from the hand region (Gonzalez and Woods, 2001). This contour can be transformed in a variety of different representations like, e.g., the curvature of the contour (Kang et al., 2004).

For the actual recognition of hand postures a variety of different representations and pattern matching techniques have been described in the literature. For example, Licsar and Sziranyi (2005) represent the hand contour by Fourier descriptors. A more local approach by Belongie et al. (2002) applies point features capturing at different points of the contour in so-called shape contexts the surrounding contour parts. In order to measure the similarity between contours, the integration of generative models and feature-based approaches in an expectation-maximization framework has been proposed by Tu and Yuille (2004). There are, therefore, many different techniques available to represent hand contours and perform the pattern matching task of associating an extracted representation with stored templates. However, no single technique has proven to be the ultimate solution for the wide variety of interaction settings and hand configurations.

Note that all methods described above for extracting contours are not well suited for human-robot interaction in unconstrained environments as they place constraints on the image background. As it cannot be assumed that the background has a certain color or is a static scene without other moving humans, approaches extracting the hand shape have primarily been used for human-computer interaction where the hand gestures were performed above an empty table or in front of a projection screen or static background.

3.2.2 Skin Color

Another way to guide the search for a hand in a color image is to use skin color. An important advantage is that color is a rotation and scaling invariant feature. For the detection of a human hand, skin color can be used to simplify the hand region segmentation process as it does not require image motion and can cope with arbitrary variations in the background as long as they are not skin-colored. Several researchers have carried out basic studies on the properties of skin color and methods to model skin color to perform pixel classification. We will review in the following some important contributions to this field.

Representing Skin-color

A basic paper by Yang et al. (1998) analyzes the color properties of face images based on a database containing about 1000 faces of people of different races. In their work three properties important for modeling skin color are identified:

1. Skin color is clustered in a small region in a color space, i.e., the individual color values are not randomly distributed. The clustering property is inde-

pendent of a specific color space, only the compactness of the cluster varies for different color spaces (see also J.-C. Terrillon and Akamatsu, 2000; Zarit et al., 1999)).

2. The variance of the skin color distribution can be reduced by intensity normalization. Choosing a color space with intensity normalization is therefore advantageous for modeling skin color. Yang et al. propose the *normalized color space* that is obtained by removing the luminance from the color representation through normalization of the individual RGB values:

$$r = \frac{R}{R+G+B} \quad g = \frac{G}{R+G+B} \quad b = \frac{B}{R+G+B} \tag{3.1}$$

   As the value for b can be calculated based on the values of r and g with $b = 1 - r - g$, it does not contain additional information and the normalized color space is therefore in the following referred to as *r-g color space*.

3. The skin color distribution for a specific lighting condition can be characterized by a multivariate normal distribution in the normalized color space.

In another study focussing on skin color present in isolated pictures collected from the internet, Jones and Rehg (1999) analyze a total of 4675 images containing skin. They compare two methods for modeling normalized skin color, namely mixture models and color histograms. Their results indicate that Gaussian mixture models are a feasible approach to recognize human skin in environments with limited training data but are outperformed by histogram models otherwise.

An important aspect that must be noted is the fact that studies using photographs do not represent the image quality provided by a normal camera mounted on a mobile robot. Pictures taken by a human photographer have a superior image quality. The images of an image sequence taken automatically during observing a gesturing human may exhibit, for example, heavy shading due to insufficient lighting. Therefore the skin pixels contained in such images will cover, compared to pixels from a photograph data set, a wider range of the color space that has a larger overlap with non-skin pixels making a discrimination more difficult. A solution to this problem is the use of a temporary skin color model that is continuously adapted to the current lighting situation (see also Section 4.3.1).

Representing the Global Skin Distribution: the Skin Locus

Störring et al. (1999, 2001) focus in their work on the properties of skin color in faces of different ethnical subjects under changing lighting conditions. Their study

verifies a physics-based model of skin color that predicts the appearance of skin color under varying lighting conditions. For this purpose images of different people under different illumination conditions are captured. The test set consists of seven subjects from around the world (Latvia, Denmark, Greece, Spain, China, Iran, India and Cameroun) to capture all possible variations of skin type.

Controlling the illumination conditions and knowing the spectral sensitivity of the camera allows to prove the validity of a theoretical framework modeling how skin color distributions are affected by changing illumination conditions. The relation between changes in the lighting condition and the resulting changes in the mean skin color chromaticity can be seen in Fig. 3.6. The distribution of the individual skin color values under the four illumination conditions for the Caucasian skin type (from Latvia) can be seen in Fig. 3.7.

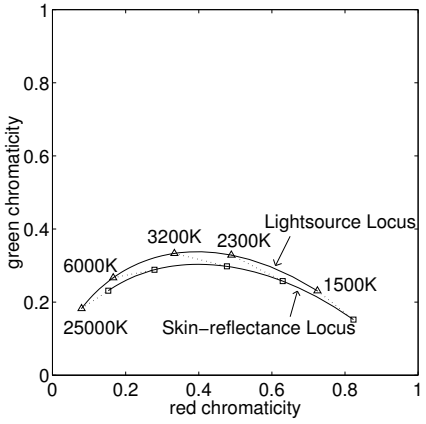


Figure 3.6: The location of light sources and corresponding mean values of skin color areas calculated with a theoretical model for different color temperatures (image from Störring et al., 1999).

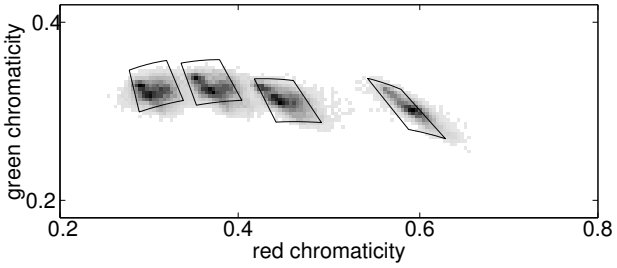


Figure 3.7: The distribution of the individual skin color values for the Caucasian skin type from Latvia (image from Störring et al., 1999).

Following the theoretical model, it is shown that the area occupied by the skin color distribution of all different skin types under all possible lighting conditions occupies a shell-shaped area in the r-g normalized color space. This area can be modeled by two quadratic functions and is referred to as the *skin locus* in the following.

Using the Skin Locus

For real applications the area of the actual skin locus can be measured from labeled training images for all relevant illumination conditions. This avoids knowing the spectral sensitivity of the camera sensor. Using a measured skin locus allows to realize a preprocessing step in skin color segmentation approaches by discarding all pixel values that are not contained in the measured skin locus (Soriano et al., 2000). The size of the measured skin locus depends on the skin color of the training subjects and the illumination conditions of the training images. With such a skin color filtering, it is possible to obtain the hand region in the image (see Fig. 3.8). By performing a connected-components analysis (Gonzalez and Woods, 2001) on the segmentation result, a polygonal description of the hand region can be obtained and the center-of-mass of the region represents the hand position.

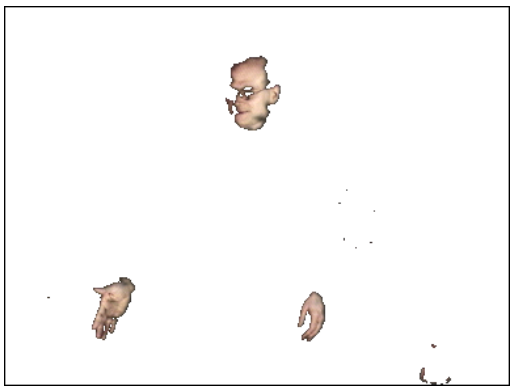


Figure 3.8: After filtering an image with a skin color model it only contains skin-colored pixels.

However, often a rather coarse skin color model has to be used to guarantee that the hand is found under varying lighting conditions. Consequently, other image parts having a color close to the skin color are also often contained in the segmentation result. The combination of a skin color segmentation algorithm with background modeling or motion detection techniques is therefore often used to improve the segmentation result. Alternatively, a posture recognition can be carried

out by applying a shape analysis on the segmentation result (see Section 3.2.1) or analyzing the region for hand-like features using model-based hand detection methods as outlined in Section 3.3.

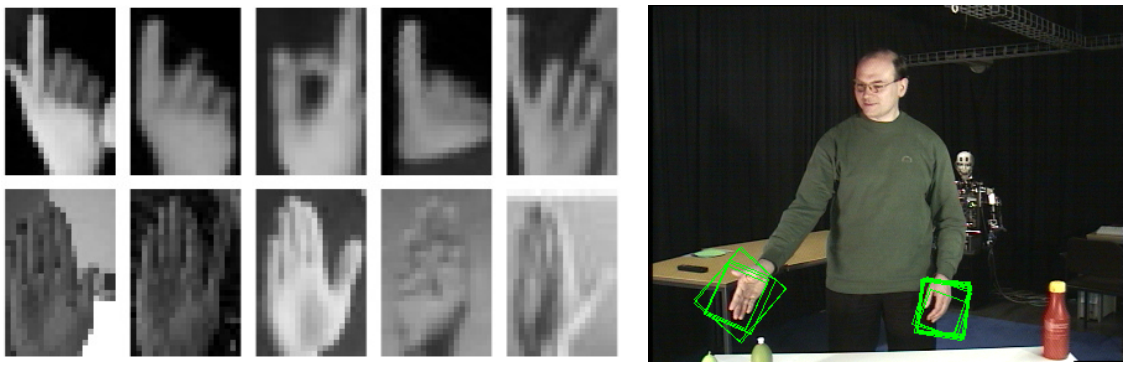
3.2.3 Hand Appearance

Instead of separating the hand from the background and relying for the recognition on a correct segmentation result, methods modeling the hand appearance directly use the complete image area depicting the hand as input. The simplest modeling of the hand appearance is to use the image directly as a template and apply a cross correlation technique to perform template matching (Gonzalez and Woods, 2001). The templates implicitly contain some part of the background, but as long as this part is small and has no explicit structure, the cross correlation technique allows a good matching performance. The drawback of this method is its sensitivity to changes in the appearance caused by the flexibility of the hand or lighting variations.

A technique that has been successfully applied for face detection and is less dependent on the appearance of the individual training images is the *Eigenfaces* approach introduced by Turk and Pentland (1991). Here, an image is considered as a vector and the covariance of a large set of such vectors, i.e., many face images, is analyzed to obtain its eigenvectors. As this technique does not require the images to contain faces and has been applied to other recognition problems as well, its more generic name is *Eigenimages*. Following its success in face recognition, it has also been applied to detect hands under the term *Eigenhands* (Horimoto et al., 2003).

Another technique that has proven successful for face detection was originally proposed by Viola and Jones (2004) and applies a boosted cascade of very simple features calculated on image patches. Through applying the AdaBoost algorithm from machine learning during training, a strong classifier is obtained as a combination of several weak classifiers. Through a careful implementation in the form of a cascaded classifier and the choice of simple features this approach is very fast and robust. Due to its simplicity, many applications have been published that use this approach for recognizing different hand gestures (see, e.g., Kölsch and Turk, 2004; Wang and Wang, 2008). Figure 3.9 shows example training data and an image where several successful detections are visible.

Besides the above mentioned techniques inspired from face detection there are many more specialized feature extraction methods applying a wide variety of different representations (see, e.g., Heidemann et al., 2004; Lu et al., 2005; Chuang et al., 2011).



(a) (b)

Figure 3.9: Hand gesture training data and detection results. (a) Training examples for *pointing* (top row) and *stop* (bottom row) gestures; b) Detection results of classifier.

For example, Heidemann et al. (2004) describe a processing chain to classify the pointing direction of hand postures on a dark background. The raw images are preprocessed by a vector quantization stage and a subsequent local principal component analysis (PCA) to obtain feature vectors. For classification of the feature vectors a variant of a self-organizing map called local linear map is applied.

To give another example, Lu et al. (2005) propose to use the Fourier transform of a circular spatial histogram to represent the hand posture. A skin color model is applied to find the hand position and six histograms are constructed based on differently sized rings around this position. The six histograms are concatenated to form a feature vector and a Fourier transformation is applied to obtain the final feature vector for matching with stored templates.

The various approaches are difficult to classify in a specific category and, more importantly, there is no common benchmark that would allow to compare their recognition quality. Many appearance-based feature extraction methods are rather engineered for a specific setting and have not found wider applicability. A general trend, however, in appearance-based approaches is the use of machine learning techniques to handle the large amounts of training data necessary to achieve robust hand detection. Since the pioneering work by Viola and Jones (2004) using Adaboost for automatic feature selection and classification, this is a dominant approach that is replacing the hand-engineered feature design in recent years.

3.3 Model-based Hand Detection

Instead of a purely bottom-up detection approach based on general feature extraction, the knowledge that the hand to be detected is observed in specific configurations and has specific constituents can be used in a top-down detection process. Modular approaches model the hand as consisting of several parts, notably fingers, and are reviewed in Section 3.3.1. In contrast, holistic approaches represent the visual appearance of the hand as a whole, these approaches are the topic of Section 3.3.2.

3.3.1 Explicit Modeling of the Hand Constituents

Detecting the hand in a modular approach amounts to successfully detecting its parts. Detecting the fingers can be done based on the approaches outlined in the previous Section 3.2 by analyzing the data locally instead of globally. For example, instead of searching for a complete hand shape, the same feature extraction technique can be used to search for a finger shape. If this is done without any a priori knowledge of the rough hand position, this would again resemble a bottom-up processing scheme. However, as the finger shape is not as specific as the shape of the overall hand, more false positive detections would result from such a bottom-up approach. Consequently, detection approaches aiming at the hand constituents are usually based on the assumption that the rough hand position is available as top-down information to restrict the search space (see Fig. 3.10).

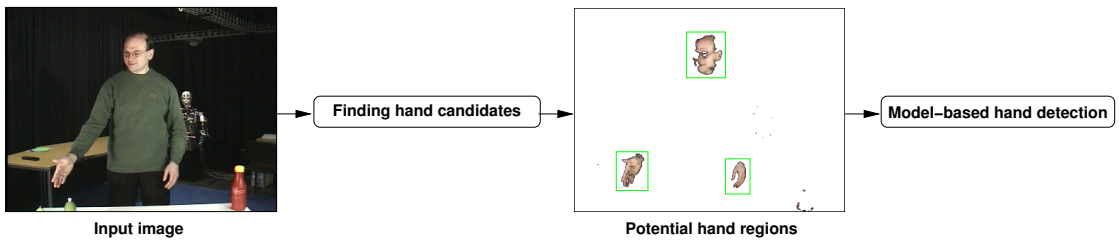


Figure 3.10: Global hand hypotheses as precondition for local model-based hand detection.

The rough hand position can just serve as a spatial prior or it can already provide preselected data like, e.g., the segmented skin-colored area, on which to perform the algorithmic steps for hand detection. For example, von Hardenberg and Berard (2001) assume that the hand segmentation is available and analyze the resulting hand region to find finger-like region parts. Figure 3.11 depicts such a local

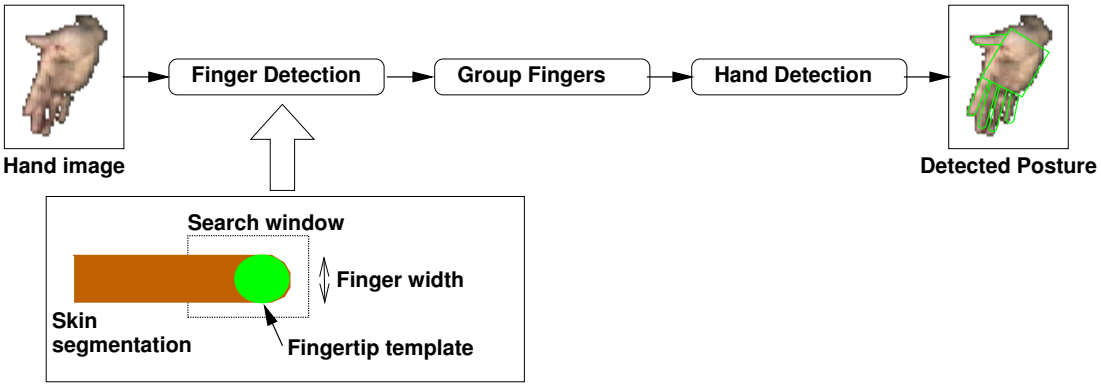


Figure 3.11: Local model-based hand detection applying a parametric model for fingertip detection.

model-based hand detection relying on the detection of individual fingertips using a parametric model.

By using an explicit hand model, the overall posture can be classified correctly if the right number of fingers, i.e., the number of fingers visible in a specific gesture, is successfully detected. Such an exact modeling of the fingers relies on the correct hand view, i.e., the complete visibility of the hand in its 'normal' orientation. If the hand has some kind of default orientation, i.e., if the gestures are performed over a desk or in front of a wall, such constraints may hold. However, for unconstrained gesturing of a hand like in human-robot interaction such an explicit hand modeling may not be appropriate due to self-occlusions of the fingers. Similarly, when a human manipulates an object, this usually results in some fingers being occluded by the manipulated object.

Taking the idea of model-based hand detection further, more complex hand models have been applied in recent studies. Instead of simply modeling the number of fingers, the hand is modeled as a kinematic model consisting of the palm with five fingers including all joint angles of the individual fingers. Consequently, such a model contains many degrees of freedom, typically around 27. Detecting with such a model the hand posture in a single image becomes a high dimensional search problem.

One way to avoid dealing with the full model complexity directly is to use detectors for the fingertips and apply inverse kinematics to obtain the joint angle configuration. For example, Nölker and Ritter (1999) apply fingertip detectors in gray level images to get the 2D fingertip position. Using a parameterized self-organizing map, the mapping between the observed 2D fingertip location and the finger joint angles is realized.

Technically, it is also possible to use the full 3D complexity of a hand model and try to map the expected 3D shapes of its constituents in a top-down approach to visually observable features. If only 2D image features are available, such an approach has to cope with possible self-occlusions and ambiguities in the 2D appearance of the 3D posture. Approaches performing such model matching are therefore usually not applied to single images but to image sequences where the hand configuration detected in the previous frame serves as top-down information to initialize and restrict the search space (see Section 4.5.1).

3.3.2 Holistic Models of the Hand

Instead of modeling the hand by its components and their relations, some algorithms apply a holistic representation of the complete hand that is constructed from training data. This type of approaches could also have been listed in the bottom-up detection approaches in Section 3.2.3, as they usually rely on the hand appearance. However, by learning a representation that realizes an abstraction from the raw appearance data, these approaches are different from a direct template-based hand representation.

For example, Triesch and von der Malsburg (2001) model a hand posture by a graph where each of the 15 graph nodes represents the appearance of a small part of the hand. The 15 node positions are chosen manually for each of the 12 different postures to be recognized. For each node position a local image descriptor is learned from training data. During recognition, elastic graph matching is applied to find the graph that matches the input image best. As this technique does not require any image segmentation, it can be applied to arbitrary input images. The approach can cope with arbitrary backgrounds and exhibits high recognition rates for a limited set of hand commands. Note that this does not represent a modular approach, as the graph nodes are chosen based on the task at hand and are therefore a holistic hand representation.

A more sophisticated approach to detect a hand is the use of a sophisticated 3D model of the hand. Different from the modular approach of projecting the hand constituents to 2D shapes, in a holistic approach the complete 3D hand configuration is projected to a 2D shape (see Fig. 3.12). Different from the direct shape-based detection introduced in Section 3.2.1, this holistic approach is based on an underlying complete 3D hand configuration. It can therefore be adapted more easily to different hand sizes and capture the perspective occlusions better. The most important advantage of this approach, however, comes only into play if it is used for tracking a hand: having detected the 2D hand shape of an associated

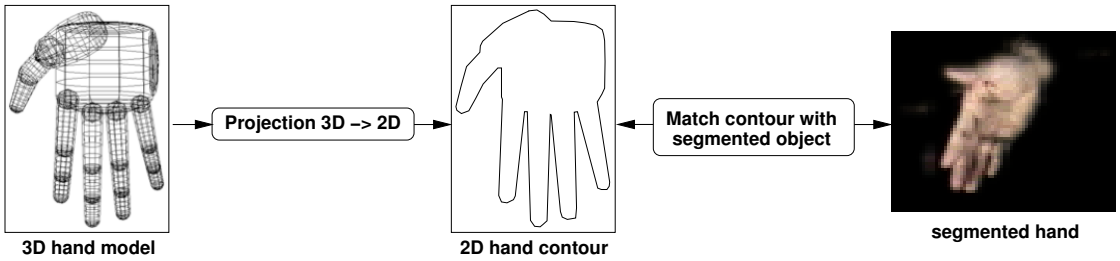


Figure 3.12: 3D model of the human hand projected to a 2D shape for matching with the pre-segmented image data.

3D posture in the last image can be used to calculate based on the 3D model all possible next configurations and then match all resulting different 2D shapes to the image data to find the best match (see, e.g., Stenger et al., 2001). Consequently, this method will be covered in more detail in Section 4.5.1.

3.4 Summary and Conclusion

In this Chapter a variety of different approaches for the detection of hands in single images have been reviewed. The most basic approaches are based on visual features that are characteristic for the hand and can be extracted easily from the image data. The hand shape and hand color are two features aiming at such a segmentation of the hand. However, aiming at a good segmentation of the hand from the background in individual images is challenging as assumptions on the image background can usually not be made in natural human-robot interaction scenarios. Compared to shape and color, appearance-based detection of a pre-trained hand patch just provides a rough position, but its advantage is the larger independence to varying image backgrounds. This, however, comes at the cost of training all hand postures to be recognized beforehand while especially the color cue is generic for all hand postures.

If a hand has been segmented, it can be used for an additional model-based processing step detecting the individual hand constituents to verify the hand detection and provide additional details of the hand configuration. This second step is very useful to reject wrong hand detections of objects with a skin-like color or hand-like shape.

A more holistic model-based hand detection approach aims at using the knowledge of the 3D hand model to predict the 2D shape of the hand. However, without any information on the approximate hand configuration, the detection of arbitrary

hand shapes in single images is very difficult due to self-occlusion. For this task an image sequence needs to be analyzed so that temporal constraints can be used to resolve ambiguities. This applies to all hand detection techniques and, therefore, the next Chapter will turn to methods for tracking the hand in image sequences.

4 Tracking of the Hand

While the detection methods outlined in the previous Chapter provide information about the hand position and possibly also its configuration in an isolated image, this Chapter deals with the tracking of the hand over a complete image sequence. First, the differences between tracking approaches relying on a separate detection process and tracking with continuous adaptation of the detection model will be pointed out (Section 4.1) and classical tracking algorithms will be reviewed (Section 4.2). Subsequent Sections cover algorithmic approaches for adaptive hand tracking based on feature adaptation (Section 4.3 & 4.4) and model adaptation (Section 4.5 & 4.6) based on the detection methods introduced in Chapter 3. A summary in Section 4.7 concludes the Chapter.

4.1 Detection vs. Adaptation

Assuming a successful detection of the hand in all images of an image sequence using the methods outlined in Chapter 3, the trajectory of the hand can be reconstructed. This amounts to a bottom-up approach where no a priori information is used, i.e., detecting the hand in the current image is not influenced by knowledge about the hand position and configuration in the previous image. This "tracking by detection" is a straightforward approach to obtain a trajectory and classical tracking algorithms for realizing hand tracking based on a successful hand detection are reviewed in Section 4.2.

The "tracking by detection" approach relies on the assumption that the hand can be detected successfully in individual images. However, this is a challenging requirement for the detection task and is not feasible in several typical situations:

- The appearance of the hand may change due to environmental conditions - like a partly shaded scene - prohibiting the use of simple detection methods like, e.g., a fixed skin color model.
- The configuration of the hand may change during the gesture. For example, some fingers may become invisible due to self-occlusion. This prohibits the use of a fixed posture detection model.

Obviously, only for very restricted settings the application of hand detection methods to all images of a sequence in order to achieve hand tracking is feasible, while for natural human-robot interaction settings more sophisticated approaches are required. One solution is the continuous adaptation of the hand representation to incorporate a changing hand appearance or hand configuration in the detection model, making the task of detecting the hand in the next image easier. Such an adaptation requires to keep track of the hand of interest, i.e., to know which of the detected objects is the hand that is represented by the current hand model and where the input data can be used for updating the model. This association is the core tracking process. For the sake of clarity, we will here make an explicit separation between tracking of a hand and model adaptation. In Fig. 4.1 the tracking and adaptation steps that are in the focus of the following Sections and that go beyond the use of a generic hand model as introduced in Chapter 3 are visualized with blue boxes.

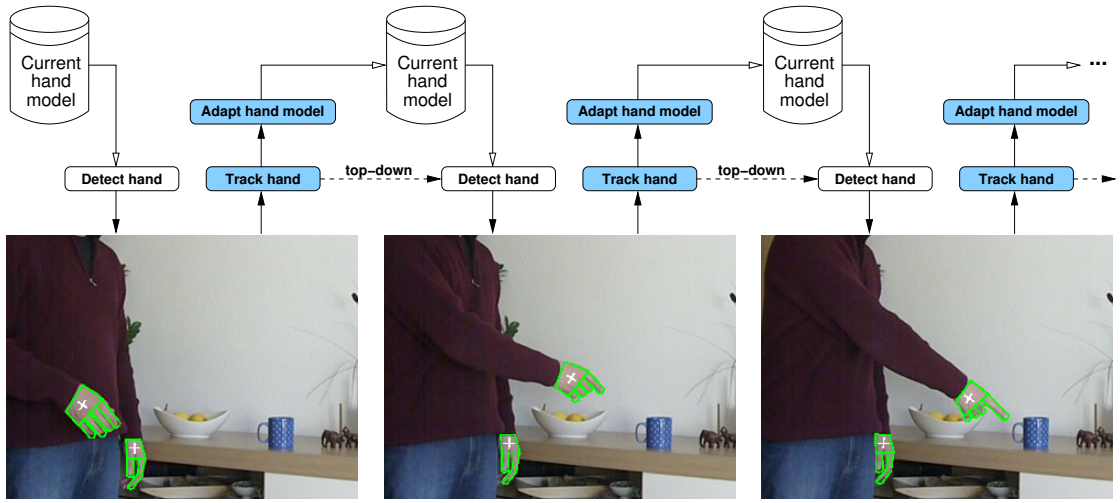


Figure 4.1: Improving detection-based sequence analysis (cf. Fig. 3.2 on page 52) by adapting the hand model to the current appearance/configuration of the hand.

Under the assumption that the hand has been tracked successfully, the adaptation of the hand model depends on the detection method chosen. For the detection methods based on color and shape features (see Section 3.2), the adaptation can be realized by iteratively updating the mean values and variances of the feature values. Approaches realizing such a feature adaptation will be described in Section 4.3 and an instructive example for adapting a skin-color representation to ease the tracking task is described in Section 4.4.

Similar to extending the feature-based hand detection algorithms by adapting the hand representations, also the model-based detection algorithms introduced in Section 3.3 can be extended to enable adaptation. As the underlying model is more sophisticated, also the benefit from adaptation is usually higher: knowing, for example, the current hand or body configuration in 3D allows to incorporate this as top-down information by predicting all possible next configurations in 3D. The 2D image features resulting from the predictions can then be calculated and matched to the image data. Approaches tracking an internal 3D representation for matching with observed image data provide a suitable solution to cope with the ambiguity inherent in 2D image data and will be reviewed in Section 4.5. An example system performing the tracking of the upper body of a gesturing human based on matching 3D models is covered in more detail in Section 4.6.

4.2 Tracking based on Hand Detection

In its most simple form, the tracking process has the task to associate the hand position of the last image with one of the detected hand hypotheses in the current image. As the hand may be currently moving, the closest match is not always the best match, so dynamic information about the past motion has to be incorporated in this process. Two standard tracking methods that are frequently used will be described exemplarily in the following Subsections, namely Kalman filtering in Section 4.2.1 and particle filtering in Section 4.2.2.

4.2.1 Kalman Filtering

Tracking a moving object with a Kalman filter is a standard technique widely applied in many domains. In a Kalman filter the state pdf $p(q_t)$ is modeled as a single Gaussian with mean μ_t and covariance σ_t (Bar-Shalom and Fortmann, 1988). In general, the state pdf of a Kalman filter can be a vector, but for simplicity we will consider here as example only the one-dimensional case (see Fig. 4.2).

The state q_t is hidden, i.e., it is not directly observable, and it is related to the observation z_t by a transfer function h_t that is a scalar value for the one-dimensional case:

$$z_t = h_t q_t \tag{4.1}$$

Given a state q_{t-1} at time $t-1$, predictions of its Gaussian parameters $(\hat{\mu}_t, \hat{\sigma}_t)$ for time t can be calculated using the system dynamics a_t and the uncertainty of

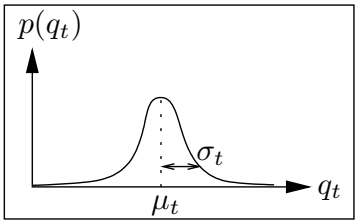


Figure 4.2: The unimodal Gaussian state probability density function modeled by a Kalman filter.

the state prediction modeled with the covariance σ_q:

$$\hat{\mu}_t = a_t \mu_{t-1} \tag{4.2}$$
$$\hat{\sigma}_t = \sigma_{t-1} + \sigma_q \tag{4.3}$$

Similarly, the observation can be predicted to time t using the transfer function h_t and the uncertainty of the measurement process modeled with σ_z:

$$\hat{z}_t = h_t q_{t-1} + \sigma_z \tag{4.4}$$

The parameters of the Gaussian density function can now be propagated over time by fusing the predictions of the parameters $(\hat{\mu}_t, \hat{\sigma}_t)$ and the prediction of the observation $\hat{z}_t$ with the actual observation z_t using the *Kalman gain* k_t:

$$\mu_t = \hat{\mu}_t + k_t(z_t - h_t \hat{\mu}_t) \tag{4.5}$$
$$\sigma_t = \hat{\sigma}_t - k_t h_t \hat{\sigma}_t$$

In this fusion process, the predictions and observations are weighted by their estimated uncertainties. To minimize the a posteriori error covariance σ_t in Eq. 4.5 (for details see Bar-Shalom and Fortmann, 1988), the Kalman gain k_t is chosen to[1]:

$$k_t = \frac{\hat{\sigma}_t h_t^T}{h_t \hat{\sigma}_t h_t^T + \sigma_z} \tag{4.6}$$

The Kalman filter as introduced above is an optimal estimator if the state density and the observation density are Gaussian and, consequently, all densities stay Gaussian during the propagation. In this case, the Kalman filter results in an optimal estimator for the error variance, and the state mean is equal to the most probable state.

[1]For a state vector $\mathbf{q}$, the transfer function h_t^T would need to be the matrix transpose H_t^T of the transfer function H_t but in the one-dimensional case is $h_t^T \equiv h_t$.

However, in real applications the state density often exhibits multiple modes due to, e.g., noise in the observations. The Kalman filter is not able to track state densities containing multiple modes. Therefore, several extensions to the standard Kalman filter have been proposed. For example, approaches for tracking of multiple hypothesis represent multimodal distributions with a series of Kalman filters where each filter is responsible for one hypothesis (Bar-Shalom and Fortmann, 1988; Cham and Rehg, 1999). In these 'multiple hypothesis tracking' approaches, specific functionalities are needed to add and remove Kalman filters if new hypotheses arise or old hypotheses can be discarded. While this extension does allow a limited form of multi-modality, it comes at the cost of explicitly handling hypothesis addition and removal, calling for a more elegant solution to handle ambiguities during tracking.

4.2.2 Particle Filtering

While the Kalman filter is well suited for tracking in situations were the probability density function is a Gaussian distribution, a full probabilistic representation is often better suited to cope with the noise and uncertainty typically encountered in vision-related tracking tasks. For coping with such non-Gaussian probability distributions, particle filters have gained a lot of interest in the last decade.

Standard Particle Filtering

The standard particle filter (PF) is a probabilistic framework that has demonstrated to be well suited for the challenges outlined before and which is used in several of the example approaches contained in this book. It models a probability density function (PDF) based on a set of particles that represent a number of different hypotheses and are propagated over time. The following gives a short overview, more details can be found in the tutorial by Arulampalam et al. (2002).

In recursive Bayesian estimation, the posterior PDF is estimated by propagating the PDF over time:

$$p(\mathbf{x}_t \mid \mathbf{Y}_t) \propto p(\mathbf{y}_t \mid \mathbf{x}_t) p(\mathbf{x}_t \mid \mathbf{Y}_{t-1}) \tag{4.7}$$

In particle filtering, a discrete representation of the PDF is represented as a weighted set of particles as depicted in Fig. 4.3. Given in the d-dimensional space $\mathbb{R}^d$ a particle set $\mathbf{S}_{t-1} = \{\mathbf{s}_{t-1}^{(n)}\}_{n=1}^{N}$ and associated weights $\{\pi_{t-1}^{(n)}\}_{n=1}^{N}$, that are normalized to $\sum_{n=1}^{N} \pi_t^{(n)} = 1$, the PDF can be approximated by:

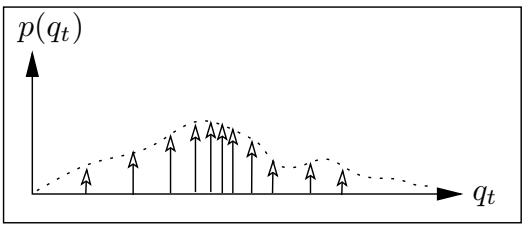


Figure 4.3: The approximation of a state probability density function by a particle filter applying a weighted set of discrete particles to model the PDF.

$$p(\mathbf{x}_t \mid \mathbf{Y}_t) \propto \sum_{n=1}^{N} \delta(\mathbf{x}_t - \mathbf{s}_t^{(n)}) \pi_t^{(n)} \tag{4.8}$$

To obtain a new particle representation of the posterior PDF in Eq. 4.8 the weights of the particles are updated at each iteration using the principle of *importance sampling*. If a proposal – the importance density $t(\cdot)$ – can be found from which it is easy to draw samples, it can be shown (Arulampalam et al., 2002) that the weights can be calculated sequentially from:

$$\pi_t^{(n)} \propto \pi_{t-1}^{(n)} \frac{p(\mathbf{x}_t|\mathbf{s}_t^{(n)}) p(\mathbf{s}_t^{(n)}|\mathbf{s}_{t-1}^{(n)})}{t(\mathbf{s}_t^{(n)}|\mathbf{s}_{t-1}^{(n)}, \mathbf{x}_t)} \tag{4.9}$$

In the specific variant of Sequential Importance Resampling (SIR) that was initially introduced to the computer vision community as CONDENSATION by Isard and Blake (1998), a re-sampling step using the new weights is applied in every iteration to avoid degeneration of the particle based representation. The overall propagation of the weighted samples over time consists therefore of three steps and is based on the results of the previous time step:

Select: Selection of N samples $\mathbf{s}_{t-1}^{(n)}$ according to their respective weight $\pi_{t-1}^{(n)}$ from the sample pool $\{(\mathbf{s}_{t-1}^{(1)}, \pi_{t-1}^{(1)}), \ldots, (\mathbf{s}_{t-1}^{(N)}, \pi_{t-1}^{(N)})\}$ of the previous time step. This selection scheme implies a preference for samples with high probability, i.e., they are selected more often.

Predict: The parameters of each sample $\mathbf{s}_t^{(n)}$ are predicted using a temporal model for the state propagation. For example, if a linear motion model is assumed, the previous position of a hand would need to be increased by the distance supposedly travelled between two time steps.

Update: Determination of the weights $\pi_t^{(n)}$ based on $p(\mathbf{z}_t|\mathbf{s}_t^{(n)})$.

Note that in high-dimensional search spaces, even if a particle escapes out of a local minimum, the probability of hitting the low-weight surroundings is much larger than that of hitting a region with high weights. This is due to the huge increase of volume with radius in high-dimensional spaces that results in the need for a large number of particles which in turn increases computation time.

Kernel-Based Particle Filtering

In order to overcome the need for many particles in high-dimensional search spaces, many different variants of particle filtering have been proposed (for a recent overview see Doucet and Johansen, 2011). One influential extension added iterative mode-seeking in the form of the mean-shift algorithm (Chang and Ansari, 2005) to shift the particles to high weight areas and is widely referred to as kernel-based particle filtering. This has been utilized first in experiments consisting of tracking objects or isolated body parts in the 2D image space (Han et al., 2005; Chang and Ansari, 2005). In this extension, the true density distribution is estimated through placing a kernel function at each sample position. The estimate of the posterior PDF with kernel K can be formulated as:

$$\hat{p}(\mathbf{x}_t \mid \mathbf{Y}_t) = \sum_{n=1}^{N} K_h(\mathbf{x}_t - \mathbf{s}_t^{(n)})\pi_t^{(n)} \tag{4.10}$$

where $K_h(\mathbf{x}_t - \mathbf{s}_t^{(n)}) = \frac{1}{Nh^d}K(\frac{\mathbf{x}_t - \mathbf{s}_t^{(n)}}{h})$, and h is the kernel bandwidth. How a PDF can be approximated by a number of kernels is shown exemplarily in Fig. 4.4 for different kernel sizes.

For a radially symmetric kernel we have $K(\mathbf{x}_t - \mathbf{s}_t^{(n)}) = ck(\|\mathbf{x}_t - \mathbf{s}_t\|)$, where c is a normalization constant which makes the integral $K(\mathbf{x}_t - \mathbf{s}_t^{(n)})$ to one, and $k(r) = k(\|\mathbf{x}_t - \mathbf{s}_t\|)$ is called the profile of the kernel K. For example, the Epanechnikov kernel is:

$$K_E(\mathbf{x}) = \begin{cases} \frac{1}{2}c_d^{-1}(d+2)(1-\|\mathbf{x}\|^2) & 0 \leq \|\mathbf{x}\| \leq 1 \\ 0 & \|\mathbf{x}\| > 1 \end{cases} \tag{4.11}$$

Given a particle set $\mathbf{S}_t$ and the associated weights $\{\pi_t^{(n)}\}_{n=1}^N$, the particle mean is determined by

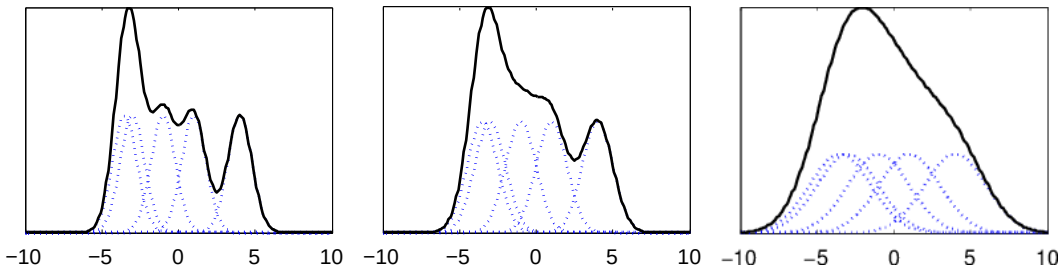


Figure 4.4: Approximation of a PDF using Gaussian kernels with increasing bandwidth.

$$m(\mathbf{s}_t^{(n)}) = \frac{\sum_{i=1}^{N} H_h(\mathbf{s}_t^{(n)} - \mathbf{s}_t^{(i)})\pi_t^{(i)}\mathbf{s}_t^{(i)}}{\sum_{i=1}^{N} H_h(\mathbf{s}_t^{(n)} - \mathbf{s}_t^{(i)})\pi_t^{(i)}} \tag{4.12}$$

where $h(r) = -k'(r)$ is in turn a profile of kernel H_h. It can be shown that the mean shift vector $m(\mathbf{x}) - \mathbf{x}$ always points toward the steepest ascent direction of the density function (Comaniciu and Meer, 2002; Chang and Ansari, 2005).

Following the shifting of particles using the mean shift vector, the particle weights $w_t^{(n)}$ are recomputed. As the shifting of the particles implies that the new particles are not distributed according to the posterior distribution, a reweighting is performed to guarantee that each mean shift iteration follows the correct posterior gradient. Using subscript j to denote the particle set after the jth mean shift iteration at time t, the weight is recomputed based on the posterior density evaluated at the new particle positions $\mathbf{s}_{t,j}^{(i)}$ and a particle density balancing factor (Chang and Ansari, 2005):

$$\pi_{t,j}^{(n)} = \frac{p(\mathbf{s}_{t,j}^{(n)} \mid \mathbf{Y}_t)}{q_{t,j}(\mathbf{s}_{t,j}^{(n)})} \tag{4.13}$$

The balancing factor in the form of the denominator is the new proposal density:

$$q_{t,j}(\mathbf{x}_t) = \sum_{l=1}^{N} K_h(\mathbf{x}_t - \mathbf{s}_{t,j}^{(l)}) \tag{4.14}$$

Without the balancing factor, several particles concentrating after a mean shift iteration at one density mode would result in a high kernel density estimation at this

position and would, therefore, heavily influence the particle mean of Eq. 4.12. This effect is avoided by the reweighting of Eq. 4.13 that incorporates in the posterior density evaluation how many particles are located in the kernel window around the new particle position.

The overall posterior density uses a sample-based approximation of the prior density and is given by

$$p(\mathbf{x}_t \mid \mathbf{Y}_t) \propto p(\mathbf{y}_t \mid \mathbf{x}_t) \sum_{l=1}^{N} p(\mathbf{x}_t \mid \mathbf{s}_{t-1}^{(l)}) \pi_{t-1}^{(l)} \tag{4.15}$$

Equation 4.15 is the Kernel particle filtering equivalent of standard recursive Bayesian filtering (see Eq. 4.7).

The choice of the kernel bandwidth h is of crucial importance in kernel based density estimation as it defines over which range the search for modes is carried out. As depicted in the three examples in Fig. 4.4, a small value can generate a very ragged density approximation with many peaks, while a large value can produce an over-smoothed density estimate. In particular, if the bandwidth of the kernel is too large, significant features of the distribution like, e.g., multiple modes can be missed. The bandwidth is usually scaled down at each mean shift iteration in order to concentrate the particle set on the most dominant modes.

Such a mode-based representation of a probability density function is still a probabilistic representation and allows to track, e.g., a hand in a probabilistic way. If a single result is required for some subsequent process like, e.g., gesture recognition, the most dominant mode has to be extracted from the mode-based representation. Depending on the type of application, this can be the weighted mean of an appropriately chosen subset of all particles or some other evaluation of the representation.

4.3 Adaptive Visual Features for Hand Tracking

With the algorithmic basis of tracking algorithms introduced in the previous Section, we can now turn to their application for hand tracking. Instead of using fixed representations of the hand for detecting it in individual images as outlined in Section 3.2, the hand representation in the form of visual features can be updated continuously in every image to account for changing visual features of the hand. Figure 4.5 depicts the processing steps of such an adaptive approach to hand detection and tracking. In this Section, typical methods for updating visual features like color, shape, and appearance will be outlined.

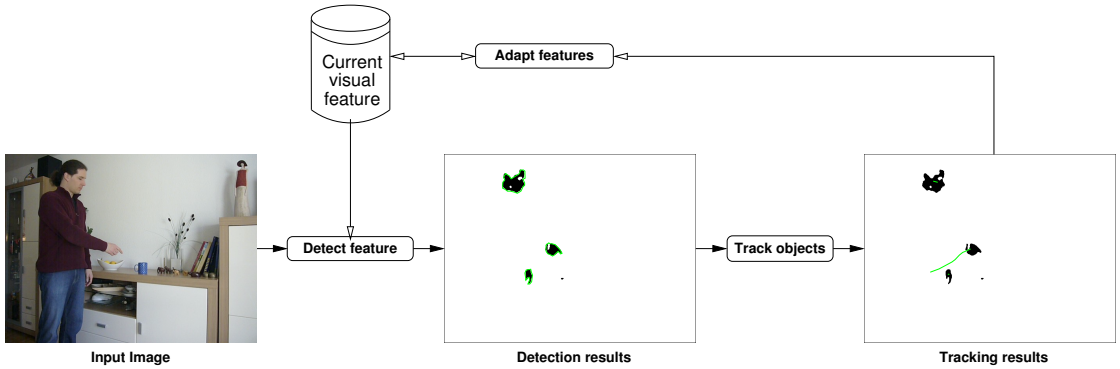


Figure 4.5: Iterative updating of visual features based on successful tracking of detected hands.

4.3.1 Adaptive Hand Color

Many approaches to detect skin-colored image areas (see also Section 3.2.2) are trained on large datasets and operate on isolated images with correct pixel classification rates around 90% (see, e.g., Phung et al., 2005; Kakumanu et al., 2007). However, the training images are usually photographs and therefore of an extraordinary high quality compared to image data acquired in natural human-robot interaction. In other words, during a hand gesture the images acquired by a standard video camera may exhibit strong shades and insufficient lighting compared to the high-quality photographic images typically used in skin color databases. Consequently, classifiers have to be trained and evaluated on data obtained in real interaction situations to capture the challenges encountered in such settings. Due to the broad range of possible lighting variations, however, the classification quality usually decreases heavily in such situations.

One way of taking advantage of the sequential nature of the individual images to be classified is to *adapt* a temporary color model that covers only the subset of the color space representing the current skin color. After every segmentation step, the color of the detected hand is used to update this temporary color model. An alternative would be algorithms that allow to *tolerate* arbitrary lighting changes, but such color constancy algorithms are still not applicable in real-world domains as pointed out in Section 2.3.1.

An adaptive algorithm needs to decide for every image whether the it contains objects of interest that have changed since the last frame and to whose changed appearance the algorithm should adapt. In other words, the question that needs to be answered is: 'What parts of this image belong to previously observed objects that are now illuminated differently and therefore have a different appearance?'.

A major challenge in the actual implementation of adaptive strategies is the lack of 'ground truth' in the color signal: it is impossible to decide based only on iconic information, i.e., the color of an individual pixel, whether a pixel still shows the same object that now exhibits a different appearance. Consequently, an adaptive algorithm may adapt the color models to 'wrong' values. For example, if a tracked human hand moves over a wooden desk with a color slightly different from skin color, the desk could be accidentally segmented to belong to the hand region. Subsequently, the color model may be updated not only with pixels belonging to the hand but also with pixels exhibiting wood color. Consequently, the hand will not be detected correctly in subsequent images and tracking will eventually fail. By using a physics-based model of the skin reflectance properties - the skin locus (see Section 3.2.2 on page 56) - as a filtering step in the update process it is possible to discard pixels not having a skin-like color (see, e.g., Soriano et al., 2000; Fritsch et al., 2002; Tsai et al., 2008). Such an additional filtering process based on general properties of skin color can reduce the risk of a wrong adaptation. An alternative way of coping with the potentially wrong update of the skin color model is the use of a probabilistic tracking framework that maintains several tracking hypotheses including associated color models (Shan et al., 2007).

Going beyond a pure adaptation to the current appearance of the hand, Sigal et al. (2000) have applied a predictive adaptation of the color model by explicitly modeling the temporal variations. In this way, an improved segmentation result is expected by anticipating how the skin color will be observed in the next image. However, as in human-robot interaction the lighting conditions are rather unpredictable, this kind of adaptation is of limited benefit.

Besides applying restrictions on the color cue itself, it is also beneficial to incorporate domain-dependent context knowledge for updating the skin color model. In a scenario where a mobile robot is intended to track a gesturing human, the appearance of the face and the motion of the hands can be used to validate the adaptation. For example, if the complete human is observable in the camera image, the appearance of the face can be used as additional information for the skin color model. After a segmentation of the current image with the current skin color model, a face detection algorithm can be used to test whether a segmented skin-colored region actually represents a face (Fritsch et al., 2002). If a face is detected at the position of the skin-colored region, an image patch of face size can be used for adapting the skin color model.

On a more elementary level, the context that pixels representing a hand have not only skin-like color but, if the hand is moving, are also subject to image motion can be used. Incorporating motion information into the pixel classification process

allows to discard pixels that have skin-like color but are static. Obviously, a simple combination of these two cues would also result in not finding a hand that rests still in the image. Choosing an appropriate combination scheme is therefore of great importance in order not to discard too many hand pixels. Lömker and Sagerer (2002) present an approach where the skin and motion information are both represented by likelihoods indicating their quality. Using an additive combination, a pixel gets high values if it is very skin-like or it exhibits strong motion or both. Through choosing an appropriate threshold, resting hands can be correctly segmented, too.

4.3.2 Adaptive Hand Shape

Similar to the color-based hand detection approaches also the shape-based hand detection approaches outlined in Section 3.2.1 assume that the hand has a fixed shape and can be extracted from an isolated image. As pointed out, this assumption is often not valid as the human hand is very flexible and in most environments there is clutter in the background prohibiting the use of simple contour extraction techniques.

To cope with these challenges, iterative approaches detect and track the hand by adapting to its changing shape. Following an initialization that is carried out with the techniques for hand detection in single images, the tracking needs to deal with small changes in the shape only, as depicted in a simplified example in Fig. 4.6. Obviously, restricting the possible shape deformations that can occur within one time step of an image sequence is computationally more efficient than searching for an arbitrary shape.

A variety of shape representations have been proposed in the literature for modeling object boundaries. Among these are Active Shape Models (ASM) proposed by Cootes et al. (1995) that have also been applied to hand gesture recognition

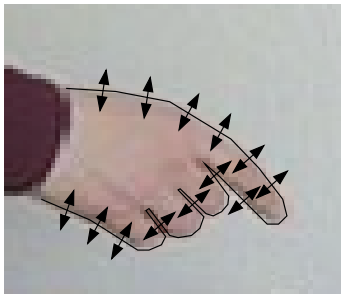


Figure 4.6: Simplified example of an intermediate hand shape model allowing for limited adaptation of the shape between two successive time steps.

(Ahmad et al., 1997). Here, a shape is defined by a number of control points and the variations of the control points are learned from training data. Knowing the hand position and its configuration from the last time step, the previous hand shape model can be fitted in a top-down manner to the current image by moving the control points within their allowed/learned range. For applications where the hand performs only a limited set of gestures, i.e., the hand shapes are limited, a model can be learned for each different configuration and the variability of the control points must account for the intermediate configurations between two shapes. During tracking, the different shape models must then be matched to the image data and the best match represents the recognition result. Obviously, this is only feasible if the set of hand configurations is small and if the model variability can cope with all intermediate hand configurations occurring in the image data.

In order to overcome these drawbacks, the probabilistic modeling of shapes has gained more interest in the last decade. The introduction of particle filtering (see Section 4.2.2) in the form of the CONDENSATION algorithm for performing contour tracking by Isard and Blake (1996) resulted in a more intensive research on probabilistic contour tracking. The important difference to standard tracking approaches is the probabilistic nature of the tracking algorithm that allows to maintain multiple shape hypotheses during the analysis. In this way, it is possible to deal with ambiguous situations occurring frequently in image sequences and, therefore, enable more robust tracking. Consequently, it has been applied often for hand shape tracking (see, e.g., MacCormick and Isard, 2000; Laptev and Lindeberg, 2001). Due to the inherent 2D nature of the hand shape, however, this approach is primarily suited for gestures that are characterized by their shape. This holds true for command gestures and pointing gestures which are recognizable from the 2D shape if the camera viewpoint is chosen adequately while the method is less appropriate for tracking hands performing manipulative gestures.

4.3.3 Adaptive Hand Appearance

An obvious alternative to modeling the hand by its contour is to incorporate the complete appearance. One of the first approaches to adaptively model the hand appearance is the work by Heap and Hogg (1996). Here, a deformable 3D model of a hand is created from a large set of training data (see Fig. 4.7) and the model points distributed over the hand surface are adaptively matched to new input images.

Later approaches have used a variety of 2D and 3D models for capturing the appearance of a hand over time. For example, Bray et al. (2004) have used an underlying 3D hand skeleton model to derive the changing 2D skin surface appear-

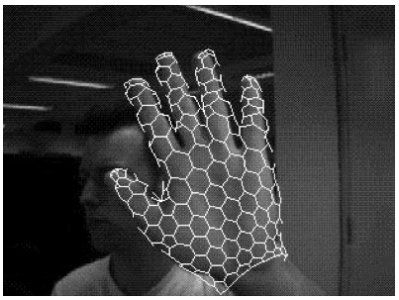


Figure 4.7: A deformable 3D model of a hand (image from Heap and Hogg, 1996).

ance and match it to the image data. This resulted in an adapted 3D configuration that was used for matching in the next time step. Such sophisticated 3D modeling approaches have a high computational load, which makes the use of 2D models more attractive.

An adaptive approach for modeling the 2D appearance that goes beyond a simple color adaptation (see Section 4.3.1) by using "flocks of features" was proposed by Kölsch and Turk (2004). Instead of modeling the color of the complete hand, the appearance of small image patches is modeled. These patches are tracked in an image sequence using optical flow and - at the same time - the appearance model of each patch has to fulfill constraints with respect to the other patches. The latter point is the "flocking" behavior as analogy to a flock of birds: Each feature patch has a minimum distance to other features but a maximum distance to the mean of all feature patches. Due to the used 2D models this approach is real-time capable and at the same time the concept of "flocking" results in a good tracking performance in unconstrained environments and with a moving camera.

An integrated method for modeling hands combining shape and appearance has resulted from extending the work on Active Shape Models for contour tracking by including appearance information. This has resulted in an algorithm termed Active Appearance Model (Cootes et al., 2001) that has been applied intensively to face detection and also to hand detection (Roussos et al., 2010).

As the appearance-based approaches incorporate implicitly the color-based features and also partially the shape-based features, they provide the most complete modeling of the visual features of hands in 2D. This is ideal for tracking the hand in 2D, but depending on how the appearance is modeled, the representation may not provide any details about the exact hand configuration. If such a low-level modeling is chosen (e.g., the approach by Kölsch and Turk), this is mainly feasible for tracking gestures that are characterized by their 2D motion, but for obtaining the detailed hand configuration additional processing steps are required.

4.4 Example: Detecting and Tracking Hands Based on Skin Color

In order to give a practical example of hand tracking incorporating adaptation, this Section will cover an approach for detecting and tracking hands based on skin color. As pointed out in Section 3.2.2, the color of human skin is a suitable cue for feature extraction in arbitrary domains. It is independent of the specific hand configuration and invariant to rotation and scaling of the hand. Furthermore, it is independent of the motion of the hand as well as any motions in the background. However, the variations in lighting conditions pose major challenges to a color-based feature extraction. Even in a room without windows, different lighting conditions are encountered at different positions in the room. This is due to the individual light sources at the ceiling, and the shading introduced by objects or moving persons. In order to compensate the influences from such lighting and shading effects, a dynamic color model has to be used that is constantly adapted to the changing color of the object of interest, i.e., the gesturing hand. Using such an adaptation allows to track hands in a wide range of environments with varying lighting conditions.

The purpose of this Section is to present an example of an adaptive skin-color segmentation algorithm developed for tracking gesturing hands. The overall processing scheme is explained in the next Subsection before further Subsections explain the processing steps in more detail. The performance is demonstrated qualitatively with snapshots from image sequences containing varying lighting conditions.

4.4.1 System Overview

The processing steps of the example approach for segmenting input images using an adaptive skin color model are as follows (see Fig. 4.8):

Based on a domain-dependent initialization step (1), an initial skin color model (2) is generated. The skin color distribution is represented as Gaussian mixture model in the normalized r-g color space (see Section 4.4.2 for details). From now on, every image is processed with the current skin color model to label every pixel as either skin or non-skin pixel. For this purpose, the input image is first transformed into the r-g color space. Based on the Gaussian mixture model of the skin color, the probability of every pixel for being skin color is calculated. A classification threshold (3) is applied to the probability values to obtain a binary label image (see Section 4.4.3 for details). An example label image is visible in Fig. 4.8 on the left beneath the input image. This label image is smoothed through applying a median filter (4) to eliminate spurious pixels with skin color. By carrying out a connected

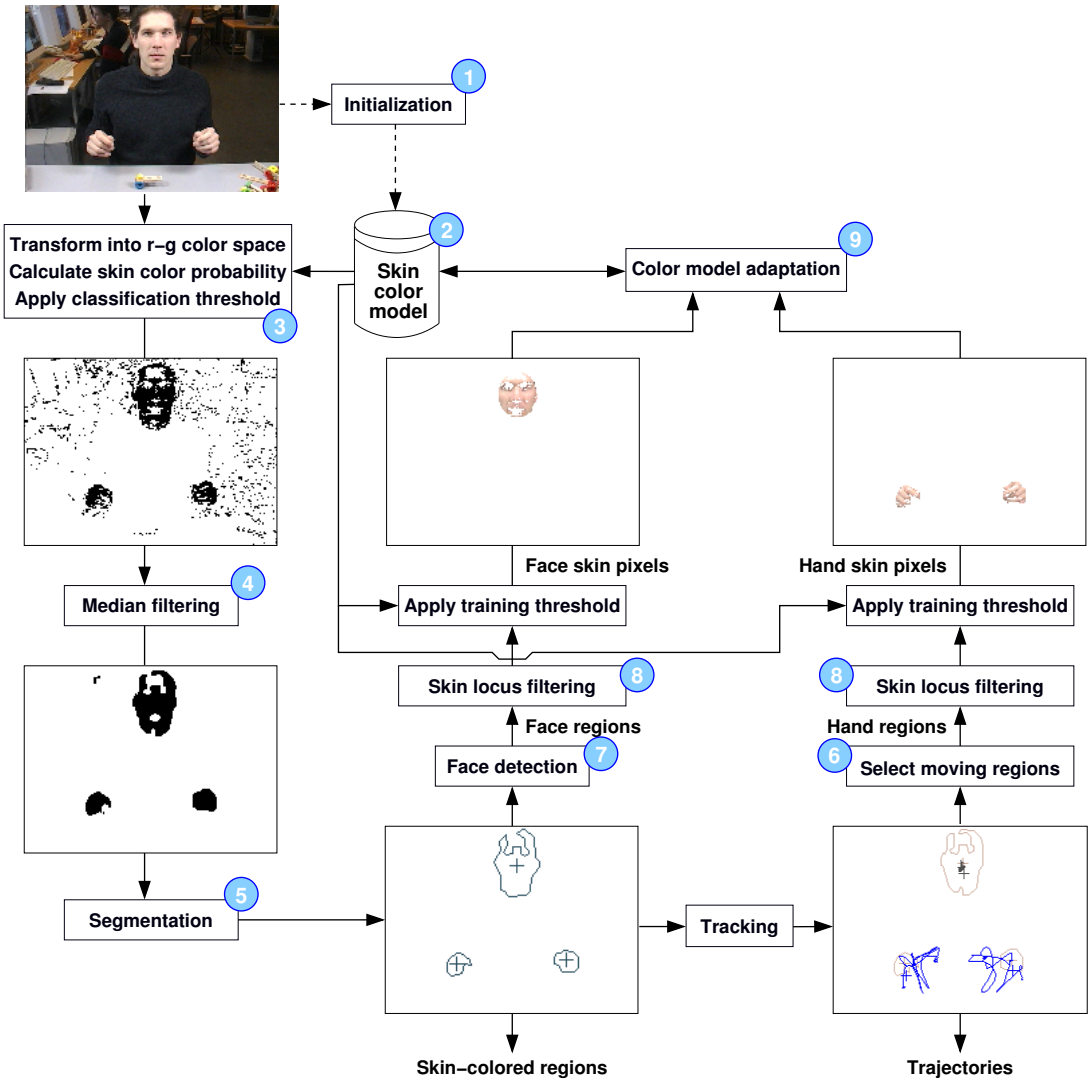


Figure 4.8: Image processing steps for performing adaptive skin color segmentation (image from Fritsch, 2003).

components analysis (5) on the smoothed label image, skin-colored regions can be extracted. These skin-colored regions represent image-specific information about the objects in the scene having skin color.

To select the skin-colored regions that can be used for updating the skin color model, the segmented regions are analyzed and only regions exhibiting either motion (6) or a face-like structure (7) are kept. Based on a pre-trained global skin locus (see Section 3.2.2), all pixels in the update regions that are not skin-like are discarded (8). For the remaining skin-like pixels lying inside the skin locus, the

skin probability is calculated using the current skin color model. This probability is compared to a threshold that was acquired in the previous skin color model training phase. All pixels with a probability value exceeding this training threshold are used for adapting the skin color model (9). The updated skin color model (2) is used for segmenting the next input image (see Section 4.4.4 for details).

With the described steps, this adaptive skin-color segmentation

1. imposes a minimal set of restrictions on the scenario.
2. enables skin color segmentation in natural settings with changing lighting conditions
3. provides a polygonal region description for all skin-colored regions instead of only the center of mass of a single skin-colored region.

A detailed description of the individual processing steps for performing adaptive skin color segmentation is the content of the next Subsections.

4.4.2 Modeling Skin Color Distribution and Skin Locus

Based on the properties of skin color described in Section 3.2.2, the normalized r-g color space is used for representing skin color. The empirical study by Yang et al. (1998) suggests that a Gaussian mixture is well-suited to model skin color distributions. Gaussian models are used very often in adaptive color segmentation approaches (Wren et al., 1997; Oliver et al., 2000; Raja et al., 1998) as they allow to model color distributions with a small parameter set. Consequently, Gaussians are applied here to model skin color distributions. The skin likelihood for a pixel with color value $\mathbf{x} = (r, g)$ can be calculated for a Gaussian $G(i)$ with mean $\boldsymbol{\mu}_i$ and covariance $\boldsymbol{\Sigma}_i$ using:

$$p_i(\mathbf{x}) = \frac{1}{\sqrt{2\pi \det \boldsymbol{\Sigma}_i}} \exp\left\{-\frac{1}{2}[\mathbf{x} - \boldsymbol{\mu}_i]^T \boldsymbol{\Sigma}_i^{-1} [\mathbf{x} - \boldsymbol{\mu}_i]\right\} \tag{4.16}$$

The remaining design decision is whether an *unimodal Gaussian* or a *mixture of Gaussians* should be used to model a skin color distribution. Obviously, this depends on the properties of the distribution, i.e., on the variations in lighting and skin types contained in the distribution.

While modeling a skin color distribution with a single Gaussian is computationally less expensive, it results in a reduced ability to deal with color inhomogeneities. Color inhomogeneities are encountered within a single skin-colored object (intra-region) due to, e.g., partial shading as well as between objects (inter-region) due

to, for example, different skin types or lighting conditions. An important situation where a skin-colored hand or face exhibits large intra-region color variations are partial shading effects. Although the r-g color space partially accounts for intensity differences, a good segmentation result can only be achieved through modeling the skin color distribution with a mixture of Gaussians.

The quality of modeling several skin colored regions in an image with one mixture of Gaussians usually increases with the number of mixture components used. The mixture can be expected to allow for a better modeling than the unimodal Gaussians if the number of components M is at least equal to the number of skin-colored regions in the image and the different regions have partially a similar color. For a single human, the potential differences between the hands and the face will be related primarily to different lighting conditions. Consequently, with a smaller inter-region variation due to similar skin appearance of the hands and the face of a single human, a mixture of three Gaussians should allow to also model intra-region variations due to shading. More important, however, is the more precise modeling of the skin color distribution that reduces the number of false positives in the background.

Having a model for the skin color distribution, the next question is how to estimate and update this model. As pointed out in Section 3.2.2 on page 57, the skin color of a large number of different human subjects is distributed in normalized color space in a restricted area, the skin locus. This skin locus can be used to ensure that only skin-like pixels are used for adapting a skin color model. The exact shape and placement of the locus depends on the camera characteristics and on the lighting conditions used for acquiring the training images. To generate the skin locus for a specific setup, images containing skin patches under different lighting conditions have to be collected. Figure 4.9 shows an exemplary skin color distribution for the recording setup used in this example approach. Following Soriano et al. (2000), two quadratic functions have been fitted to this distribution to obtain the setup-specific skin locus.

The parameters of the two quadratic functions that enclose 95% of the pixels in the empirically determined skin color distribution are:

$$A_u = -5.05 \quad b_u = 3.71 \quad c_u = -0.32 \tag{4.17}$$

$$A_d = -0.65 \quad b_d = 0.05 \quad c_d = 0.36 \tag{4.18}$$

The decision whether a specific pixel $\mathbf{x} = (r, g)$ is contained in the skin locus is now readily available by calculating the values of the two quadratic functions based on the r value

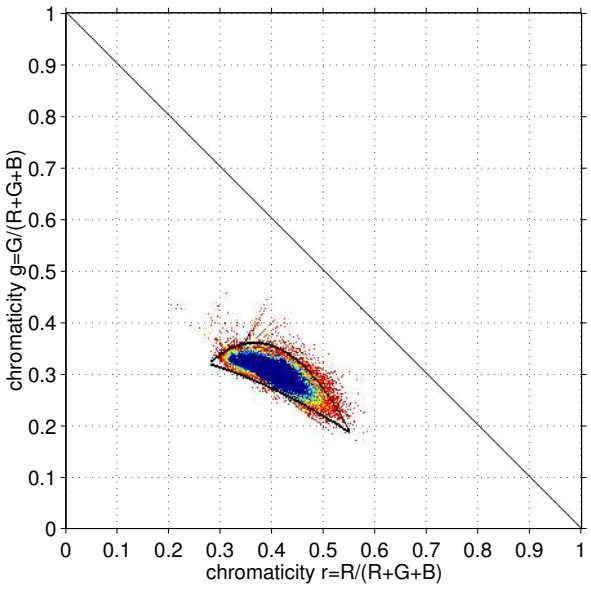


Figure 4.9: Histogram of the global skin color distribution with the skin locus fitted to the distribution. The color indicates the number of samples for a given position - from red (few samples) to blue (many samples).

$$F_u = A_u r^2 + b_u r + c_u \tag{4.19}$$

$$F_d = A_d r^2 + b_d r + c_d \tag{4.20}$$

and checking for the g value to lie between the two calculated function values:

$$\text{PixelInSkinlocus}(r,g) = \begin{cases} 1 & \text{, if } (g < F_u) \ \wedge \ (g > F_d) \\ 0 & \text{, else} \end{cases} \tag{4.21}$$

With this simple classification rule, a preprocessing step to discard all pixels with colors that are not contained in the training set can be realized before constructing the Gaussian mixture model.

One remaining problem is the fact that the skin locus contains also the color values for white colors resulting from highlights in the skin-colored areas of the training images, i.e., reflections of the illumination from the light sources. Often a scene also contains several artificial objects with a white color like, e.g., a door, a table, or a computer display. To avoid including the color white in the skin locus, a small region around the *white point* in r-g color space at $(0.33, 0.33)$ is therefore excluded from the skin locus.

4.4.3 Applying Skin Color Segmentation

For segmentation of the input image, the normalized r-g color values are calculated from the RGB color values. Subsequently, the skin likelihood is calculated for each individual pixel based on its color and the current color model. The overall skin probability value for the pixel x is determined by adding up the M individual Gaussian mixture components (see Eq. 4.16) with their weights c_i:

$$p(\mathbf{x}) = \sum_{i=1}^{M} c_i \; p_i(\mathbf{x}) \tag{4.22}$$

Obtaining for every pixel its skin likelihood $p(\mathbf{x})$ results in a skin probability image for the complete input image. This probability image is binarized using a classification threshold S_{class} to obtain a label image containing skin and non-skin pixels (see Fig. 4.8 on page 82).

The value of the threshold S_{class} has to be chosen carefully, a low threshold will potentially classify a larger number of non-skin pixels as skin-like color (false positives) while a high threshold may not classify all skin-colored pixels correctly (false negatives) due to inhomogeneities caused by, for example, shades. The probability values $p(\mathbf{x})$ that are encountered in the probability image are not normalized as the variances of the individual mixture components influence the range of probability values that is occupied by skin pixels. It is therefore advantageous to use an adaptive threshold that is based on the actual probability values present in the training set. To obtain the classification threshold S_{class}, the training pixels used for constructing the Gaussian model are also used as test set for building a histogram of the probability values generated by the Gaussian model.

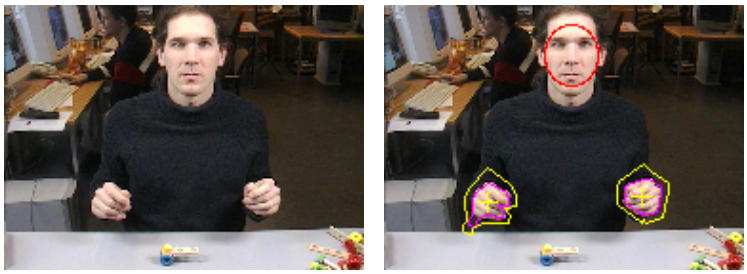
The probability value histogram is analyzed starting at the bin representing the largest probability value p($\mathbf{x}$). The bin counts are successively added up until the sum contains more than 98.5 % of all N_{train} training pixels (see Eq. 4.23). Setting the threshold to classify a fraction of 98.5 % of the training pixels correctly has been determined empirically to ignore spurious outliers contained within the last 1.5 % of the training pixels. The probability value represented by the last histogram bin added is taken as the threshold S_{class} for skin color classification, i.e., all probability values below this threshold are classified as non-skin.

$$Pr(Y > S_{class}) = 0.985 * N_{train}, \qquad Y = p(\mathbf{x}) \tag{4.23}$$

To remove isolated pixels classified as skin and provide a more homogeneous result, a median filter of size 5×5 is applied to smooth the label image. Next, a connected components analysis is carried out in the segmentation step to obtain

the region segmentation result (see Fig. 4.8 on page 82). Subsequently, the feature extraction step calculates polygonal descriptions of the image regions and region features like, e.g., compactness, pixel size, center of mass.

Figure 4.10(a) depicts an example image where a successful face detection (see red ellipse in Fig. 4.10(b)) is used for initializing a Gaussian mixture model (Fritsch et al., 2002). Based on this model, the hands can be successfully segmented as visualized with the purple polygons around the hands.



(a) (b)

Figure 4.10: Initializing skin segmentation. a) input image; b) image with overlaid face detection result (red ellipse), the resulting hand segmentation (purple polygon) and the image area for updating the skin color model for the next time step (yellow).

Figure 4.11 depicts the Gaussian mixture model, the resulting label image, and the final binary segmentation result after median filtering.

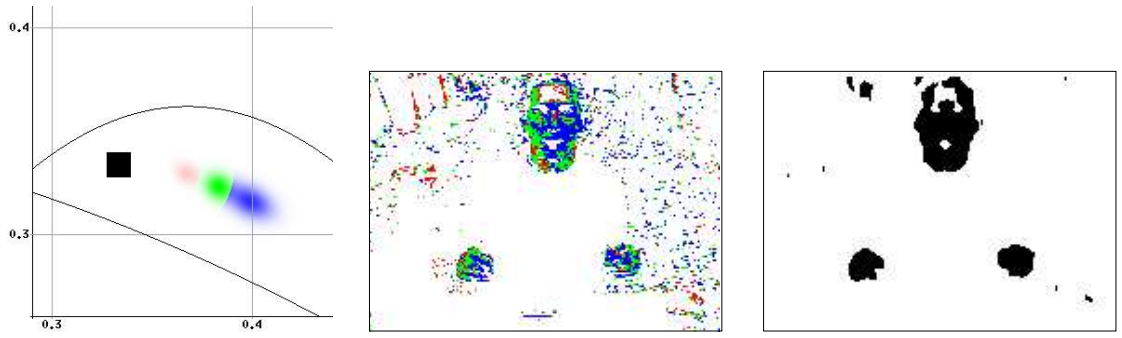


Figure 4.11: Example of pixel classification using a mixture of Gaussians: (a) the three mixture components in r-g color space; (b) the label image with the color indicating the mixture component having the highest probability; (c) the label image after median filtering.

The segmentation scheme described above operates on the pixel level to construct the label image. Applying a threshold to the individual probability values ignores spatial information that may have been present in the overall probability image. For example, a hand results in an area of high probability in the probability image. This spatial relation between the iconic probability values can be used for a segmentation scheme that focuses on finding elliptical regions representing human hands and faces (see, e.g., von Hardenberg and Berard, 2001; Kruppa et al., 2002). In such a scheme, a pixel that lies in an area with many pixels having high probability values and forming an elliptical shape is assigned the 'skin' label even if its probability value is below the threshold. In this way, the influence of partial shadings or non-skin objects (e.g., a ring on a finger, glasses) can be reduced and the form of the segmented region is closer to the 'expected' elliptical form. However, the associated computational cost of fitting ellipses of arbitrary size and orientation to the probability image is very high. Only in domains where the ellipses can be restricted, for example to 'face' ellipses with a fixed orientation and size, this processing scheme can be applied.

4.4.4 Updating the Skin Color Model

To realize an adaptive skin color segmentation, the skin color model generated in the initialization step has to be adapted to the current appearance of the skin-colored object. As there is no ground truth available on the exact shape of the object, heuristic rules have to suffice. To select the skin-colored pixels that can be used for updating the skin color model, two different types of context knowledge for determining the 'true' skin areas are applied:

- If a region $R(i)_{segment}$ exhibits motions, i.e., it is not a skin-colored background object, it is considered an 'interesting' region and used for updating. Whether a region is a background object or not can be determined if this region is tracked over time.

- If a region $R(i)_{segment}$ segmented in an individual image exhibits a face-like structure, an elliptical update region R_{update} is constructed based on the size of the face as determined by the face detection. If this face region is also an 'interesting' region due to its motions, the elliptical update region replaces the original update region.

In case of a perfect segmentation, all pixels belonging to a skin-colored object would have been segmented correctly despite of a changed appearance. In this

case, the segmented object region $R(i)_{segment}$ and the image region for updating $R(i)_{update}$ would be identical, i.e., all segmented pixels could be used to update the skin color model.

Under realistic circumstances, however, the appearance variations within a skin-colored area are often so large that not all pixels are correctly segmented. Consequently, the segmented region $R(i)_{segment}$ represents only those pixels that are sufficiently close to the current skin color model but may miss some pixels that already have a too different color due to, e.g., shading. Therefore, it is advantageous to construct a larger update region $R(i)_{update}$ under the assumption that it contains most or all of the pixels belonging to the object due to spatial closeness.

For skin-colored regions that exhibit motion, the update region $R(i)_{update}$ is generated by enlarging the segmented region $R(i)_{segment}$. If the shape of the object is known, e.g., a hand with fingers stretched out, this shape model could be matched to the segmented region to construct $R(i)_{update}$. For all skin-colored objects with unknown or flexible shape, such a priori knowledge is not available and therefore the polygon describing $R(i)_{segment}$ is stretched to describe a region with an area two times the size of the segmented region. This enlarged region forms the update region $R(i)_{update}$. For skin-colored face regions, the elliptical update region is used as determined by the face detection.

All pixels within $R(i)_{update}$ that do not lie inside the skin locus (Eq. 4.18-4.21) are discarded. This step removes all pixels that are contained in the update area but have no skin-like color like, e.g., rings, or parts of the background resulting from the update area stretching. However, pixels from image areas having a color similar to skin, e.g., a wooden desk, are not removed by skin locus filtering. To avoid adapting the color model to background areas with skin-like color, a training threshold S_{train} is used to select only those pixels that exhibit a skin color close to the current skin color model. Similar to the classification threshold S_{class} (see Eq. 4.23), the training threshold is calculated from the probability value histogram of the previous training set. The threshold value is chosen such that the probability values $\mathrm{p}(\mathbf{x})$ of all N_{train} training pixels from the previous update step are above the threshold:

$$Pr(Y > S_{train}) = N_{train}, \qquad Y = p(\mathbf{x}) \tag{4.24}$$

Only those pixels in the current update area that have a skin probability above the training threshold S_{train} are considered for updating the skin color model. This enforces a smooth adaptation of the skin color model and has been found to allow coping with hands and faces in front of wooden furniture and other skin-like background objects. After applying the skin locus filtering and the training threshold

to remove all pixels that are not skin-like, the remaining training pixels contained in $R(i)_{update}$ are used for updating the Gaussian mixture model.

For updating the mixture of Gaussians, the training pixels of all update regions $R(i)_{update}$ are collected. Using this training set, the parameters of the $M = 3$ individual components of the Gaussian mixture are calculated using a *k-means* clustering algorithm. Clustering is done by choosing an initial set of cluster centers based on the first M color values. Subsequently, the remaining color values are processed one-by-one to update the initial cluster centers. For every training pixel $\mathbf{x}_{training,j}$, the following three steps are performed:

1. assign the pixel $\mathbf{x}_{training,j}$ to the closest cluster center.
2. calculate the new center of mass for this cluster.
3. update the cluster center with the calculated center of mass.

After processing all color values, each final cluster is used to compute the mean and variance of one Gaussian mixture component.

4.4.5 Evaluation Results

The described adaptive segmentation approach can find hands (and also faces) in image sequences with varying lighting conditions. The segmentation accuracy is presented qualitatively as the effort of a quantitative evaluation does not match its limited expressiveness: the often inferior lighting conditions in typical image sequences and the adaptive nature of the proposed processing scheme would require the evaluation of a large number of 'representative' image sequences to obtain a well-founded quantitative result. Even such a quantitative result of a system approach is not comparable to the evaluations performed on typical photographs as the image quality encountered in automatically acquired image sequences differs substantially from the data sets used in non-interactive systems (see, e.g., Yang et al., 1998; Jones and Rehg, 1999).

Moreover, the processing of image sequences for the extraction of motion trajectories in order to enable gesture recognition favors small processing times to obtain a high frame rate over exact segmentation results.

The dynamic nature of image sequences makes the presentation of results difficult, as only 'snapshots' of the performance of the adaptive skin color segmentation can be shown. An example for the segmentation quality is given in Fig. 4.12 for a scene depicting a human drinking a cup of coffee in a typical office setting. Although the

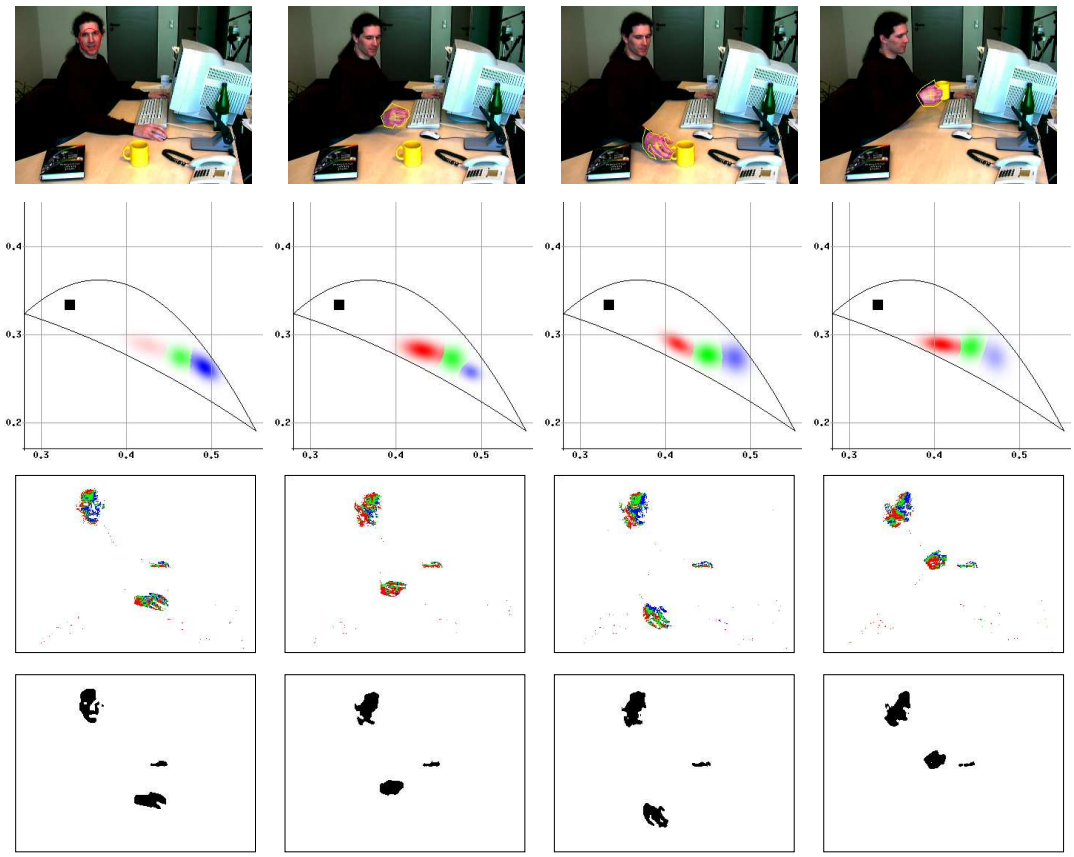


Figure 4.12: Exemplary segmentation results for a scene depicting a person drinking a cup of coffee.

human hand is not completely labeled as skin in the smoothed label image (bottom row), the algorithm is capable of keeping track of the hand and the variations in the skin color distribution of the hand during the drinking action. Despite of the color of the wooden desk looking similar to skin, the adaptation of the skin color model is not distracted.

Besides the use of the described adaptive skin color segmentation for tracking gesturing hands of a human, it can also be applied for other tasks like, e.g., the tracking of persons moving in an office environment from onboard a mobile robot (Fritsch et al., 2003). Having the information of a human in its surrounding is the precondition for a mobile robot to apply then another skin segmentation focussing on the detection and tracking of the gesturing hands of this human.

4.5 Model-based Approaches to Hand Tracking

Similar to realizing the hand tracking by continuously adapting the visual features used for detection, the model-based detection approaches outlined in Section 3.3 can be used adaptively, too. Note that several different 3D hand or body configurations will result in similar 2D shapes and, consequently, the dynamics modeled for a 2D shape have to account for these different sources of the same 2D appearance. To avoid this ambiguity, research approaches on model-based tracking have shifted away from bottom-up matching of planar 2D shape representations as outlined in Section 4.3.2. Instead of adapting a 2D visual shape representation directly, an internal 3D representation is tracked over time and used for generating in a top-down manner the 2D data to be matched to the image data. This can be done either for individual hands with a detailed hand model (see Section 4.5.1) or using a complete body model with less detail for the individual parts (see Section 4.5.2).

It should be noted that also dense depth measurements as delivered by, e.g., the Kinect sensor (see Section 2.2.2 on page 35) could be used for 3D body model tracking. However, in this Section the focus is on vision-based approaches to model-based tracking.

4.5.1 Model-based Tracking of Hand Configurations

In the 2D model-based hand detection approaches (see also Section 3.3) a palm and five fingers are mapped to the image data. Especially for tracking it becomes possible to apply a 3D hand model as the continuous tracking reduces the search space drastically. In other words, not all possible palm and finger configurations are likely, but only those that can directly follow from the configuration in the previous image. Due to the high degrees of freedom of the articulated hand, there are several such configurations possible. This ambiguity needs to be tackled in the model matching where often probabilistic methods are applied. Figure 4.13 depicts the general scheme of such a model-based approach to hand tracking assuming for simplicity a deterministic matching.

For example, Stenger et al. (2001) use a 3D hand model with 27 degrees of freedom that is projected to the image plane to provide a 2D contour. In their approach, a variant of Kalman filtering is used to track the hand pose. As they do not apply a probabilistic approach, the tracking experiments in their publication focus on translation and rotation of a specific hand shape and only allow for a single configuration change - thumb up or not.

A similar modeling of the hand is used by Wu et al. (2005) to generate a projection

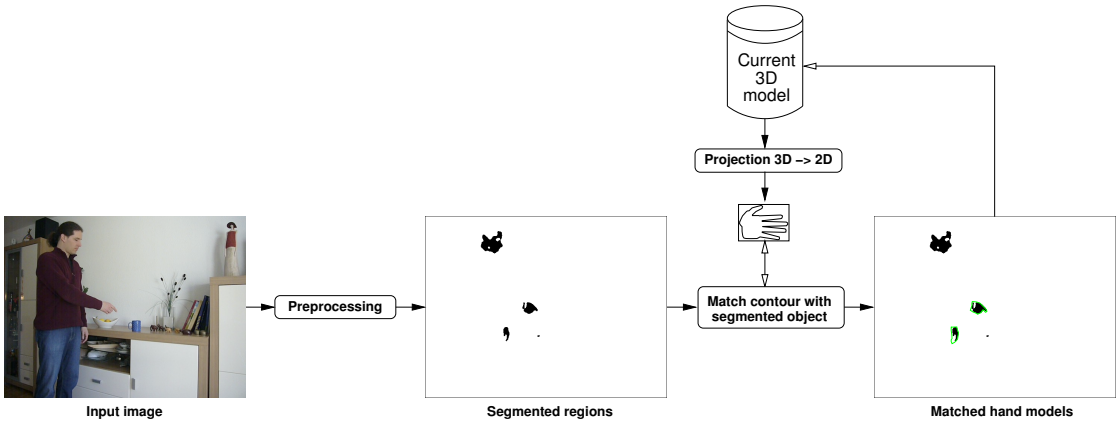


Figure 4.13: Iterative updating of the hand model based on successful tracking of detected hand configurations.

onto the image plane. In order to perform probabilistic tracking of good matches, a particle filtering algorithm (see Section 4.2.2) is applied for processing the ambiguous hand configurations. The 2D projections of the different possible hand configurations maintained by the particle filter are individually evaluated based on comparing the 2D contour with edge measurements. In addition to the evidence for the individual configurations from the image data, their approach incorporates priors learned from typical hand configurations and from typical configuration changes to improve the tracking quality.

A more complex 3D deformable hand model consisting of a polygonal surface with underlying skeleton is proposed by Bray et al. (2004) for recovering the hand pose. The algorithm makes use of a structured light sensor (see Section 2.2.2) providing 3D data directly. In order to cope with the huge search space, an extension to particle filtering is proposed by combining it with gradient descent techniques.

The publications on hand tracking mentioned above should give the reader a feeling for the broad range of approaches that are followed in the research community. In a recent review (Erol et al., 2007), the literature on extracting the 3D hand configuration from visual data has been analyzed. Surprisingly, only two somewhat older approaches were identified that actually perform hand posture recognition in real-time (Shimada et al., 2001; Rehg and Kanade, 1994). In addition, the authors of the review conclude from the lack of an implementation operating in a real world system that there are unsolved theoretical questions, most importantly the high dimensionality of the search space and the various occlusions occurring during arbitrary hand motions. This underlines important limitations for the current use of such model-based hand tracking approaches on interactive robots:

- Most approaches require that the approximate hand position is known at the start of the tracking. This constraint can be met in static setups where the start and end of an interaction are clearly defined, but it is not applicable in a human-robot interaction where the start of the gesture can happen any time and a large part of the hand may be occluded in between the gestures.
- Due to the large search space and the high flexibility of the hand, approaches using a single camera, i.e., no depth data, are usually not able to operate in real-time with onboard computational power. In addition, for extracting initially the coarse hand position (and also after full hand occlusion), other techniques like model-based body tracking have to be used. These impose additional computational constraints.

As the exact hand configuration is often not needed at all times but only at certain points during performing a complete gesture, it seems more feasible for recognizing hand gestures to combine an approach for continuously tracking the overall body configuration (see Section 4.6) with a hand posture detection from Section 3 activated on demand. In this way, the dynamic aspects of the gesture are captured adequately and computational effort to extract the hand posture is only used if the information about the hand configuration is relevant for the gesture recognition. Obviously, this requires to estimate from the coarse hand position and the overall trajectory when to actually perform a detailed hand posture recognition. Applying model-based approaches also to full body tracking will be considered in more detail in the following Section.

4.5.2 Model-based Tracking of Arm/Body Configurations

Tracking the full body can be done by applying either implicit body models capturing the overall body configuration in a holistic representation or with explicit body models specifying the individual body constituents.

Holistic, Implicit Body Model

Using an implicit body model means learning a direct mapping from a known body configuration to its visual appearance. Within an image sequence, the tracking of the current 3D body configuration amounts then to comparing the individual images to previously learned, image-based reference models. This mapping is likely to be ambiguous if only a single image is used. However, through considering temporal restrictions, i.e., body configurations that are closely related to the configuration detected in the last image, this approach becomes feasible.

Agarwal and Triggs (2006) present such an implicit modeling approach for 3D human pose extraction from silhouettes. Here, a silhouette image of a human is obtained by applying background subtraction (cf. Section 3.2.1). Local shape contexts are computed on edge points of this silhouette by building angular histograms with 60 bins around the edge points. By applying clustering techniques, the feature vector for a specific shape can be reduced to a reasonable size of 100 components instead of representing it by the large sum of all angular histograms. A mapping between the obtained feature vectors and the true body joint angles is learned using nonlinear regression techniques. In addition to this static mapping, the dynamics are incorporated into the tracking framework to deal with ambiguities inherent in 2D shapes and the associated feature vectors. The overall approach is, therefore, capable of tracking the 3D human pose over an image sequence without an explicit body model.

It should be noted, however, that learning such implicit models - similar to the appearance-based hand detection methods outlined in Section 3.3.2 on page 64 - requires that all relevant body configurations are included in the training set to capture their visual appearance and temporal relations. Furthermore, the available pose information always refers to the complete body, i.e., no details of the pose are captured.

Modular, Explicit Body Model

Tracking the body by using some kind of explicit body model is a more typical approach to deal with the wide variety of different body configurations exhibited by humans during everyday life. The level of detail of the body model depends on the required precision of tracking and the available computational resources. Especially the latter point has seen drastic improvements in the last years, leading to more complex body models in recent research activities.

As image data from a single video camera contains only 2D information, a straightforward approach is to use algorithms that perform body tracking based on a simple 2D model of the human body. One well-known approach that models the individual limbs by a blob-based representation containing their color and contour is the PFINDER system developed by Wren et al. (1997). Due to the coarse body model, the 2D system achieved tracking at video frame rates. However, it provides just a rough estimate of the body configuration that is only suitable for detecting gestures made up by gross body motions. For the recognition of most gestures that may occur in human-robot interaction (see Section 2.1.3), a more detailed body model including the 3D body configuration is needed.

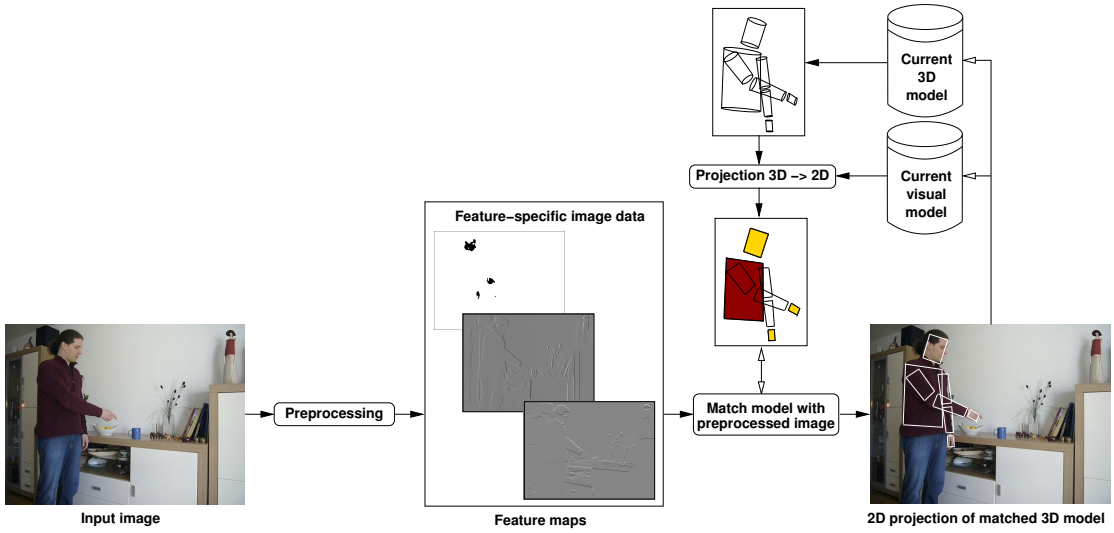


Figure 4.14: Projecting a 3D body model and an associated visual model to a 2D representation and matching it with preprocessed image data.

The use of an explicit 3D model of the arm or the whole body for tracking is nowadays very prominent. Surprisingly, already in the beginning of the 80s Hogg (1983) proposed one of the first approaches using a 3D body model. The target was the tracking of a walking human in monocular image data. The body was modeled consisting of articulated 3D cylinders viewed under perspective projection. The input image was segmented and pairs of parallel lines were extracted and subsequently matched to the legs of the projected 3D model.

In more recent approaches, all parts of an articulated 3D body model are used for matching the model to the input data. Figure 4.14 depicts the different processing steps of such an approach. Cylinders with ellipsoid cross sections are often used for the 3D body model as this representation generates good result when a cylinder is observed by the camera from the side and is computationally not too demanding. Such a coarse model of the whole body needs at least 34 degrees of freedom. The kinematic structure is completed by defining individual joint angle limits which model the physical constraints of the human body.

In surveillance and intelligent room setups, multiple cameras are used for human tracking in order to cope with body self-occlusions (see, e.g., Ramanan and Forsyth, 2003; Sigal et al., 2004; Kehl et al., 2005; Deutscher and Reid, 2005; Pons-Moll et al., 2011). Some of these approaches operate in a top-down fashion, i.e., similar to Fig. 4.14 they use a body model and aim to find features matching the limbs in the image (Deutscher and Reid, 2005; Kehl et al., 2005). Other approaches are

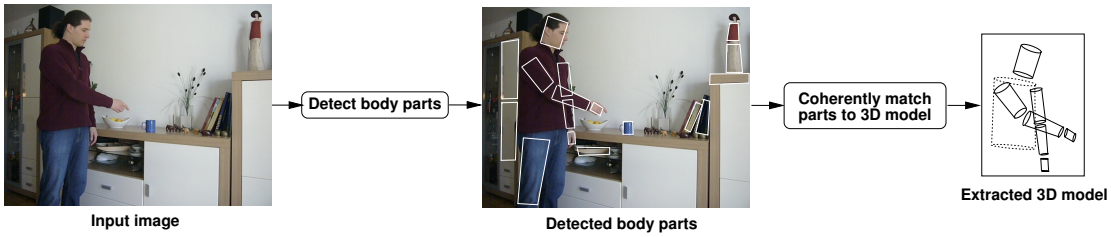


Figure 4.15: Bottom-up detection of 2D body parts and finding the 3D body model that best matches with the individual part detections.

taking the opposite direction and construct in a bottom-up fashion a body model out of individual body parts detected independently from each other in the image (Sigal et al., 2004; Ramanan and Forsyth, 2003). Such a data-driven approach is visualized in Fig. 4.15.

Notice that all multi-camera approaches are computationally expensive due to the large mount of input data. Another more important drawback is the fact that these approaches require multiple cameras observing the human from different directions. This is obviously only possible in a stationary setup like, e.g., in surveillance tasks, but not for the targeted application of gesture recognition on a mobile robot that has to carry all sensors onboard.

Although 3D tracking of an arm or whole body model using a single viewpoint is challenging, several approaches have addressed the problem (Sidenbladh et al., 2000; Agarwal and Triggs, 2006; Sminchisescu and Triggs, 2005; Schmidt et al., 2006; Ziegler et al., 2006; Hahn et al., 2008; Wöhler, 2009; Hecht et al., 2009), partially using similar techniques as in multi-camera approaches.

For example, Sidenbladh et al. (2000) apply several different gray-level image cues for tracking a detailed 3D human body model. A particle filter with importance sampling (see Section 4.2.2) is applied for tracking the human motions. To cope with the huge search space, motion priors are used to predict the 3D body configuration prohibiting the tracking of unconstrained motions. Extending this framework, Sminchisescu and Triggs (2005) added a more precise modeling of the 3D body model and a more complex parameter space exploration, but the computational time required prohibited its use for real-time tracking.

To cope with a large parameter space, kernel-based Bayesian filtering (see Section 4.2.2) has been applied by Schmidt et al. (2006) to 3D body tracking to reduce the computational effort necessary for searching good body configurations. A detailed description of the algorithmic framework of this body tracker is provided in the next Section.

The result of the different model-based arm/body trackers outlined above is the sequence of body configurations over time. This data is technically similar to what can be obtained from intrusive sensing methods (see Section 2.2.1 on page 31). For the actual recognition of gestures from this body tracking data, a variety of pattern matching algorithms can be applied. Especially the choice of suitable features for gesture recognition is strongly dependent on the types of gestures to be recognized. Often the 3D hand trajectory is chosen, but this only allows for the recognition of gestures that are independent of the rest of the body. In nearly all categories of gestures occurring in human-robot interaction (see Section 2.1.3 on page 26), however, there are gestures that cannot be recognized based on the hand position alone. For example, in a conventional gesture one might point with the finger to the head to indicate 'I have an idea!'. Similarly, in a pointing gesture not only the pointing finger but also the head position may be relevant to identify via the eye-hand-object line (see Fig. 1.6 on page 16) which object was referenced. Having the information of the complete body posture allows to incorporate this context from the current body configuration for understanding hand gestures (see also Section 6.1.1).

4.6 Example: Tracking 3D Human Body Configurations

As pointed out earlier, multiple cameras observing the human from different directions cannot be used onboard a mobile robot. Consequently, for human-robot interaction an algorithm estimating the pose of the gesturing human from a single observation point is needed. Obviously, with a monocular camera motions of body segments in depth, towards or away from the camera, cannot be tracked precisely. Note, however, that the limited tracking quality of an approach operating on a monocular image sequence is not critical for most gestures occurring in human-robot interaction. As long as tracking is not lost and a rough estimate of the body configuration is always available, such an approach is well suited for human body tracking onboard a mobile robot. Nevertheless, the tracking could be made more robust if depth from the disparity of a stereo camera setup is available.

This Section gives an example of a model-based approach for 3D human pose tracking from monocular images, a more detailed description can be found in Schmidt (2009). The described method focusses on tracking only the upper body as the legs usually have no influence on gestures performed with the hands and arms. An overview of the overall method is given in Section 4.6.1. The approach relies on

a variety of image cues that are extracted from the color images and matched with a 3D body model. For the detection of the torso as well as the upper and lower arm the approach applies edge, ridge, and color cues. A skin color model is used for detecting a person's hands and face. These cues as well as the internal 3D body model are described in Section 4.6.2.

The matching of the image cues to the body model is performed by way of a probabilistic tracking framework described in Section 4.6.3. More specifically, a kernel particle filter which avoids the need for a huge number of particles to represent probability distributions in high dimensional state space is used (see Section 4.2.2). On the basis of mean shift based mode-seeking the dominant mode is determined and then used to select the particles for the next time step. In order to better track moving body parts, a linear motion model is used for propagating a fraction of the particles while uniform random noise is used for the other part.

The parameterization of this framework for tracking in an interaction setting is detailed in Section 4.6.4 together with a quantitative evaluation of the tracking performance. The experimental results demonstrate that such a type of approach is well suited to support human-robot interaction by enabling body tracking onboard a mobile robot.

4.6.1 System Overview

An overview of the processing steps necessary for matching 3D object features of a generic human model to 2D image features extracted from input images is depicted in Fig. 4.16.

Assuming an initial body pose has been acquired in a special initialization step (Schmidt and Castrillon, 2008) or the system is already successfully tracking a human, one iteration of the algorithm can be described as follows:
An input image is acquired with an uncalibrated monocular color camera (1) and preprocessed. The resulting image features (2) are used for evaluating the probability of the different configurations of the body model from the previous time step. The outcome (3) is a probability distribution which is further explored in multiple iterations (4) of the mean shift algorithm. This enables the identification of different modes (5) of the underlying probability distribution from which a single mode (6) representing the most likely human body pose is selected. This mode serves as output for subsequent recognition algorithms and is also used as input (7) for the next time step of the particle filter. Particles generated from this mode are then - partly after applying a motion model (8) - disturbed (9) to give an estimation of body configurations for the next time step. This concludes the processing of

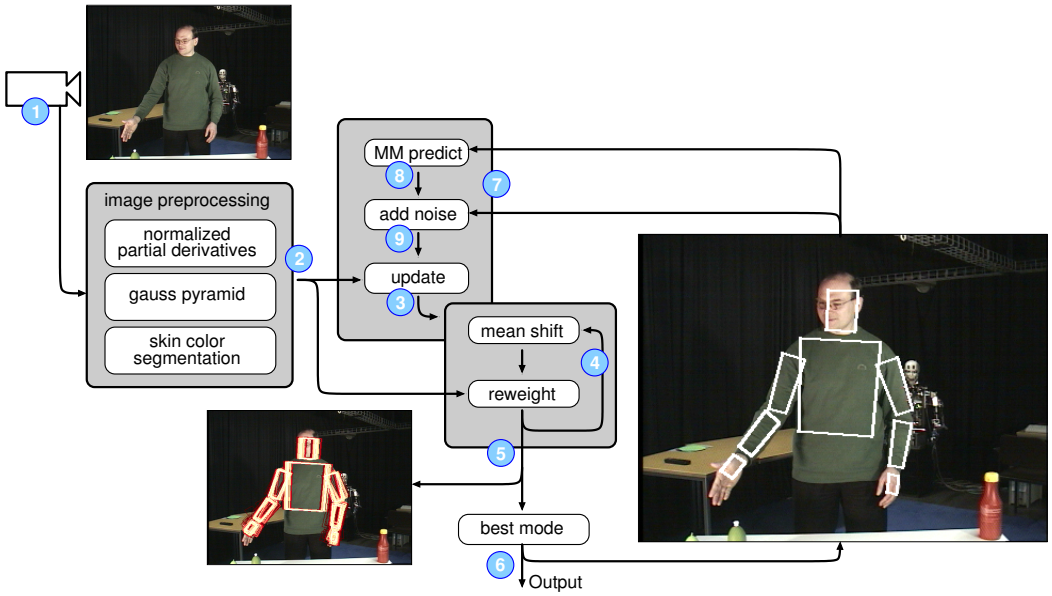


Figure 4.16: Outline of the probabilistic pose tracking algorithm.

the current input image and a new image is acquired (1). This is preprocessed (2) and now the configurations propagated from the last time step are evaluated (3) on these new image features.

4.6.2 Modeling the Appearance of Humans

In order to track the body configuration for human-robot interaction, a model of the upper body with two arms is often sufficient and is used in this example approach. The 3D body model is back-projected into the image plane using a pinhole-camera model. This yields an approximate 2D representation of the 3D body model consisting of a number of 2D polygons in the image plane that can now be used to evaluate how well the observed image matches a 3D body pose. Estimating the likelihood of a specific pose is done by combining likelihoods for each limb $l = 1, \ldots, L$ of the body model. The likelihood of an individual limb is calculated from up to four image cues $c \in \{E, R, C, S\}$, where E stands for the edge cue, R is the ridge cue, C is the mean color cue and S denotes the skin color cue. For each cue, a filter response is obtained by evaluating the cue at specific image positions and normalizing over all evaluated image positions belonging to a single limb. Figure 4.17 shows the backprojected 3D model of the upper body together with a visualization of some details for calculating the different cues.

The image processing of the individual cues resulting in associated likelihoods

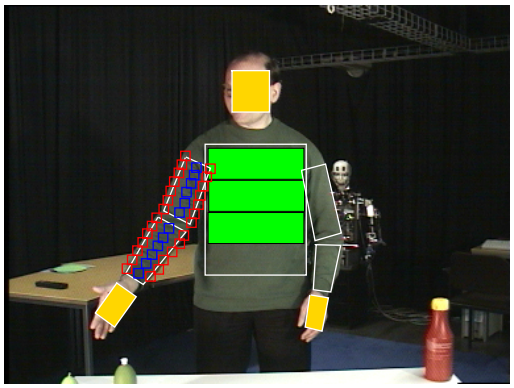


Figure 4.17: Backprojection of 3D body model on 2D image plane. Green rectangles show color cue area and yellow boxes denote skin color cue. The left arm depicts feature point positions exemplarily: red boxes denote sample points for edge cue, and blue boxes show sample points for ridge cue.

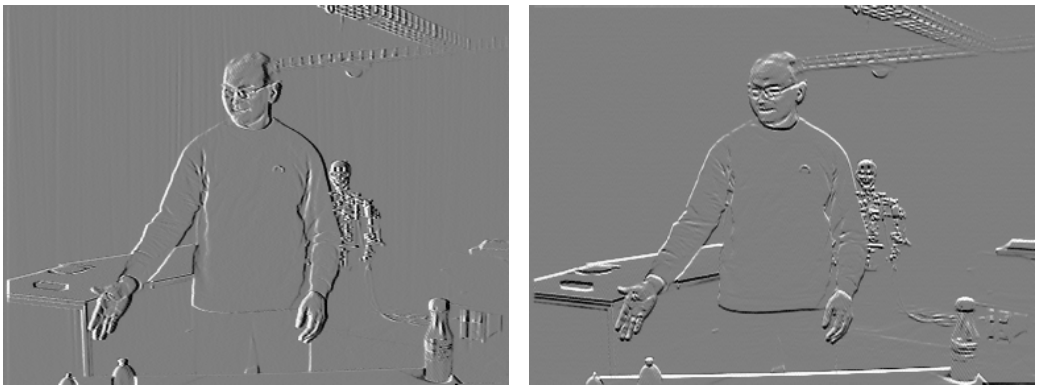
for each different limb is discussed in the next paragraphs.

Edge Cue

The edge cue as proposed by Sidenbladh (2001) uses the first partial derivatives that are sensitive to strong changes in contrast. For recognizing human body parts the presence of edges is most important, not their magnitude. Therefore all images with partial derivatives are scaled with a nonlinear normalization function. This function smoothes low magnitude edges stemming from textured backgrounds and emphasizes stronger ones that are expected to result from objects overlapping at different depths. The edge cue provides an accurate match for the position of a limb by comparing the angle of the edge gradient $[\partial_x(\mathbf{z}), \partial_y(\mathbf{z})]^T$ (see Fig. 4.18) with the estimated limb angle α, which has been obtained from the 3D model. This is done for $m = 1, \ldots, M_E$ feature points $\mathbf{z}^{(m)}$ positioned equally spaced on the limb boundaries (see also Fig. 4.17). Here and in the next paragraphs the notation $\mathbf{z}^{(m)} = [x, y]^T$ is used to denote the location of one pixel in the image plane. The response of such an edge filter at pixel position $\mathbf{z}^{(m)}$ is:

$$f_E^{(l)}(\mathbf{z}^{(m)}) = \partial_y(\mathbf{z}^{(m)}) \cos(\alpha) - \partial_x(\mathbf{z}^{(m)}) \sin(\alpha), \tag{4.25}$$

The filter response for the whole limb can then be calculated by averaging over all M_E feature points of this limb:



(a) (b)

Figure 4.18: Partial derivatives after scaling with the normalization function. (a) derivative in x direction; (b) derivative in y direction;

$$\bar{f}_E^{(l)} = \frac{1}{M_E} \sum_{m=1}^{M_E} f_E^{(l)}(\mathbf{z}^{(m)}). \tag{4.26}$$

The filter response represents how well the boundaries of a limb of the 3D body model match the edges in the image data.

Ridge Cue

The ridge cue inspired by Sidenbladh (2001) is utilized to find elongated structures of a specified thickness in the image. The torso as well as the upper and lower arms are limbs that have such an elongated structure. As the ridge cue depends on the size of the limbs in the image, it will only provide appropriate results if the observed limb is in a particular distance to the camera. In order to enable the calculation of this cue for limbs having different distances to the camera, a Gaussian image pyramid is constructed from the input image. Based on the evaluated body model providing the distance of the limb to the camera, the correct resolution level μ in the Gaussian image pyramid is then selected for calculating the ridge cue. The cue suppresses point-like edge features by searching for edges parallel to the expected limb angle α and missing edges in perpendicular direction. This is achieved by evaluating the normalized second partial derivatives $[\partial_{xx}^{(\mu)}(\mathbf{z}), \partial_{xy}^{(\mu)}(\mathbf{z}), \partial_{yy}^{(\mu)}(\mathbf{z})]^T$ at $m = 1, \ldots, M_R$ feature points $\mathbf{z}^{(m)}$ equally distributed on the main limb axis (see also Fig. 4.17):

$$f_R^{(l)}(\mathbf{z}^{(m)}, \alpha) = \tag{4.27}$$
$$\left|\sin(\alpha)^2\, \partial_{xx}^{(\mu)}(\mathbf{z}^{(m)}) + \cos(\alpha)^2\, \partial_{yy}^{(\mu)}(\mathbf{z}^{(m)}) - 2\sin(\alpha)\cos(\alpha)\, \partial_{xy}^{(\mu)}(\mathbf{z}^{(m)})\right|$$
$$- \left|\cos(\alpha)^2\, \partial_{xx}^{(\mu)}(\mathbf{z}^{(m)}) + \sin(\alpha)^2\, \partial_{yy}^{(\mu)}(\mathbf{z}^{(m)}) + 2\sin(\alpha)\cos(\alpha)\, \partial_{xy}^{(\mu)}(\mathbf{z}^{(m)})\right|$$

The filter response for the whole limb l is computed by averaging over all M_R feature points on the main limb axis:

$$\bar{f}_R^{(l)} = \frac{1}{M_R} \sum_{m=1}^{M_R} f_R^{(l)}(\mathbf{z}^{(m)}, \alpha). \tag{4.28}$$

Compared to the edge cue, the ridge cue gives a coarser estimate of the limb position. Its advantage over the edge cue is its robustness as it produces less false maxima.

Mean Color Cue

The mean color cue models the appearance of a limb using for different parts of the limb specific color models. The algorithm positions B_l rectangular polygons on a limb l. The mean color value is calculated trough averaging over the color values of the M_C pixels in the polygon b positioned in the back-projected limb. The number of polygons B_l and their positions are chosen on the basis of the limb type (see also Fig. 4.17).

To calculate the filter response, the mean color $C_t(\mathbf{z}^{(b,l)})$ of each polygon b at position $\mathbf{z}^{(b,l)}$ on limb l is compared to the adapted mean color $\bar{C}_{t-1}^{(b,l)}$ of this polygon using the L2 norm in the utilized RGB color space:

$$f_C^{(b,l)} = \sqrt{\left(C_t(\mathbf{z}^{(b,l)}) \;-\; \bar{C}_{t-1}^{(b,l)}\right)^2}. \tag{4.29}$$

The filter response for the complete limb l is then calculated from:

$$\bar{f}_C^{(l)} = \frac{1}{B_l} \sum_{b=1}^{B_l} f_C^{(b,l)}. \tag{4.30}$$

To deal with varying illumination conditions the current mean color values are adapted according to the mean color values $\hat{C}_{t-1}(\mathbf{z}^{(b,l)})$ extracted on the basis of the back-projection of the best mode extracted from the tracked 3D body models in the last time-step $t-1$:

$$\bar{C}_{t-1}^{(b,l)} = \beta \cdot \hat{C}_{t-1}(\mathbf{z}^{(b,l)}) + (1-\beta) \cdot \bar{C}_{t-2}^{(b,l)} \tag{4.31}$$

where β is an adaptation factor. In most situations the mean color cue reliably finds the coarse limb position as color is usually a very discriminative cue. In situations where parts of the background have a color similar to the clothing of the human, the cue becomes less reliable.

Skin Color Cue

The skin color cue uses a simplified form of the skin-color based hand detection outlined in Section 3.2.2. Similar to the description there, the rg color space is employed as it allows to quickly generate a person- and situation-specific color model that is relatively robust to ongoing illumination changes. The skin color model needs to be initialized in the setup phase with skin-colored pixels from, e.g., a face detector (Fritsch et al., 2002; Schmidt and Castrillon, 2008). Subsequently, it allows to find the position of the hands and the head (see Fig. 4.19) but may fail if lighting conditions start to differ from the initialization phase. This drawback could be overcome if the skin color model would be actively adapted to changing lighting conditions as described in the example approach in Section 4.4. However, as a probabilistic tracking framework is applied here, it is computationally more effective to use a rather coarse static skin color model without adaptation and put more effort in the probabilistic combination with the other cues.

Figure 4.19: Binary segmentation image after applying skin color model.

For calculating the skin color cue, all M_S pixels $\mathbf{z}^{(m)}$ in a single polygon on the limb l are evaluated. A single polygon is sufficient as only the head and hand limbs (see Fig. 4.17) are analyzed. The filter response for the limb l is calculated using the ratio of pixels being classified as skin or non-skin within the polygon:

$$\bar{f}_S^{(l)} = \frac{1}{M_S} \sum_{m=1}^{M_S} \psi(\mathbf{z}^{(m)}) \tag{4.32}$$

where $\psi(\mathbf{z}^{(m)}) = 1$ if the pixel belongs to the skin class and $\psi(\mathbf{z}^{(m)}) = 0$ if not. As neither the hands nor the head have a rectangular form, this cue does not provide a precise position estimate. However, if only small parts of the background are skin-like, this cue is very good at giving a coarse estimate of the positions of head and hands.

Observation Model

The edge, ridge, mean color, and skin color cues generate a separate filter response for each limb. The filter responses are converted into likelihoods using the following Gaussian weighting function:

$$p(c, l) = \exp\left(-\frac{(\bar{f}_c^{(l)})^2}{2\,\sigma_c^2}\right) \tag{4.33}$$

where the standard deviations σ_c are derived from the variability of the responses of each utilized cue c. To account for the variations in the number of cues per limb, the cue likelihoods of an individual limb l are scaled according to the total number of cues N_l for this limb. In this way, the likelihood of each limb contributes equally to the likelihood of the overall body pose. Assuming that the cues and limbs are independent, the likelihood of a pose is calculated as:

$$p(\mathbf{y}_t \mid \mathbf{x}_t) = \prod_{c \in \{E,R,C,S\}} \prod_{l=1}^{L} p(c, l)^{\frac{1}{N_l}} \tag{4.34}$$

Using this observation model, the likelihood that a certain body pose $\mathbf{x}_t$ consisting of the configuration of the L different limbs causes the observation $\mathbf{y}_t$ can be calculated. Here, the observation is the actual image that is used for calculating the filter responses of the individual cues.

4.6.3 Tracking Multiple Body Configuration Hypotheses

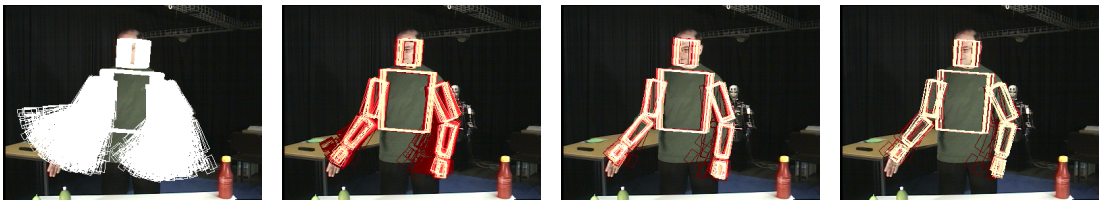
Through modeling the appearance of the human as described in the previous Subsection and applying the observation model of Eq. 4.34, it is possible to calculate the likelihood for a specific body pose. The question is for which body poses to calculate the likelihood and how to incorporate a temporal constraint, i.e., the fact

that the body pose can change only smoothly over time. Abstracting from the specific application domain, the problem of searching in a high-dimensional space for the best matching hypotheses is a common pattern recognition task. Especially in image processing, the evaluation of a specific point in search space is often computationally costly so that the evaluation of all possible points becomes impossible. In sequence processing, temporal constraints can be incorporated to reduce the search space. However, due to the many ambiguities resulting from the projection from 3D to 2D in the observation model, tracking a single body pose with a Kalman filter (see Section 4.2.1) will be too restrictive. Therefore, we here employ particle filtering for tracking likely body poses (see Section 4.2.2).

Earlier approaches for 3D human pose tracking have made use of standard particle filtering (see, e.g., Sidenbladh, 2001), but the large search space resulting from the high degrees of freedom of the body model made tracking difficult. A large number of particles was needed for successful tracking, resulting in high computational requirements of the tracking algorithm and prohibiting real-time operation of the algorithm.

To overcome this drawback and enable real-time tracking of the body configuration, a kernel-based particle filter can be applied (see Section 4.2.2). In this implementation, the initial bandwidth h_0 of the kernel is scaled at every iteration i according to $h = 0.8^i h_0$ where the value 0.8 has been determined empirically, similar to (Chang and Ansari, 2005). Mode-seeking is performed until a maximum number of iterations has been reached or until the Euclidean distance between the corresponding modes in the last two iterations is below an empirically determined threshold, i.e., the mode position does not shift anymore. Figure 4.20 shows a graphical representation of the posterior distribution for different numbers of iterations.

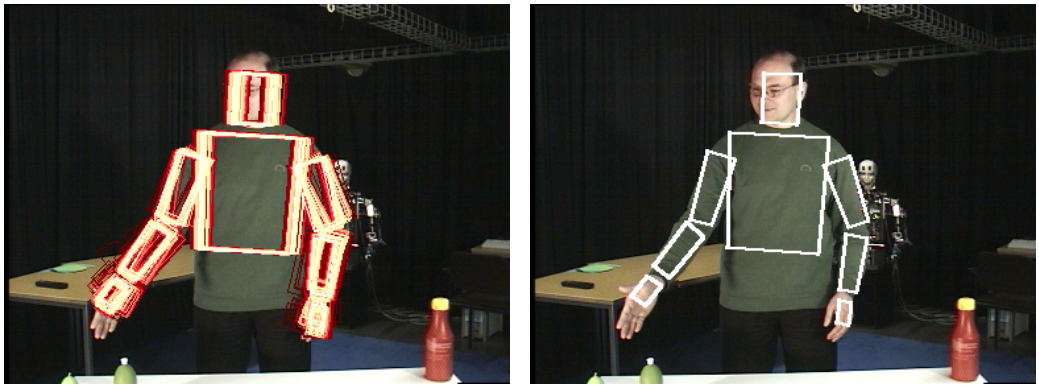
Following mode-seeking, the most dominant mode is obtained by a weighted



(a) (b) (c) (d)

Figure 4.20: Spread of hypotheses of current body configuration for different numbers of mean shift iterations (increasing from left to right)

averaging over all particles in a window centered at the peak of the posterior distribution. This mode serves as estimate of the current body pose and is called mean mode. The back-projection of this pose into the image plane (see Fig. 4.21(b)) is utilized as reference model for updating the mean color using Eq. 4.31.



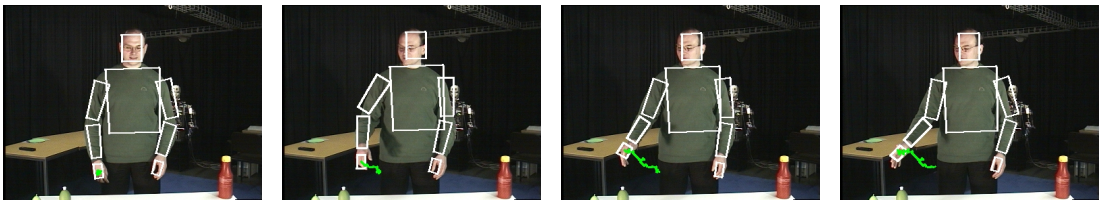
(a) (b)

Figure 4.21: (a) Particle-based body configurations colored according to their likelihoods; (b) Estimate of body configuration generated through calculating mean mode.

The particles are propagated from the dominant mode obtained at the end of one iteration to the next time step based on two different strategies:

- A linear motion model is used for propagating a fraction of all particles. The velocity of the individual limbs captured in the linear motion model is estimated on the basis of the best mode at time t and the best mode at time $t-1$, i.e., by comparing the two configurations.
- The remaining particles are subject to random propagation.

The ratio of particles propagated with the two strategies is derived from the success of the strategies in the previous time step. For each of the two strategies the cumulated probabilities of the particles in the posterior are calculated. If one strategy is better than the other, the particles stemming from the propagation with this strategy will exhibit higher probabilities and, therefore, this strategy will be allocated a larger fraction of the particles in the current time step. A minimum percentage of random propagation is enforced to guarantee the ability to recover if the motion model does not adequately capture the current motion.



(a) (b) (c) (d)

Figure 4.22: The tracked hand trajectory (green) resulting from model-based 3D body tracking.

Particles propagated with the motion model are subject to uniform noise with variance $0.25\,D$ while uniform noise with variance D is used for the particles that are subject to random propagation. The values of the joint angle variances D have been determined experimentally based on the camera view and the application domain. If during the propagation the addition of the noise results in an invalid body configuration, the propagation step is repeated until a valid body pose is obtained. For the mean shift iterations, the initial bandwidth is chosen as $h_0 = D$. The window for determining the most dominant mode after mode-seeking is set to $0.25\,D$, this value has been determined empirically.

Note that a linear motion model is applied only for a fraction of the particles and that the velocity captured in the motion model is derived from the motion exhibited in the last time step. This model makes a very simple prediction without including any knowledge possibly available in a specific domain. Other approaches have included specific motion models like, e.g., a walking motion (Sidenbladh et al., 2000). Using motion priors makes tracking easier, but comes at the cost of a limited ability to cope with other unknown motions. Without specific motion models tracking is more challenging, but it allows to track any human motion as well as to recover from tracking failures more easily. Figure 4.22 depicts an example sequence including the hand trajectory obtained from tracking the overall 3D body model.

4.6.4 Evaluation Results

The framework described above can be used for tracking arbitrary human body motions. However, the computational requirements make it necessary to restrict the body model if tracking in real-time has to be achieved. Since in an interactive scenario with a robotic platform the hand gestures are most interesting, the human body model is limited to the upper body and the two arms (see also Section 4.6.2 on page 100). The hands are fixed to the lower arms and the head is fixed to the

torso leading to a model with 14 degrees of freedom. This reduction of the model complexity is acceptable for tracking a human that is oriented roughly towards the camera and interacts with a mobile robot.

The likelihoods of the different limbs are calculated by combining the likelihoods of the individual cues (Edge, Ridge, Color, Skin) for each limb:

$$\begin{aligned} c_{torso} &= \{E, C\} \\ c_{upperarm} &= \{E, R, C\} \\ c_{lowerarm} &= \{E, R\} \\ c_{hand} = c_{head} &= \{S\} \end{aligned}$$

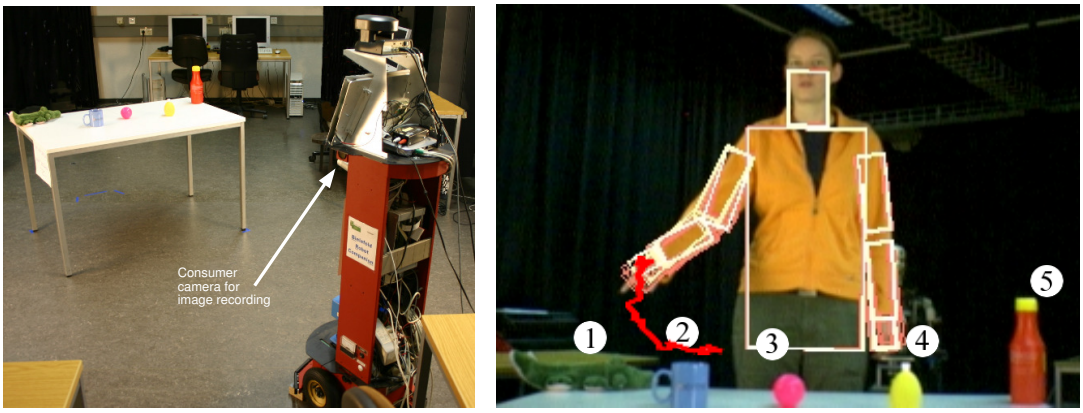
For the edge cue, $M_E = 20$ feature points are used and the ridge cue is evaluated at $M_R = 30$ feature points. The mean color cue is computed using $B_{arm} = 3$ polygons per arm and $B_{torso} = 4$ polygons for the torso. For the three shoulder joint angles and the elbow the joint angle variances are set to $D_{arm} = [22°, 20°, 35°, 25°]$. The variance of the torso D_{torso} is between 1°-3° for rotation and $1cm$-$5cm$ for translation. These parameters have been found to be an acceptable tradeoff between detection capabilities and computational load.

Before starting the actual tracking, the body model has to be initialized. Two fundamentally different types of initialization have to be performed:

Adapting the body model: The adaptation of the body model is done manually to account for the variability of the limb sizes of different humans. This is necessary as the matching is rather sensitive and a too large diameter of, e.g., the lower arm will result in a tendency to 'push the arm away' from the camera to reduce the backprojected arm width and allow a good match with the image data.

Setting the initial pose: To start the iterative tracking, the adapted body model must be in a configuration close to the current human body pose. With a body posture initialization scheme this can be done automatically (Schmidt and Castrillon, 2008). Similarly, the probabilistic tracking can be started on the initial image with initial particle configurations distributed over the complete search space. Through iteratively processing the initial image, the tracking may converge to the current pose of the human. However, in the evaluation described here, manual initialization of the body pose was performed.

In order to speed up processing, no occlusion test is performed. Consequently, matching difficulties can arise if the hands occlude the head or point in the direction



(a) (b)

Figure 4.23: Setup for evaluation: a) Overall scene including the mobile robot with the camera used for evaluation; b) Image depicting view of the onboard camera with an example trajectory and the five annotated objects.

of the camera. In such situations tracking may temporarily fail. However, the recovering capabilities of the particle filter allow the algorithm to continue the broken tracking when the occlusion ends.

For evaluation the setup depicted in Fig. 4.23(a) has been used: The human is standing in front of a table with five objects, facing a mobile robot. The person is asked to show the objects to the robot by pointing at them. The human's actions are recorded by the body tracking camera while it is ensured that the upper body is in the robot's field of view (see Fig. 4.23(b)).

The complete data set will be also used in later example approaches outlined in this book, therefore it described here in more detail: The experiments were performed with a total of four persons. After an initialization phase for the body tracking every person pointed at each object in sequence (crocodile, cup, ball, lemon, bottle), withdrawing their hand after each pointing gesture. Subsequently, the subjects had to raise the hand and wave into the camera, finally lowering the arm. All participants had to perform the same sequence of gestures three times changing the order of targets each time. Each subject was recorded performing the experiments four times, resulting in 16 sequences with a total number of 18572 images, equivalent to more than 20 minutes of video, counting 496 performed gestures in total.

The body tracking algorithm has been evaluated on three recorded sequences with 836 images in total. Ground truth for this evaluation has been generated by manually annotating the position of the human's hands and the head in the images.

The measures of the body model and an initial posture of the 3D body model are also defined manually for each sequence.

The system is configured in two ways:

1. Using 1500 particles and 6 meanshift iterations is the configuration that is used for carrying out the overall system evaluation.

2. Using 500 particles and three meanshift iterations results in a lower accuracy but achieves faster computation that is more likely to be real-time capable on embedded hardware, shown for comparison only.

Tab. 4.1 shows that a relatively high number of particles is still needed to ensure accurate tracking over a longer image sequence. Gestures that effect in self-occlusion are more likely to produce noisy results (cf. Fig. 4.24 around frame 270),

		1500 particles		500 particles	
Person	**# pic**	RMSE	$\boldsymbol{\sigma}$	RMSE	$\boldsymbol{\sigma}$
subject A	318	18.7	13.9	52.7	41.3
subject B	242	14.5	8.6	32.9	23.3
subject C	276	12.0	10.1	72.6	38.3

Table 4.1: Body tracking evaluation: Position error for the right hand. RMSE (root mean squared error) position error and standard deviation σ in [*pixel*]. RMSE for subject B also shown in Fig. 4.24.

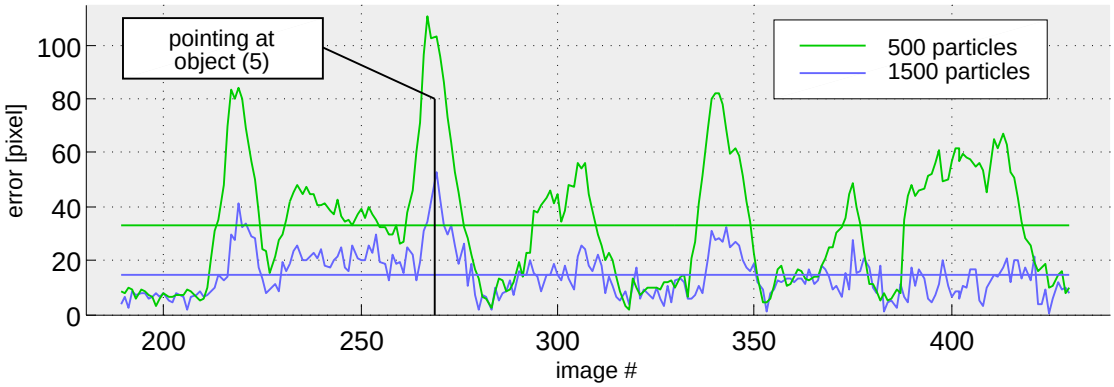


Figure 4.24: Tracking error of subject B for two configurations. Mean errors μ as in Tab. 4.1. Pointing at object (5) around frame 270 is difficult to track.

as the observation function for occluded limbs (e.g., the torso) provides imprecise or even wrong results. Fusing multiple cues for each limb and considering the body model as a whole helps to keep up tracking even if detectors for individual body parts were mislead. Both evaluated configurations are able to recover from tracking errors, but only the first configuration with 1500 particles permanently provides reliable trajectories.

4.7 Summary and Conclusion

In this Chapter the incorporation of temporal information into the gesture recognition process has been introduced. In its most simple form, the temporal tracking can be based on using the detection techniques outlined in Chapter 3 for finding the hand in individual images. By associating the individual detections over time it becomes possible to track a hand. This Chapter has introduced two standard techniques, Kalman filtering and particle filtering, for this association. A successful tracking is the basis for subsequently adapting the underlying representation used for achieving the hand detection in the individual images.

The two main types of detection methods introduced in Chapter 3 are using feature-based and model-based representations, respectively. In this Chapter, different methods for adapting these representations have been introduced. For adapting the visual features, the feature representation used for detection can be modified like, e.g., the mean skin color value. Alternatively, the detection process itself can be performed adaptively. For example, instead of only modifying the 2D hand shape to be matched after each detection, the shape model itself can be realized as adaptive representation. Tracking the hand then results in a continuous shape modification where knowledge about past modifications can be used to influence the matching process. Although this is still a feature adaptation, such a kind of processing is related to the model-based approaches to tracking that use explicit models of how the representation changes over time.

In order to provide an insight into the tasks necessary for feature adaptation, a tracking approach applying skin color adaptation has been outlined. In this example approach, low-level image data processing provides an abstraction of the input data by segmenting the image. The result of this processing are a number of image regions with a skin-like color. An important aspect is that for this feature adaptation neither knowledge about the gesturing human nor about later processing stages needs to be available. In other words, the algorithm works independently of the context in which it is applied. While this makes its implementation easier,

it results in a stronger susceptibility to noisy input data or ambiguous situations. Furthermore, the results are only of a limited expressive power providing information about the 2D position of skin-colored moving image regions which may or may not be human hands and faces. Consequently, for separating hands and faces, additional face detection methods have to be applied or the tracked object positions have to be analyzed to separate moving hands from static faces.

As alternative to feature-based approaches, model-based approaches can be used in an adaptive manner. The model can be either some kind of hand representation or even a representation of the complete body. These models are, therefore, providing more details about the configuration of the hand or body while feature-based approaches are usually more focussed on tracking the position only. Tracking of the changing 3D hand configuration is important for understanding the details of an object manipulation. Many temporal gestures performed when interacting with an intelligent robot, however, are characterized by larger hand motions and not only the changing hand configuration. Through tracking the overall 3D body pose, not only the hand motion itself but also its relation to the other body parts can be used for gesture recognition. Depending on the features used for matching the model to the image data, such an implementation can be much more robust with respect to the image background than just tracking a single hand. Such a system, therefore, seems to be most suited to provide the basis for the recognition of a wide range of different gestures and for the dynamic extension of the gesture vocabulary.

This Chapter also provided an example of a probabilistic framework for model-based 3D human pose tracking. An upper body model is used to represent the search space and joint constraints limit the search space. Temporal propagation is applied to restrict tracking in the search space to smooth transitions between body poses. The feature extraction method is obviously quite sophisticated, but it does provide features at a high abstraction level simplifying later processing stages. While for skin-colored region features it must be assured by later stages that the regions are actually hands, this information comes for free when applying the pose tracking feature extraction. In addition, the probabilistic nature allows to incorporate the confidence of the tracking approach (i.e., how focussed the PDF is) in the later processing stages. Consequently, feature extraction approaches that incorporate model information are becoming more important in recent years for realizing gesture recognition systems.

The next Chapters will detail how the tracking data can be used to perform gesture recognition and how recognized gestures can be combined with context information to, for example, help to resolve multi-modal object references given by a human interacting with a mobile robot through speech and gesture.

5 Recognition of the Gesture

In the previous Chapters the detection of the hand and the tracking of the hand motion over time have been covered. In this Chapter different algorithmic approaches for the task of recognizing gestures based on this preprocessed data are reviewed. There is a huge variety of different pattern matching algorithms applied for gesture recognition, but quantitative evaluations between algorithms are rarely performed. The quality of a recognition algorithm is related to the quality of the features and these in turn are usually depending on the application domain, preventing the finding of generic rules for choosing the pattern matching technique. Therefore, this Chapter will present a qualitative overview over the different approaches that have been developed for recognizing gestures.

As already pointed out in Chapter 1, many gestures in human-robot interaction are related to the objects in the environment. Their recognition is, therefore, only possible by combining information about the hand motion with such context data. However, as the extraction of context information is highly dependent on the application domain, approaches targeting the recognition of context-dependent gestures often contain domain-specific processing methods for context extraction. Such approaches will be covered in the subsequent Chapter 6. A common part of all gesture recognition approaches, however, is the analysis of the hand motion which is the focus of this Chapter.

Recognition algorithms can be separated into holistic and modular approaches. Section 5.1 will outline holistic algorithms that are based on holistic image representations and usually do not require any preprocessing like detection and tracking. However, due to the inherent limitations of holistic gesture recognition approaches for recognizing a wide range of gestures, modular approaches building on the tracking algorithms outlined in the previous Chapter are much more common. Section 5.2 will introduce the general design of such approaches for matching trajectories. Subsequently, methods for performing gesture recognition by matching hand trajectories to gesture templates using deterministic (see Section 5.3) and probabilistic (see Section 5.4) algorithms will be outlined. Furthermore, an example of a probabilistic algorithm is given in Section 5.5 for trajectory-based recognition of pointing gestures before a summary concludes this Chapter.

5.1 Holistic Methods Applying Implicit Models of Hand Gestures

Holistic approaches to gesture recognition do not require an explicit tracking of the human hand motion. While Chapter 3 focused on approaches for hand detection in isolated images, approaches for recognizing whole-body gestures from single images are covered in Section 5.1.1. Holistic approaches operating on image sequences and implicitly including temporal information in the modeling are described in Section 5.1.2.

5.1.1 Single-Frame Gesture Recognition

Approaches for the holistic recognition of gestures in individual images require that the gesture resembles a characteristic body posture, i.e., that there is no relevant temporal information. If this precondition is met, similar techniques as outlined for the task of hand detection in Chapter 3 can be used to detect the overall body shape. However, for interaction with a robot this precondition mainly applies to command-like gestures. One type of gesture that is relevant in the context of this book and that can also be understood as a command gesture are static pointing gestures.

Figure 5.1 depicts a recent example of a holistic approach for deictic gesture recognition by Martin et al. (2010). The pointing gesture is used for telling a mobile robot where to go. At first, the user triggers the gesture recognition by saying 'Go there!' to the robot. For recognition the system first classifies the general pointing direction (left/right). After this preprocessing, a number of detailed features are extracted from the image (see Fig. 5.1). The classification is performed using a cascade of Multi-Layer Perceptrons (MLP) where first the rough pointing direction is extracted and a subsequent classification step determines the detailed pointing direction.

In the example by Martin et al. the recognition is carried out when a user command is given. In general, the recognition of gestures from a continuous video stream requires to have a high robustness against non-relevant gestures, i.e., to filter the input stream for relevant gestures. In analogy to the speech recognition community this is called 'gesture spotting' and is a highly relevant challenge for single-frame holistic methods. Such approaches can therefore only be used for gesture recognition if the gesture recognition is explicitly triggered, i.e., the gesture spotting problem is circumvented, or the gesture to be recognized is very characteristic.

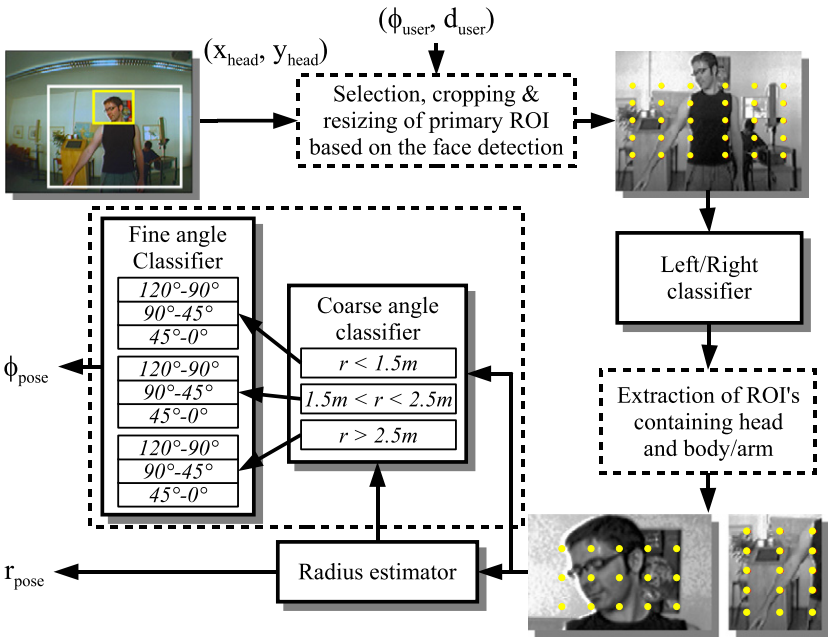


Figure 5.1: Holistic approach to pointing gesture recognition (image from Martin et al., 2010).

5.1.2 Holistic Temporal Models

Approaches performing the holistic recognition of gestures from image sequences represent the complete image sequence depicting a specific gesture or action by an integrated representation. Here, the image features of individual images are not considered in isolation, but rather their temporal changes within the complete sequence are encoded in the holistic representation. Consequently, this kind of approach implicitly solves at the same time not only the tracking task but also the pattern matching task of recognizing a gesture. The recognition result is therefore the specific gesture performed in the image sequence while no detailed information about the hand motion or configuration is available. The granularity of the representation, however, is an important choice in the design process, as it must be fine enough to enable the construction of a characteristic holistic representation.

A well-known approach providing the implicit recognition of gestures from image sequences was proposed by Bobick and Davis (2001). It uses for each sequence depicting a specific gesture two image-based templates encoding the motion known as *Motion-Energy Image* (MEI) and *Motion-History Image* (MHI). The MEI is a binary image representing the image positions where motion has occurred in the image sequence. The MHI represents the recency of motion in the sequence

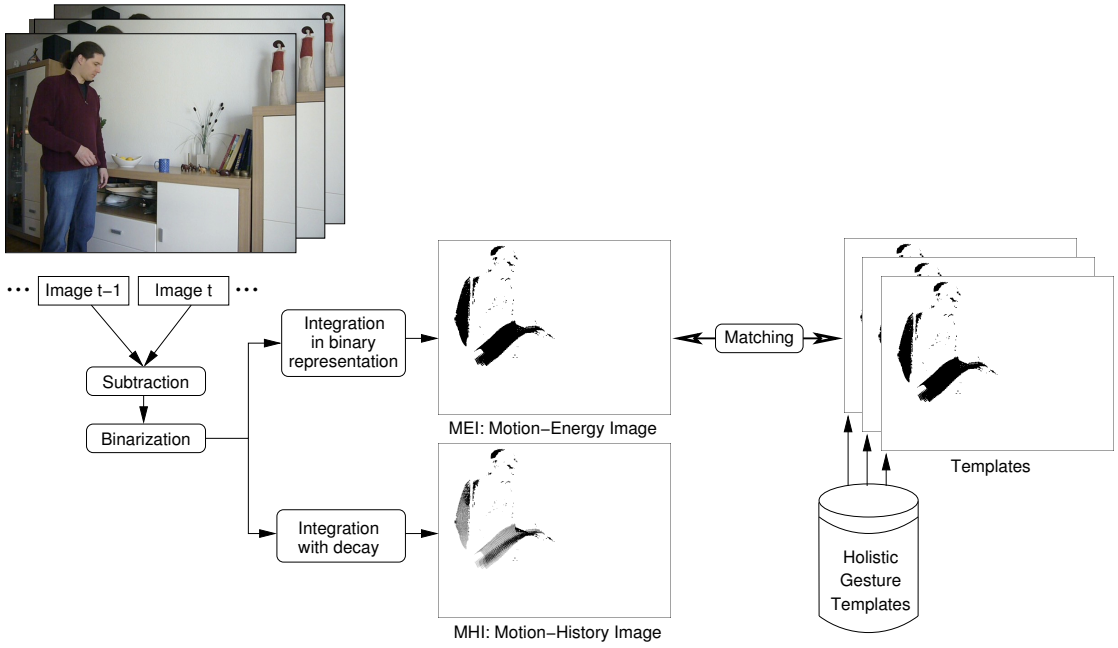


Figure 5.2: Motion-History Image (MHI) and Motion-Energy Image (MEI) for an image sequence depicting a pointing motion.

through integrating the motion images with a decay factor for "old" motion. The combination of MEI and MHI provides a view-based temporal template of a gesture. An example of an MEI and an MHI is depicted in the center part of Fig. 5.2.

As the complete image sequence depicting a specific gesture is represented in a single representation, the recognition simply has to perform a matching between the representation of the current sequence and all stored representations for the possible gestures. In the case of Motion-History Images, this amounts to a matching between image representations (see Fig. 5.2 right) where an obvious match criterion is the similarity on pixel level. However, some spatial similarity criterion may be needed as well to capture the possible spatial variations in different performances of the same gesture.

The approach of Bobick and Davis has a low computational cost and is therefore very well suited for implementation on a mobile robot. However, their method requires to build models of all relevant gestures, i.e., it cannot deal with unknown gestures. A more complex motion can also be a problem as – during the course of the gesture – the gesturing human may 'overwrite' image areas that contained the beginning of the gesture. In addition, it requires to have a static background, as any image motion not generated by the gesturing human will hamper the recognition.

This requires the observing camera to be fixed and is also a relevant restriction when considering interaction scenarios with more than one human. Nevertheless, a large number of different applications have been realized with this holistic representation, partially extending the framework to cope with the mentioned restrictions (for an overview of different applications see Ahad et al., 2010).

To give another example of a simple holistic representation, Ahn et al. (2009) record the complete gesture of a user and then construct a single image of the complete trajectory, i.e., a 2D representation of the motion. Subsequently, a neural network is applied to match this trajectory image to all stored templates.

Another prominent group of holistic approaches that has been used to recognize whole-body motions like walking, running, boxing, a.s.o., applies spatio-temporal features for representing the body motion (see, e.g., Schüldt et al., 2004; Dollár et al., 2005; Niebles et al., 2008; Laptev et al., 2008). In these approaches, sparse interest points are extracted using local space-time features. Figure 5.3 depicts two examples of such interest points.

Figure 5.3: Spatio-temporal feature points for two example images (images from Laptev et al., 2008).

For example, Laptev et al. (2008) perform the processing of these spatio-temporal features for the actual recognition as follows: During training, all interest points are grouped into classes and a representative for each class (a codeword) is determined, this is known as Bag-Of-Features (BOF). For the actual recognition, every interest point is converted into its associated codeword. For each cell of a predefined spatio-temporal grid (see Fig. 5.4), the histogram of codewords is calculated from all interest points in this grid cell. The overall feature vector for the complete image is then the concatenation of the histograms for all grid cells and its classification is performed using a support vector machine.

Many other holistic representations can be constructed and require other match criterions, but the basic setup is the same: the extracted holistic representation

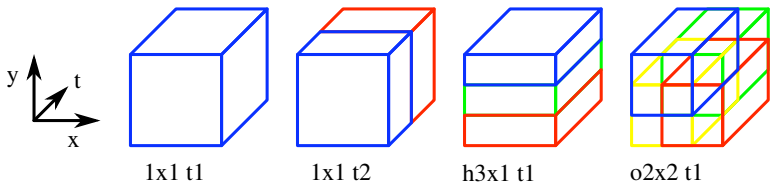


Figure 5.4: Examples of spatio-temporal grids (image from Laptev et al., 2008).

is matched to all stored representations. Correctly identifying the best matching template is a challenging task, but many pattern recognition techniques can be applied to this problem. It should be noted that in some applications the holistic gesture recognition is combined with hand detection and hand tracking techniques to capture not only the hand gesture but also the spatial embedding in a scene like, e.g., the distance of the hand gesture to some scene object.

Although a holistic representation has low computational requirements in the calculation of the representation and the matching to the templates, some practical challenges remain that limit the application of holistic approaches for gestural understanding on an intelligent robot:

Background dependency: The holistic nature of the representation usually results in a very limited abstraction from the context in which a gesture was performed. Especially early approaches like the Motion-History Images (Bobick and Davis, 2001) relied on a fixed background as the complete image area was used for building up the holistic representation. Later approaches have resolved this issue by using a preprocessing stage to extract the gesturing human from the background before generating the holistic representation (see, e.g., Agarwal and Triggs, 2006).

View-point dependency: After removing the background, the visual observation of the gesturing human is still view-point dependent. However, the orientation of the gesturer to the camera is not fixed, so a certain amount of view-point variability is to be expected. In order to capture this variability in the holistic representation, a more sophisticated abstraction from the raw image data by using, e.g., volumetric 3D information can be applied (see, e.g., Shin et al., 2005).

Granularity: The key difference between holistic and modular approaches to gesture recognition - the inherently coarse granularity of holistic approaches - has important consequences for their applicability in real applications. Holis-

tic approaches require that the gestures to be recognized are known at design time, i.e., no online learning of new gestures is possilbe, but the representation can be constructed for optimal recognition results. Furthermore, the gestures to be recognized need to have a certain level of granularity: the motion should not be too short in order to exhibit a characteristic representation, but it should also be of limited length to avoid an 'overwriting' of the holistic representation. For example, in the Motion-History Images a long motion will result in the blending out of the start phase while the Motion-Energy Images will exhibit a large energy stemming from two different time points in the gesture. For recognizing longer motions, therefore, a splitting into several elementary motions may be necessary to capture the motion adequately.

With the increasing interest in gesture recognition for intelligent robots, many holistic approaches have been proposed. However, as the previous paragraphs have pointed out, the strength of these approaches are the recognition of short, characteristic motions that are subject to only small variations in view-point. Consequently, they are typically used to recognize symbolic gestures that can serve for commanding a robot (see, e.g., Singh et al., 2006).

Due to the fact that the approaches have to be trained with the motions of interest beforehand, they are not the primary choice when targeting a general recognition method. Especially when the aim is to recognize a dynamically growing set of gestures, modular approaches like the ones covered in the next Sections are more suited.

5.2 Modular Methods for Matching Trajectories: General Design

Modular approaches to gesture recognition assume that hand detection and tracking have been carried out independent of the gestures to be recognized. These preceding stages provide the hand motion in the form of a hand trajectory which has to be matched with trajectory templates in order to perform the gesture recognition. This basic processing scheme is depicted in Fig. 5.5 using a 2D hand trajectory as example.

For the actual gesture recognition, a variety of trajectory representations are in use. Approaches using visual features provide trajectories representing the absolute hand position in 2D (Wren et al., 1997; Kölsch and Turk, 2004; Shan et al., 2007) while model-based approaches typically provide 3D trajectories (Stenger et al., 2001; Sminchisescu and Triggs, 2005; Schmidt et al., 2006; Ziegler et al., 2006; Hahn

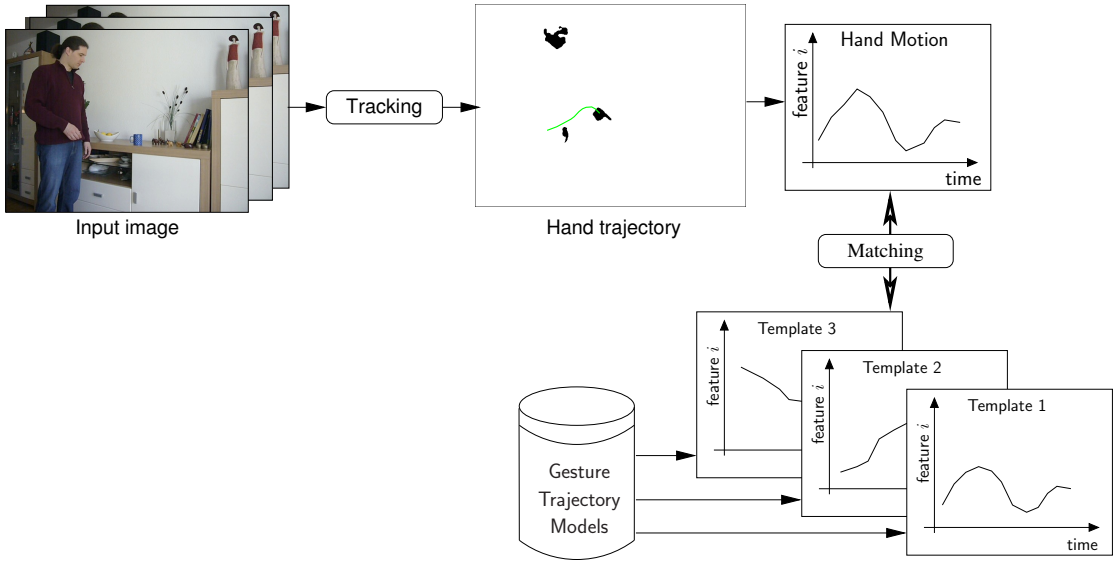


Figure 5.5: Schematic sketch depicting for a single feature i the recognition of a gesture by mapping the trajectory to stored templates.

et al., 2008; Wöhler, 2009). The 3D trajectories can be in absolute coordinates or relative to some reference point like, e.g., the torso of the gesturing human. In addition to the 2D or 3D trajectory information of the hand, other features like, e.g., the hand configuration can be included in the trajectory representation.

The algorithmic approaches for actually matching the observed hand motion to the stored templates are relatively independent of the contents of the representation. In the following Sections, different methods for achieving template matching are reviewed from the perspective of gesture recognition, i.e., the focus of the description is on what implications the algorithms have for realizing a gesture understanding system.

5.3 Deterministic Matching Methods

Deterministic methods for matching an observed feature vector to stored templates have been the starting point of the field of pattern recognition which today spans a wide range of algorithmic approaches. In the following, a short review of prominent approaches is given:

5.3.1 Direct Comparison

The most straightforward matching is the direct comparison of each feature vector element with each template element using some distance metric. This kind of matching requires certain preconditions to provide meaningful results:

Common length: All templates and the input feature vector have to have the same length to obtain comparable match values for the different templates.

Common temporal basis: The direct comparison of each element from the feature vector with elements from the templates requires that the individual elements are temporally associated to each other. Such a common temporal basis is not given if parts of a trajectory can be performed with varying speed, leading to a non-linear temporal stretching of parts of the feature vector.

Common signal amplitude: The matching criterion for a single feature element will contain some kind of distance metric, implicitly assuming a common signal amplitude between the observed trajectory and the templates. Obviously, this assumption is invalid if the trajectories can exhibit different amplitudes due to, e.g., variations in arm lengths or different expressiveness of the gesturing humans.

As these preconditions are usually difficult to meet, the direct comparison is not applicable for any challenging real world matching task.

5.3.2 Dynamic Time Warping

In order to enable a more flexible matching of feature vectors with templates, the temporal alignment restriction was the first to be relaxed. The initial proposal of the dynamic time warping algorithm was put forward by Sakoe and Chiba (1978) for the task of isolated word recognition by matching speech signal feature vectors to recorded templates. The basic idea of the algorithm is to find the optimal matching between two symbol sequences by warping the sequences non-linearly in the time dimension while adhering to some pre-specified restrictions. Figure 5.6 depicts a conceptual sketch of the matching process. The application of dynamic programming techniques resulted in very fast algorithmic implementations being widely used for deterministic matching of speech signals.

Following the success in isolated word recognition, some gesture recognition approaches also applied the dynamic time warping algorithm (see, e.g., Darrell and Pentland, 1993; Corradini, 2001; Li and Greenspan, 2005; Veeraraghavan et al., 2006; Ten Holt et al., 2007). A disadvantage of the method is the reliance on pre-segmented input vectors to be warped and matched. While in isolated word

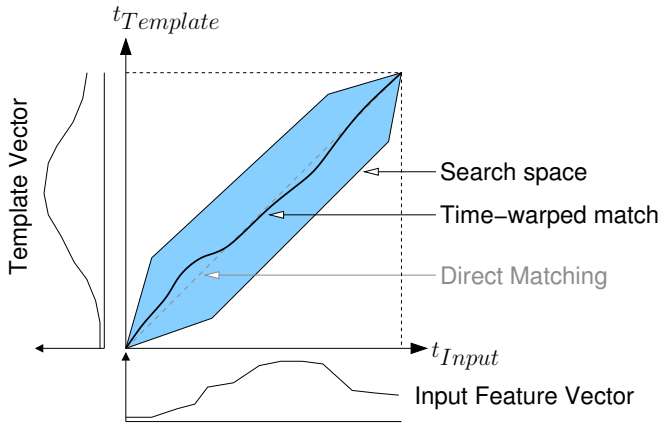


Figure 5.6: Conceptual sketch of matching an 1D input feature vector to a template using dynamic time warping.

recognition tasks like, e.g., recognizing the digits of a phone number, there are clear pauses allowing easy segmentation, normal speech is a rather continuous signal. Similarly, gesturing is a continuous process where the beginning and end of a gesture are not exactly specified. For applying time warping there is, therefore, some kind of gesture spotting required to avoid a large number of false recognitions. Again, this limitation was first tackled in the speech recognition community that turned to using probabilistic approaches for template matching, these will be considered in the Section 5.4.

5.3.3 Modeling Sequences of Atomic Gestures

While the direct comparison and the dynamic time warping outlined in previous Subsections are based directly on (preprocessed) input data, there are also deterministic approaches that are based on a successful recognition of atomic gesture elements. This essentially amounts to having already processed the sensor input with any of the other techniques for gesture recognition outlined in this Chapter. Based on the recognition of elementary gestures, the modeling of sequences can be done by realizing some kind of hierarchy, i.e., recognizing a complex gesture can be based on observing a specific sequence of elementary gestures. The sequence can be either strictly sequential as in syntactic approaches or can have more complex temporal, spatial, and logical relationships that are represented in description-based approaches.

Syntactic Approaches

One typical method for a syntactic model is a Context-Free Grammar (CFG) that represents a gesture by parsing the temporal sequence of atomic gesture elements using parsing techniques known from the field of programming languages. A gesture is therefore represented as a set of production rules generating the sequence of atomic gesture elements that represents a complex gesture. However, such a deterministic modeling with a CFG depends completely on a successful recognition of the elementary gestures, otherwise the complex gesture will not be recognized. Therefore, CFGs have mainly been applied to gesture recognition in their probabilistic variant, the stochastic CFG (SCFG, see, e.g., Ivanov and Bobick, 2000; Moore and Essa, 2002; Yamamoto et al., 2006). While the probabilistic variant reduces the reliance on correctly recognized elementary gestures, a general drawback of syntactic approaches is their strong reliance on the sequential structure of the elemental gestures.

Description-based Approaches

In order to model more complex relationships of elementary gestures, description-based approaches have been proposed (see, e.g., Hakeem et al., 2004; Ryoo and Aggarwal, 2006; Gupta et al., 2009; Ijsselmuiden and Stiefelhagen, 2010). A dominant technique for modeling temporal relations are temporal predicates as proposed by Allen (1983) that are used by Ryoo and Aggarwal (2006) to capture the temporal relationships of gestural interaction between two humans. For example, the complex gesture of pushing another human requires on the level of elementary gestures that one human p1 stretches out his arm and keeps it stretched while already touching the other human p2. Figure 5.7(a) depicts this temporal relationship graphically while the computational modeling is depicted in Fig. 5.7(b).

Although description-based approaches allow to model a variety of relationships, they are in essence a deterministic technique and rely on the correct recognition of their constituent elementary gestures. Consequently, they are typically used for high-level scene interpretation (see, e.g., Nevatia et al., 2003) while the low-level elementary gestures are often recognized by applying HMMs or DBNs which are detailed in the next Section.

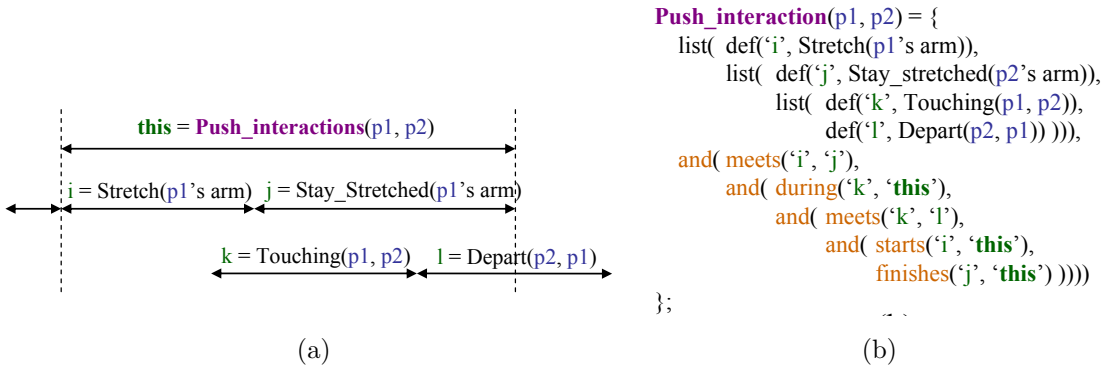


Figure 5.7: Description-based modeling of temporal relations. a) graphical illustration of the overlapping elementary gestures and the overall complex gesture; b) Formal representation of the complex gesture using a CFG-like specification (images from Ryoo and Aggarwal, 2006).

5.4 Probabilistic Approaches for Trajectory Matching

Since the start and end points of gestures are not explicitly given, it is advantageous if the classification algorithm implicitly selects the relevant parts of a trajectory for classification. Additionally, as the same gestures are usually not identically executed the classification algorithm should be able to deal with a certain variability of the trajectory. These requirements have resulted in the application of probabilistic approaches for gesture recognition that are reviewed in the following Subsections.

5.4.1 Hidden-Markov-Models for Trajectory Recognition

Based on the success of Hidden-Markov Models (HMM) for speech recognition applications (Rabiner, 1989), this technology was subsequently also used for gesture recognition (see, e.g., Campbell et al., 1996; Starner et al., 1998; Li et al., 2006; Nickel and Stiefelhagen, 2007; Axenbeck et al., 2008; Hahn et al., 2009; Gehrig et al., 2009; Droeschel et al., 2011).

While in dynamic time warping the templates are used directly for matching, the HMMs abstract from the exact trajectory. Instead, a stochastic process with a number of states S_i is assumed to be generating the observed feature vectors. The true sequence of states generating the observed feature vectors is not known, this is why the term 'hidden' is used. In other words, the true state S_i is a hidden variable and has to be estimated from an observed feature vector. Transitions between states i and j are modeled by transition probabilities a_{ij} and self-transitions with

a_{ii}. The probability to be in a certain state depends only on the previous P states, this is the Markov property and the value of P is usually one, i.e., a first-order Markov process is modeled. An example for such a first-order Markov model is depicted in the top right of Fig. 5.8.

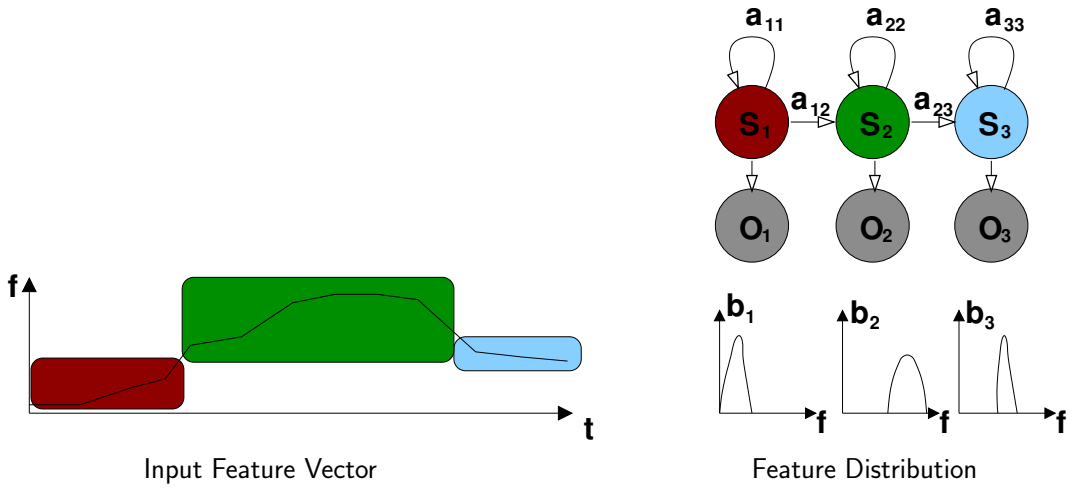


Figure 5.8: Conceptual sketch of the relation between an input feature vector to be recognized and a Hidden-Markov Model with hidden states S_i and observable features O_i.

For each state the output probability density function b_i provides for a certain feature value f the probability that this feature has been emitted by the hidden state S_i. Figure 5.8 depicts a schematic example where the rough relation between the values of the input feature vector (left) and the HMM states (right) has been indicated by color coding. Given an input feature vector, one way of performing recognition is to calculate the most likely state sequence together with its overall probability for each different HMM gesture model using the Viterbi-algorithm (Rabiner, 1989). For the task of gesture recognition, the HMM with the highest probability can be selected as recognition result if this probability exceeds a pre-defined threshold.

Although the HMM framework provides a powerful tool for probabilistic gesture recognition, there are some challenges when modeling a gesture vocabulary with a set of HMMs:

- For a set of gestures of varying length a different number of states is needed for each gesture model. This in turn influences the overall probability of the different gestures, requiring some kind of scaling of the model probabilities based on the state size of each model.

- In its original form, an HMM does not impose a minimal duration for each state. This has the effect that a model may have the highest overall probability even if some of its states have not been observed for more than one frame. While in speech processing the swallowing of some phonemes is acceptable, in gesture recognition the leaving out of a part of the trajectory may result in a very different gesture.

While HMMs have had a huge success in speech recognition, the application of standard HMMs to the complete gesture recognition task has declined in recent years. Instead, more recent approaches use HMMs to model state sequences but apply more explicit probabilistic trajectory matching techniques like, e.g., described in Section 5.4.3 for obtaining the probability of individual states (see, e.g., Li et al., 2006; Hahn et al., 2009).

5.4.2 Hierarchical HMMs and Dynamic Bayesian Networks

The HMMs described before allow the modeling of time series data. However, standard HMMs are not well suited for modeling more complex gestures or actions consisting of several individual gestures, i.e., having some kind of internal model hierarchy. For such complex gestures, the extension of standard HMMs to hierarchical HMMs (HHMM, see Fine et al., 1998) has been proposed for use in gesture recognition (see, e.g., Kawanaka et al., 2006). Besides the HHMMs for which an example is depicted in Fig. 5.9 there have also been other extensions to HMMs. For example, a coupled HMM (CHMM) was introduced by Brand et al. (1997) to model two-handed gestures. Later, this was extended to coupled hidden semi-Markov models (CHSMM) by Natarajan and Nevatia (2007) in order to capture also the duration of activities and apply this method to the recognition of two-handed American sign language.

Besides the different variants of HMMs that have been used for the recognition of complex gestures, also the use of general Dynamic Bayesian Networks (DBN) has been proposed (see, e.g., Wang and Tung, 2007; Li et al., 2007; Suk et al., 2010; Zhu and Sheng, 2011). From a mathematical viewpoint, HMMs are a special case of DBNs (Ghahramani, 2002) and a DBN has less restrictions on the number and relation of hidden variables resulting in more powerful modeling capabilities. Two examples for dynamic Bayesian networks are depicted in Fig. 5.10.

For example, in the work of Suk et al. (2010) for recognizing two-handed gestures the DBN structure is roughly similar to the example in Fig. 5.10(b) depicting two time slices. The states S_B and S_C model aspects of the individual left and right hand and the shared state S_A models the relation of the two hands. Similar to HMMs,

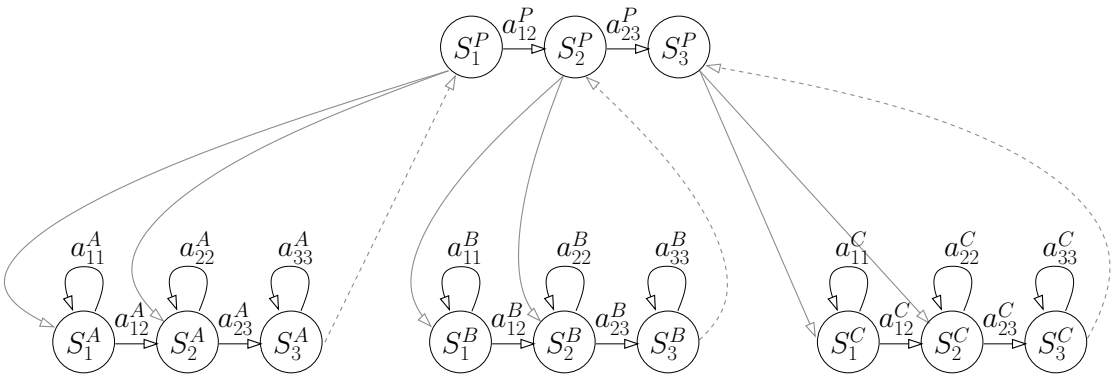


Figure 5.9: Conceptual sketch of a simple Hierarchical Hidden-Markov Model (HHMM). The gray-colored links indicate transitions from a parent state to its child states and the dashed arrow represents the completion of a child model and the transition back to the parent model. Such a completion typically leads to a state transition on the parent level.

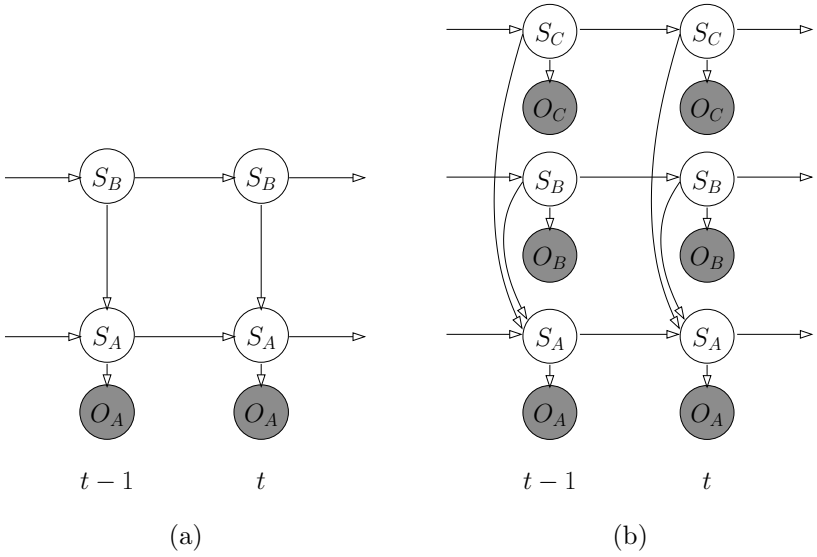


Figure 5.10: Simplified examples of dynamic Bayesian networks for two time slices $t-1$ and t. a) A hierarchical DBN where the observation O_A is generated by the state S_A which itself is dependent on the higher level state S_B; b) A DBN where the observation O_A is generated by the state S_A which depends on two other states S_B and S_C, i.e., the hidden states have a more complex relationship;

for each different gesture to be recognized an individual DBN has to be created. While for recognizing gestures with HMMs usually the fast Viterbi-algorithm is used, the inference process in DBNs depends on the complexity of the network. Often no exact inference is possible and an approximation has to be used which can be computationally expensive. Consequently, the structure of DBNs used for gesture recognition is usually of limited complexity.

5.4.3 Particle Filtering for Trajectory Recognition

Following the success of probabilistic trajectory modeling with HMMs, also other probabilistic techniques were applied to the task of motion recognition. A highly influential approach is a variant of particle filtering (see also Section 4.2.2 on page 71) called *Conditional Density Propagation* (CONDENSATION) that was introduced to the vision community by Isard and Blake (1998) for performing shape tracking. Subsequently, the CONDENSATION algorithm has been adapted for classifying commands drawn at a blackboard by Black and Jepson (1998). In this adaptation, gestures are represented by parameterized trajectory models which are matched with the input data, calling this approach CONDENSATION-based trajectory recognition (CTR).

Importantly, the CTR formulation goes beyond the modeling capabilities of HMMs by matching the entire temporal trajectory to the template model using a few explicit scaling factors. Each gesture model $\mathbf{m}$ consists of a 2-dimensional trajectory, which describes the motion of the hand in the image plane during execution of the activity.

$$\mathbf{m}^{(\mu)} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_T\}, \quad \mathbf{x}_t = (\Delta x_t, \Delta y_t) \tag{5.1}$$

For comparison of a model $\mathbf{m}^{(\mu)}$ with the observed data $\mathbf{z}_t = (\Delta x_t, \Delta y_t)$ the state vector $\mathbf{s}_t$ is used. This vector defines the sample of the activity model μ where the time index ϕ indicates the current position within the model trajectory at time t. The parameter α is used for amplitude scaling while ρ defines the scaling in time dimension. See Fig. 5.11 for a visual description of the parameters for one feature dimension.

$$\mathbf{s}_t = (\mu_t, \phi_t, \alpha_t, \rho_t) \tag{5.2}$$

By applying particle filtering (see Section 4.2.2), i.e., maintaining a large number of N state vectors $\mathbf{s}_t^{1:N}$, the best fit between a model trajectory and the observed data $\mathbf{z}_t$ can be found. The weight $\pi_t^{(n)}$ of the sample $\mathbf{s}_t^{(n)}$ is the normalized probability $p(\mathbf{z}_t|\mathbf{s}_t^n)$. This is calculated by comparing each scaled component of the model

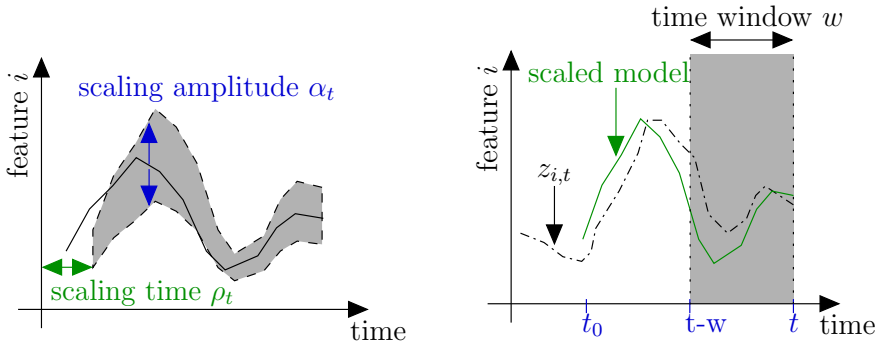


Figure 5.11: Matching observed data with the available models. a) Scaling of trajectory in amplitude and time; b) Matching of scaled trajectory over a time window w.

trajectory in a window over the last w time steps with the observed data. For calculating the difference between model and observed data a Gaussian density is assumed for each point of the model trajectory:

$$p(z_{t,i}|\mathbf{s}_t^{(n)}) = \frac{1}{\sqrt{2\pi\sigma_i}} \exp \frac{\sum_{j=t-w}^{t} \left(z_{j,i} - \alpha m_{(\phi-\rho j),i}^{(\mu)} \right)^2}{2\sigma_i^2 (w-1)}. \tag{5.3}$$

The propagation of the weighted samples over time consists of the typical three steps for particle filtering (see Section 4.2.2) and is based on the results of the previous time step:

Select: Same as in standard particle filtering.

Predict: The parameters of each sample $\mathbf{s}_t^{(n)}$ are predicted by adding Gaussian noise to α_{t-i} and ρ_{t-1} as well as to the position ϕ_{t-1} that is increased in each time step by ρ_t. If ϕ_t is larger than the length of the current model $\phi_{\max}$ a new sample $\mathbf{s}_t^{(n)}$ is initialized.

Update: Same as in standard particle filtering.

Based on the continuous matching of the feature vector to the model templates contained in the sample set, the sample set represents at each point in time probabilistically the gesture that currently has the highest likelihood. Now, the recognition of individual gestures can be achieved by summing up for each model μ_i all the weighed samples that represent a nearly completely matched trajectory to

obtain the so-called end probability $p_{\text{end}}(\mu_i)$. The minimal trajectory length L_{min} that has to be successfully matched can be expressed as fraction of the total model trajectory:

$$p_{\text{end}}(\mu_i) = \sum_{n=1}^{N} \begin{cases} \pi_t^{(n)} & \text{, if } \phi_t > L_{min}\phi_{\max} \\ 0 & \text{, else} \end{cases} \tag{5.4}$$

For example, a value $L_{min} = 0.9$ would require a sample to be within the last 10% of the trajectory model to contribute to the end probability. If the end probability for a certain gesture model exceeds a pre-defined threshold, this gesture is provided as recognition result.

5.4.4 Trajectory Matching Approaches employing Neural Networks

An alternative to the explicit modeling of probabilistic aspects is the use of neural networks. Standard neural networks are very well-suited for pattern classification like, e.g., the recognition of hand postures (see Section 3.2.3 on page 60) or the holistic recognition of gestures as outlined in Section 5.1. For the recognition of temporal signals, extensions to neural networks have to be used that allow to capture the temporal aspects (see, e.g., Yang and Ahuja, 1999; Bailador et al., 2007; Modler and Myatt, 2008; Sigalas et al., 2010).

One early approach to trajectory matching by Yang and Ahuja (1999) used a Time-Delay Neural Network (TDNN) to recognize American sign language from 2D motion trajectories. In the TDNN, a window is shifted over the input signal in order to present to the network only a part of the overall trajectory and make a local holistic classification for each input part. For the overall classification, these local decisions have to be integrated to take a final decision on the gesture.

In later approaches, other methods for handling the temporal aspects have been proposed like the Continuous Time Recurrent Neural Network (CTRNN, see, e.g., Bailador et al., 2007) or the buffering of the input data and a subsequent classification of the complete sequence with a Multi-Layer Perceptron (MLP, see, e.g., Sigalas et al., 2010).

Compared to the activities on probabilistic trajectory recognition, there are relatively few activities on employing neural recognition techniques. One reason for the few applications to trajectory recognition are the challenges in modeling the temporal aspects in neural network approaches. Consequently, neural networks are primarily applied to the detection of hand postures or some other form of prepro-

cessing of the input data that is independent of temporal aspects (see, e.g., Zhu and Sheng, 2009).

5.5 Example: Trajectory-Based Recognition of Pointing Gestures

One important type of gestures used very often in every-day communication between humans are deictic gestures that are used to reference objects (see Section 2.1.4). In this Section we will provide an example of an algorithm for recognizing pointing gestures based on the trajectory of the 'hand position' only, i.e., independent of any context information (for a detailed description see Hofemann, 2007). In addition to pointing gestures, also the recognition of waving gestures is included in the gesture repertoire, as such a waving is often used in human-human communication to attract the attention of the communication partner.

First, we give in Subsection 5.5.1 an overview of the presented system and the used modules. The features used as input for the trajectory recognition based on Particle Filtering (see Section 5.4.3) are described in Subsection 5.5.2. In Subsection 5.5.3 results of the system acquired in a demonstration scenario are presented.

5.5.1 System Overview

The overall deictic gesture recognition system is depicted in Fig. 5.12:

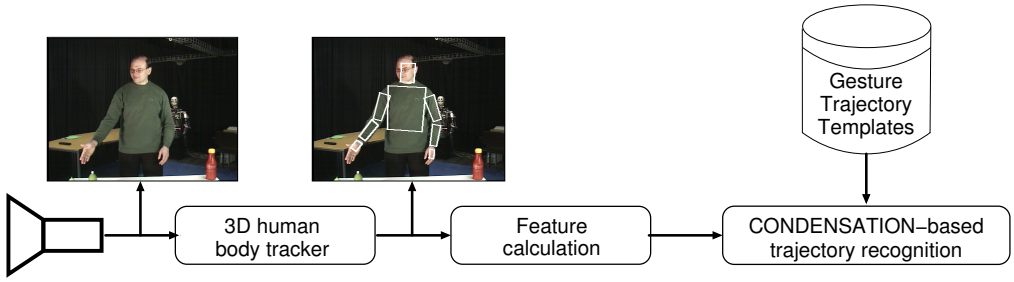


Figure 5.12: Architecture of the deictic gesture recognition system.

The first module operates directly on the image data and extracts the 3D body pose of the acting human from the video data. For this task, the example system presented in Section 4.6 on page 98 is used. From the 3D human body configuration, the 3D features can be extracted easily to serve as basis for the recognition process.

5.5.2 Recognizing Pointing Gestures

Based on the 3D hand position provided by feature extraction, the recognition of pointing and waving gestures can be done. Here, a modified version of the CONDENSATION-based trajectory recognition based on particle filtering approach is used (see Section 5.4.3). In the original approach motions were represented in an image coordinate system $(\Delta x, \Delta y)$. In this application the hand motions are represented in 3D using a cylindrical coordinate system with its basis in the human's shoulder as depicted in Fig. 5.13.

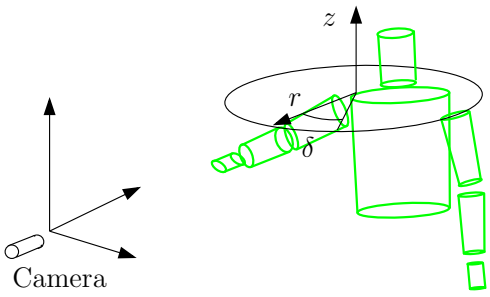


Figure 5.13: Cylindrical coordinate system with its origin located in the shoulder for feature calculation.

The motion features for representing a pointing gesture are the relative radial velocity Δr and vertical velocity $\Delta\gamma$ of the hand with respect to the torso:

$$\mathbf{m}^{(\mu)} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_T\}, \quad \mathbf{x}_t = (\Delta r_t, \Delta\gamma_t) \tag{5.5}$$

In this way the absolute direction of the gesture is removed and a wider range of gestures can be represented with one generic model. As the user typically orients himself towards the dialog partner when performing pointing and waving gestures, the used representation can be considered view-independent in such a scenario.

For the overall recognition system the repertoire consists of the elementary gesture models listed in Table 5.1. For each model a trajectory template is created and the recognition is performed as outlined in Section 5.4.3. Figure 5.14 depicts three example images from a pointing gesture with the model probabilities (bottom left) and the associated model end probabilities (bottom right). Note that the x-axis of the probability plots points to the past, i.e., the most recent probability values are inserted at $t = 0$.

Pointing	
point	hand performs pointing to an object
back	hand is retracted after pointing
Waving	
up	raising the arm to start a waving gesture
wave	waving
down	lowering the arm after waving

Table 5.1: The elementary gesture models used in this example approach.

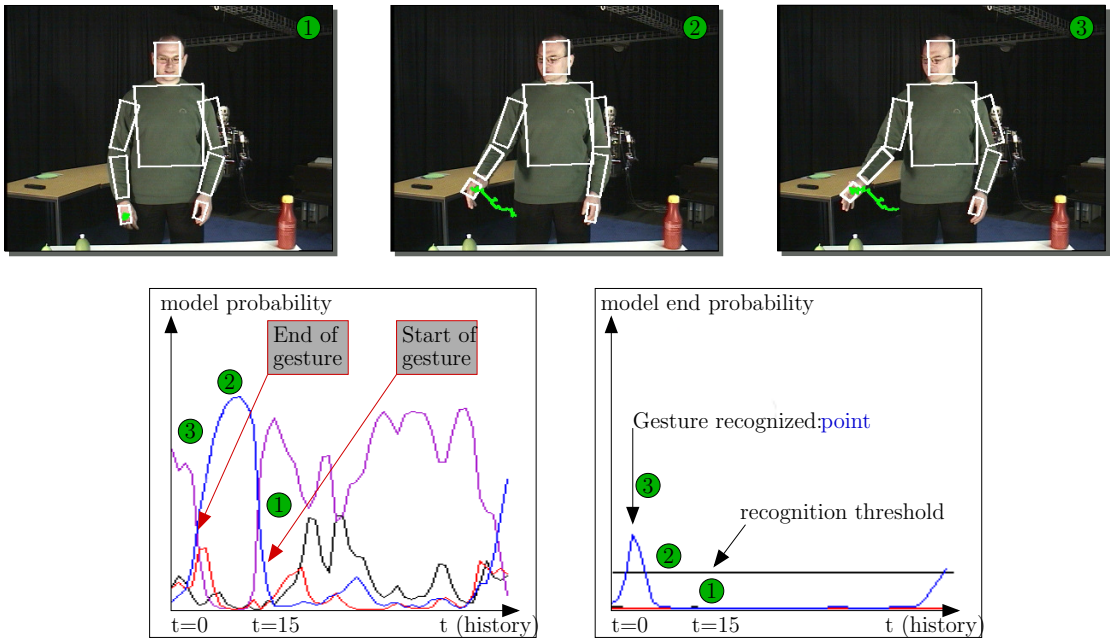


Figure 5.14: Example body tracking results and gesture recognition results for a pointing gesture.

5.5.3 Evaluation Results

The gesture recognition is evaluated using the trajectories provided by the body tracking component (see the dataset description in Section 4.6.4 on page 108). The models of the gestures are trained separately for the four subjects using the tracked and annotated motions from three of the four runs (each run consisting of sequentially pointing to the five objects on the table and waving to the camera). The remaining sequence is used for evaluation. This results in sets of only five to

20 training samples for each gesture model.

For the intended application, fast adaption to new users is an important feature but obviously prevents extensive collection of training data. The presented algorithm shows robust performance even if only a low number of training examples are provided. Note that systematic errors of the body tracking are co-learned in the training process, thus enabling robust gesture detection even for coarse tracking results.

Table 5.2 is illustrating the recognition results and error types for the five different motions. In each column the number of annotated and correctly detected gestures is noted followed by the error types of double detections, deletions, false insertions, and substitutions. Finally, the resulting error rate and recognition rate are given for each gesture model. The last column summarizes the results for all models. Subject IV executed the wave motion simply by moving the hand instead of the forearm which is not recognizable for the body tracking. The marked column (wave*) therefore displays the results neglecting subject IV for a better comparability.

gesture	point	back	up	wave*	down	all
# gestures	182	187	29	71	27	496
correct	*167*	182	29	65	26	469
double	49	102	0	0	0	151
delete	15	5	0	5	1	26
insert	24	26	1	2	6	59
substitution	0	0	0	1	0	1
WER [%]	21.4	16.6	3.5	11.3	25.9	*17.3*
recognition rate [%]	91.8	97.3	100	91.5	96.3	*94.6*

Table 5.2: Gesture detection evaluation: Errors and recognition rates for all objects and subjects.

The gesture recognition shows a high recognition rate of 94.6% and an acceptable word error rate (WER) of 17.3%. The WER is calculated by adding up all errors (deletions, insertions, and substitutions) for the respective gesture and dividing this by the total number of gestures. The false positive recognitions (insertions) make up the major part in this error (11,9% ; e=59). For a robot system, this could lead to false assumptions about the gestures performed by the user.

Comparing these results based on 3D hand position data to a similar approach using only 2D data (see Haasch et al., 2005) shows that appropriate depth information about the current body pose can significantly improve the robustness of gesture

recognition. This importance of 3D information has resulted in a huge increase in interest in 3D gesture recognition when the Kinect sensor (see Section 2.2.2 on page 35) became widely available, providing raw 3D depth data at a low cost to many researchers. The example approach that has been outlined in this Section requires 3D hand position data and could also be based on tracking algorithms using Kinect data as input data.

5.6 Summary and Conclusion

In this Chapter different algorithmic approaches for the recognition of gestures have been outlined. The holistic approach to gesture recognition is not the best choice for gesture understanding on a mobile robot as it inherently captures temporal aspects and is applicable only in limited settings: It is usually assuming a static camera to filter out the acting human from the background and has an inherent view-point dependency. In addition, the granularity of the gesture models must be chosen carefully, making this approach primarily suitable for recognizing command gestures.

Much more dominant are modular approaches for gesture recognition that are based on matching explicit trajectory models to the observed hand positions. This problem is similar in nature as the speech recognition task of matching word models to observed acoustic signals and has, therefore, benefitted from the progress in that field. Early approaches used deterministic matching methods, but these resulted in heavy restrictions on the model similarity of the different gestures to be recognized. On a more abstract level, syntactic and description-based methods for matching sequences of elementary gestures to complex models have obtained a larger interest. These methods assume that the elementary gestures have been recognized beforehand, which is typically done today with some form of probabilistic approach.

In order to provide a coarse insight into typical probabilistic approaches this Chapter has introduced the state-based matching techniques of Hidden Markov Models and Dynamic Bayesian Networks as well as the template-based matching with particle filters. A short look on the suitability of neural networks completed the overview of gesture recognition approaches.

As a prototypical example of the steps that need to be considered when implementing a gesture recognition approach, an example for the trajectory-based recognition of pointing gestures with a particle filter has been given. For this task, an appropriate choice of the feature vectors is of crucial importance to achieve

good recognition. The analysis of the recognition performance demonstrates that the quality of a gesture recognition approach should not be limited to the amount of correctly recognized gestures but also additional errors like double recognitions or insertions should be considered. While the quality of recognizing pointing gestures is already quite good employing only the visual input channel, the exactness of the extracted pointing direction depends on the quality achieved in the 3D body tracking. Obviously, the ability to identify "pointed at objects" could be further increased if verbal descriptions for size and color were available.

With the overview of the techniques provided in this Chapter we can now go on to considering the integration of context knowledge about the environment and the current task, going beyond the pure hand motion.

6 Incorporating Context for Understanding the Gesture

The previous Chapter has outlined gesture recognition techniques that are applicable if the hand trajectory alone is sufficient to recognize a gesture. This assumption holds for symbolic gestures, but for other types of gestures usually the context like, e.g., an accompanying verbal utterance or an object in the environment has to be considered as well.

For example, it is intuitively clear that deictic gestures are not performed independently of the environment but stand in a relation to the object they are referring to. Understanding deictic gestures, therefore, means not only to recognize the hand motion as *pointing* but also to determine the referenced object. Besides considering the spatial relation between the gesturing hand and surrounding objects, humans usually employ more information sources for referencing objects. For example, a verbal utterance denoting specific features of the referent may accompany the gesture, and possibly also the gaze and the overall body posture provide additional hints which object is referenced. This is termed here *User-Provided Context* and is considered in more detail in Section 6.1. An example system performing gesture understanding by enabling a robot to resolve such multi-modal object references by combining the user's pointing gestures with verbally specified information is outlined in Section 6.2.

Similar to deictic gestures, also manipulative gestures are not performed independently of the object they manipulate. However, in the case of manipulative gestures there is usually no explicitly provided context from the user. Instead, the spatial and temporal context of the gesture has to be considered for gesture understanding. The objects contained in the environment form the spatial context while the temporal setting in which a gesture is performed is providing the temporal context. These different context types are subsumed here under the term *Situational Context* and are detailed in Section 6.3. An example approach incorporating situational context for the understanding of manipulative gestures is provided in Section 6.4. A summary in Section 6.5 concludes the Chapter.

6.1 Incorporating User-Provided Context for Pointing Gestures

The capability of humans to identify what object another human is referencing to is known as *joint attention* (Emery, 2000). From the robotics perspective, joint attention describes the process that enables a robot to look at the object which the user is referring to (for a detailed discussion see Kaplan and Hafner, 2004). Only if the robot is able to focus on the object referenced by the human, it can acquire information about this object, e.g., learn the name of an object. Additionally, the robot should not only store this information in a specific knowledge base, but also efficiently access all the information gathered during interaction. This acquisition and usage of object information enables the robot later to perform tasks involving objects without assistance of the human instructor.

In order to realize such a joint attention, the recognition of a hand trajectory as 'pointing' gesture is not sufficient as the object referenced by the pointing needs to be identified. Intuitively, the pointing direction of a dynamic pointing gesture seems to be specifying the object the human is referencing. However, besides the dynamic gesture there are many more bodily informations like, e.g., the gaze of the gesturing human. In human-human communication, these cues are all combined by the recipient to infer what object the other human pointed at. However, all subtle communication cues are usually not perceivable with the sensors in today's mobile robots and only the coarse overall body posture can be used to restrict the search space.

In addition to the body posture, the verbally specified information is of major importance for understanding pointing gestures. In human-human communication this information can become quite complex like, e.g., it can include the current communication topic or some abstract knowledge of the current situation. Consider, for example, a situation where a human is saying 'I will get myself a coffee.' and performs a pointing gesture. For another human the inference that the pointing gesture references the empty cup on the table is possible even if the pointing gesture was very imprecise or the table is far away from the gesturing hand. However, a robot does not have access to the broad world knowledge allowing a human to make such kind of inferences. Similarly, the semantic content of complex utterances accompanying a gesture cannot be understood by today's robots. For example, if a human says 'First thing in the morning is to get my daily dose of caffeine.' and performs a pointing gesture to a table with several objects including one mug, the referenced object cannot be identified by the robot. Nevertheless, more simple verbal utterances can be processed by today's speech understanding methods in

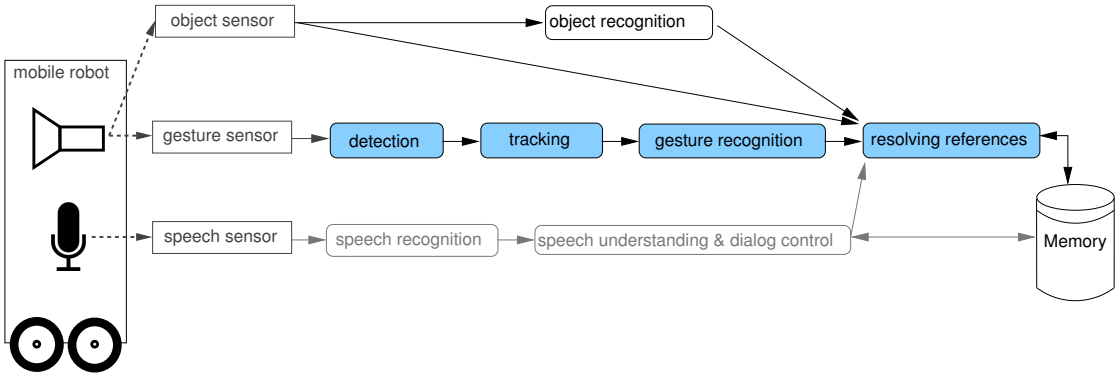


Figure 6.1: Conceptual system architecture for resolving multi-modal object references.

order to make the robot understand object references. Although the focus is here on using such input in combination with gestures, there are also attempts to use only the verbal information channel to identify objects in the scene (see, e.g., Skočaj et al., 2011; Johnson-Robertson et al., 2011).

Summarizing the restrictions of context usage by a mobile robot, the context information that can be extracted with sensors onboard a mobile robot are the coarse body posture as well as verbally specified object/location information. Figure 6.1 depicts a conceptual system architecture with the different information sources that are combined in the *resolving references* module. For actually using this context, the information provided by the user has to be processed adequately. The next Subsections will provide more details on approaches for the incorporation of user-provided context in order to enable the robot to understand pointing gestures.

6.1.1 Posture Information Restricting the Object Search Space

As mentioned in Chapter 1, not only the gesture trajectory itself 'points' to a referenced object, but also the gazing direction of the human and its overall body posture provide cues on the object of interest. The combination of the hand position with an assumed gazing direction based on the head position to draw a virtual eye-hand-object line has been found to help in identifying a referenced object (Nickel and Stiefelhagen, 2007; Droeschel et al., 2011). For the task of identifying a referenced object, the body posture can, therefore, be used to define an eye-hand line that provides a restriction of the search space where to search for the referenced object as depicted in Fig. 6.2.

While in this way the spatial area to search for an object can be restricted in a

Figure 6.2: Using body posture information to create an eye-hand line that restricts the search space for object search.

more appropriate way than based on the hand trajectory alone, the identification of the object still needs to be realized. If there is no additional verbal information available, the separation between two potentially referenced objects is impossible. Furthermore, finding an unknown object can only be realized using general cues. For example, if the color or texture of the background like a table surface is known, then an object on this table can be found by analyzing the differences in terms of color or texture. However, this requires the object to be actually visually distinctive in these visual features. If depth information is available, then the extraction of the depth profile to find objects 'popping out' from a flat surface is a more reliable cue (see, e.g., Kim et al., 2008).

6.1.2 Verbal Information Complementing Pointing Gestures

Besides using the body posture to restrict the search space, also verbal information can provide context to understand pointing gestures. Before the use of gesture recognition on mobile robots, the verbal information in the form of commands like, e.g., 'Go forward. Stop. Turn right. Go forward.' to control the robot's movements was the primary human-robot interaction interface (see Bischoff and Graefe, 1999 for an early system implementation).

With the additional availability of gesture information, the verbal information can also complement the gestural information. A pointing gesture performed in isolation without any context cannot be interpreted in terms of its meaning. For example, the pointing can indicate a direction for the robot to go to or it can

reference an object. In settings lacking an advanced speech interface, the pointing interpretation is predefined to be either a location to go to (see, e.g., Richarz et al., 2007) or to reference an object (see, e.g., Bekel et al., 2004). With a more advanced speech interface, it becomes possible to decide on the interpretation of the gesture as either a command ('Go there.') or an object reference ('This is the coffee mug.') based on verbal information. Focussing on the speech channel as the main communication channel, the pointing gesture can "fill in" the information missing in speech, like, e.g., in 'Put the bottle there.' the pointing direction provides the missing information on where to place the bottle (see, e.g., Burger et al., 2008).

Except for the gesture interpretation itself, the verbal information can also support the localization of referenced objects if the hand-eye line cannot be extracted or is not applicable. For example, Ghidary et al. (2002) use the speech interface to provide information to the object search process on what side of the referencing hand the object is positioned ('left'). Alternatively, they also allow the possibility to indicate for large objects by a verbal command ('two points') that two gestures pointing to the corners of the object are used to indicate its size. Obviously, such verbal commands complementing rather artificial pointing gestures do not resemble a natural interaction. A more prominent and powerful role of verbal information is the specification of the properties of an object as outlined next.

6.1.3 Verbal Information Specifying Object Properties

Due to the expressive power of natural language, it can basically serve to specify each and every aspect of an object referenced with a pointing gesture. However, the human communication partner will usually, i.e., in a human-human interaction, specify only those aspects relevant in the current situation. When interacting with a robot, the human makes assumptions about the understanding capabilities of the robot in order to provide appropriate additional information (see also Section 8.4).

Early fusion approaches combining gesture recognition with context from a speech interface used the symbolic nature of language for a very elementary combination. In this basic combination, the object is extracted based on generic cues like the difference to the surroundings (see Section 6.1.1) and the verbal information is just associated to this object for storage. In such a setting, the verbal information can provide the name of the object (see, e.g., Iwasawa et al., 2009; Schmüdderich et al., 2008) and also additional properties like, e.g., the size (Ghidary et al., 2002; Schmüdderich et al., 2008). It is important to stress that this is not a real use of context for gesture understanding but rather an enriching of the object information independent of the gesture.

In a more advanced use of verbal information, the human has to name the object he/she is referencing in addition to the pointing gesture in order to start the object identification process. This symbolic label is then used to select an object recognition algorithm trained on the object beforehand. Following the recognition of the pointing gesture, an object recognizer is applied on the image area the human is pointing to and the recognized object being closest to the pointing hand is selected. This setting assumes that the human uses an object name that can be associated to a pre-trained object recognition system. Besides using an explicit object label, also general object properties can be used for identifying the referenced object. Prominent cues are the color of an object and its size (see, e.g., Haasch et al., 2005; Stiefelhagen et al., 2007; Schauerte et al., 2010). This use of verbal information to complement pointing gestures for identifying referenced objects will be described in more detail in the example system in the next Section.

6.2 Example: Including Verbal Cues for Resolving Object References

In this Section an example system will be outlined combining gesture recognition with verbally specified object properties to identify an object referenced by a human instructor (for a detailed description see Haasch, 2007). For the resolution of object references it is necessary to distinguish between known and unknown objects since it cannot be assumed that all objects are known beforehand. Especially for identifying unknown objects it is necessary to process multi-modal input by incorporating gesture information and verbally specified object properties.

An overview of the system is given in Section 6.2.1 outlining the individual components. Section 6.2.2 and Section 6.2.3 will detail the individual processing steps necessary for resolving references to previously known and unknown objects, respectively. Exemplary results obtained with this approach are given in Section 6.2.4.

6.2.1 System Overview

The Object Attention System (OAS) for resolving object references achieves its functionality by combining input from several information sources. For pointing gestures the example approach outlined in Section 5.5 is assumed to perform the recognition. The user input is processed by speech understanding and dialog control and an object sensor allows to analyze the scene for potential objects. In order to retrieve more detailed visual information about objects, the object sensor is a pan-

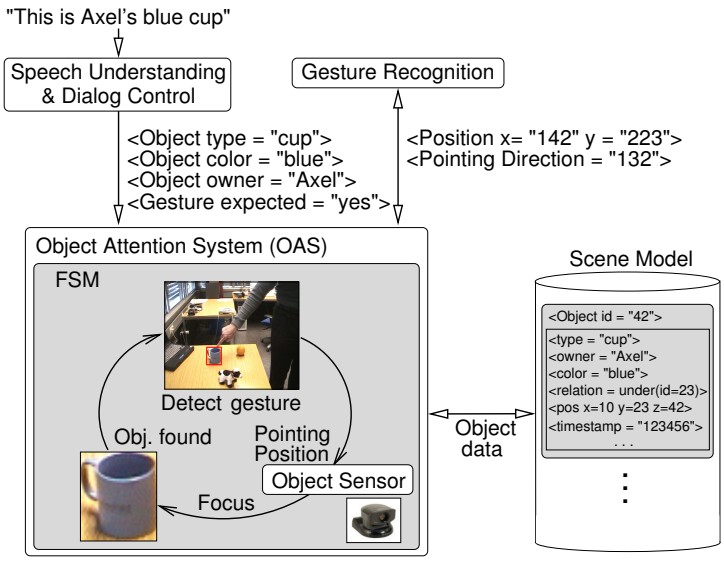


Figure 6.3: Processing chain of the object attention system including exchanged data for an example input utterance with accompanying gesture.

tilt camera that is used to focus on a referenced object, i.e., it is assumed that the object sensor is independent of the gesture recognition. Figure 6.3 depicts the conceptual system and the processing chain for such a *resolving references* module (cf. Fig. 6.1 on page 141).

The coordination of verbal information, gesture recognition, and object feature extraction (e.g., color) perceived by the camera, as well as the control of hardware components like the pan-tilt camera and the robot basis is realized by a finite state machine (FSM). This approach allows to define what conditions must be met in order for the system to change to a different processing state. Here a state represents what information is currently known by the system and/or what action the system is executing to, e.g., acquire missing information.

The OAS is activated on demand, i.e., when the user utters a phrase which contains the description of an object like "This is Axel's blue cup". At first, the FSM (see Fig. 6.4) is in the idle state, called *Object Alertness* (ObjAlert). If the OAS is provided with data from dialog control, the FSM changes to the *Input Analysis* (IA) state. Depending on a lexical cue like, e.g., "this" or "that", the speech understanding determines that a gesture is expected. As a consequence, the gesture recognition module is activated. On a successful recognition of a pointing gesture, this module supplies the user's hand coordinates and the direction of the corresponding pointing gesture. Thus, an area within the camera image can be selected as resulting ROI. In case the dialog module sends a verbally given description of

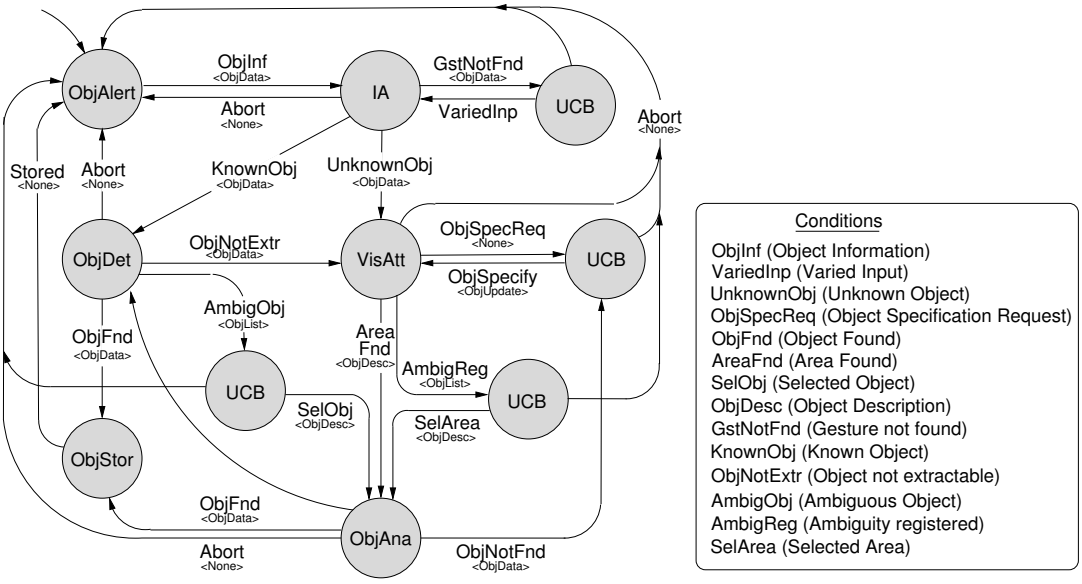


Figure 6.4: The finite state machine of the object attention system.

the object (e.g., type, color, owner, etc.) to the OAS, this information can be used for resolving the object reference.

However, it cannot be assumed that every object is known to the robot a priori. Therefore, during processing a distinction between known and unknown objects has to be made. To determine whether an object is already known, the robot's *scene model* is used. The scene model could be viewed as some kind of combination between long-term memory for object properties and short-term memory for current object positions. It provides for known objects not only visual low-level features but also appropriate object models for initializing an object recognizer. In addition, information that is given verbally by the user is stored as well (e.g., the owner of an object). Given such a scene memory, an inquiry from the OAS initiates a check whether the verbally referenced object type that is sent by the dialog component is already known or not. The next two Subsections describe the different processing steps in case the object type is known to the robot (6.2.2) or if it is unknown (6.2.3).

6.2.2 Finding Previously Known Objects

The search for a known object type involves an object detection process that is initiated by the verbally specified object type. For this task the system described here makes use of a simple appearance-based Normalized Cross-Correlation (NCC) object recognizer (Gonzalez and Woods, 2001) that is trained for a few objects

only. Any more advanced object recognition method could be used here as long as it supports searching for a specific object, but for simplicity the NCC approach is used.

After retrieving an appropriate image pattern, i.e., the object model, for the known object, the FSM switches from the IA state to the *Object Detection* (ObjDet) state. Within the ObjDet state the OAS uses the image patterns (e.g., for cups) as templates for the NCC object recognizer. The camera is reoriented based on the hand coordinates and the pointing direction that are provided by the gesture recognition module to obtain a useful view of the scene. In addition to the reorientation of the camera, a search region is determined based on the eye-hand line (see Section 6.1.1). Now, the object detection process using the NCC is initiated. Depending on the recognition result, different processing steps are performed:

- If an object is detected, a confirmation message is sent to the dialog control and the FSM switches to the *Object Store* (ObjStor) state. In this state the position of the object is stored in the scene model. Now the FSM returns to the ObjAlert state for the next object reference.

- If two or more objects of the same type are found during the detection phase in the ObjDet state, the FSM switches to the *User callback* (UCB) state. In this state, a message is sent to the dialog control to find out which specific object is meant by the user. The dialog queries the user for a more detailed object reference to resolve the ambiguity. After receiving a more detailed description, like "The left one.", the FSM switches to the *Object Analysis* (ObjAna) state. In this state a new ROI is determined based on the information from the gesture recognition and the lexical cue "left". Now the FSM returns to the ObjDet state and initiates a new search. This cycle is performed until a single object is found, or the user aborts the action within the UCB state.

- If no object is found in the ObjDet state, the FSM switches to the *Visual Attention* (VisAtt) state, that is used for the localization of unknown objects (see next Subsection).

Following the processing of the detection result, the FSM is either in the ObjAlert state for processing the next object reference or it is in the VisAtt for learning an unknown object. The latter will be described in the next Subsection.

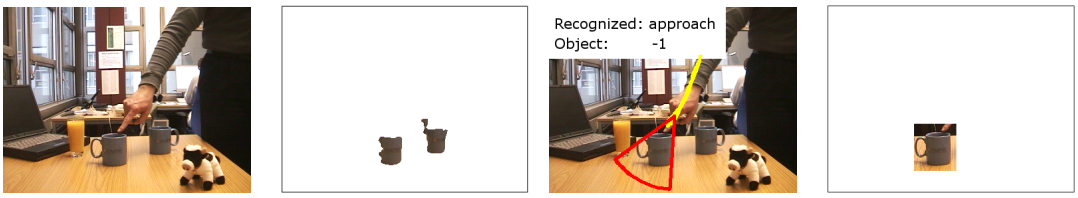
6.2.3 Learning Views of Unknown Objects

If no object detection is available, the OAS uses different visual filters – in the following called *attention maps* – in the VisAtt state to extract visual information that can be associated with verbally specified information. The attention maps are inspired by Itti et al. (1998) and bring out salient image features like distinctive colors that correspond to bottom-up image cues. The appropriate attention map is selected based on the additional verbal information (e.g., the color "blue") given by the user. As pointed out before, the eye-hand line (see Section 6.1.1) allows to restrict the object search area. By combining the object search region with the attention map, a coarse ROI depicting the referenced object can be extracted. Using this ROI, a view of the referenced object can be extracted from the scene and stored in the scene model.

If the verbal information given by the user is insufficient to determine a ROI, the FSM changes to the UCB state. In the UCB state the dialog control is requested to obtain more information about the object which the user refers to. The UCB state is also reached if more than one ROI is found. When the dialog control, after querying the user, can provide more object properties, the FSM returns to the VisAtt state. After successfully determining the ROI, the FSM switches to the ObjAna state to acquire the position of the object based on the hand position of the user. Next, the FSM switches to the ObjStor state, to store the extracted view and the position of the object in the scene model. Then, the FSM returns to the ObjAlert state to process the next object reference.

To give an example, consider the scene with the two blue cups depicted in Fig. 6.5(a) and imagine the user said "This is Axel's blue cup". In order to extract all possible ROIs, an attention map that highlights the color 'blue' corresponding to the verbal input is selected by the OAS. Consequently, two possible ROIs are extracted in the color-filtered image as shown in Fig. 6.5(b). This ambiguity is resolved by evaluating the user's gesture as depicted in Fig. 6.5(c). The yellow line in the image denotes the hand's trajectory and the circle segment marks the search area depending on the pointing direction. Based on this search area, an object is expected in the movement direction of the hand. This enables the system to set a bounding box surrounding the blue-colored region in the search area, resulting in a view of the blue cup to which the user refers to depicted in Fig. 6.5(d). In this way a new object view is identified and after creating an appropriate object recognition model it can be easily referenced in subsequent interactions.

The information acquired by the dialog control and the OAS during the interaction with a user must be stored adequately. Because the same information from dif-



(a) (b) (c) (d)

Figure 6.5: Resolving ambiguities by color and gesture evaluation.

ferent modalities requires different ways of representation, the management of such a multi-modal database is challenging. The realization of the scene model is based on the concept of an active memory (Wrede et al., 2004). For converting between symbolic and sensory information, the scene model includes a so-called *modality converter* that allows to map requests to both types of information. Modalities that are supported are color, relation, size, and shape. For example, the symbolic information 'red' received from the dialog control can be mapped to a range of color values based on the HSI color model. The scene model can, therefore, be searched for both types of entries: symbolic labels and sensory data in the form of image patches with a matching color value.

6.2.4 Evaluation Results

An example of how the verbal information can help resolve ambiguities was demonstrated in Fig. 6.5. For giving an idea of what quality can be reached using posture information to restrict the search space (see Section 6.1.1), in the following an evaluation of the object attention system based on using this modality is presented. Note that the overall evaluation of the complete interactive system is of limited value, since the task of the dialog control is to handle the different errors occurring in the different processing parts through interacting with the user. In the end, all object references in scenes of limited complexity will be successfully resolved, as the user is queried in a dialog to provide support as long as the OAS is not able to resolve the reference. Only if the visual scene is so complex that objects are partly occluding each other or their visual features are similar, an object reference may not be resolvable by the OAS in combination with a dialog. One way of evaluating such an evaluation would be to measure the interaction complexity like, e.g., the dialog length, but this would be highly dependable on the scene complexity and difficult to specify adequately. Therefore, this Subsection is restricted to evaluating

the use of posture information.

Here, the same setup is used as for evaluating body tracking (Section 4.6.4) and pointing gesture recognition (Section 5.5.3). For evaluating the object attention module in this setup (see also Fig. 4.23 on page 110) only the 167 correctly recognized pointing gestures are used.

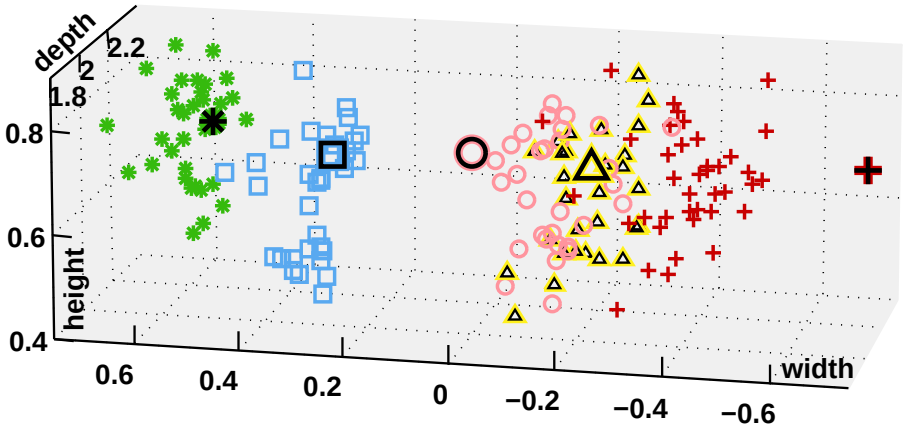


Figure 6.6: ROI center positions for recognized pointing gestures and ground truth (bold markers) for each object. Objects (1-5) from left to right.

Five of those gestures have been neglected by the object attention (cf. Table 5.2 on page 136), because their calculated ROI was outside the interaction space observable by the object camera. The final object position error is calculated as the euclidian distance in $[m]$ between the measured object position and the ROI center. Table 6.1 presents the RMSE error μ and the variance σ for the ROIs separately for each subject. The objects (1-5) are ordered in the same manner as in Fig. 6.6.

The ROI errors are for most objects not bigger than 25cm. The error in the ROI positions is mainly due to the coarse gesture information from the preceding modules. Also, estimating the pointing direction as a line from the head through the hand does not always suit well. For objects (4) and (5) to the left of the person (at the right in the image), the ROI is estimated to be too close to the human (cf. Fig. 6.6) for two reasons:

1. The body tracking cannot handle the self occlusion very well and produces more noisy results (cf. Fig. 4.24 on page 111)

2. The assumption of an eye-hand line is invalid as for these objects the pointing gesture is more a 'shooting from the hip'.

	subj A			**subj B**			**subj C**			**subj D**			**all**		
Obj	#	μ	σ	#	μ	σ	#	μ	σ	#	μ	σ	#	μ	σ
(1)	9	0.22	0.04	7	0.10	0.03	7	0.15	0.04	9	0.13	0.06	32	0.15	0.06
(2)	8	0.28	0.02	10	0.21	0.07	7	0.23	0.02	7	0.23	0.03	32	0.24	0.05
(3)	8	0.29	0.06	7	0.14	0.05	8	0.25	0.04	7	0.22	0.07	30	0.23	0.08
(4)	7	0.38	0.05	7	0.25	0.03	7	0.25	0.05	8	0.34	0.03	29	0.31	0.07
(5)	8	0.58	0.04	10	0.47	0.07	11	0.51	0.05	10	0.50	0.04	39	0.51	0.06
all	40	0.35	0.14	41	0.25	0.15	40	0.30	0.14	41	0.29	0.14	*162*	*0.30*	*0.14*

Table 6.1: Position error for the calculated region of interest (ROI) for all objects (1-5) and subjects (A-D). #: number of recognized gestures, μ: mean error, σ: standard deviation, both in $[m]$.

The results for objects (1-3) show that good results are achievable if the referenced objects are positioned at places not leading to the problems just outlined. However, for all cases where only the gesture recognition in combination with an eye-hand line based on the body posture is not sufficient, the inclusion of verbal information and a dialog interaction with the human is crucial for resolving the object reference.

6.3 Incorporating Situational Context for Manipulative Gestures

While user-provided context is explicitly generated by the pointing human to enable the understanding of object references, the production of manipulative gestures is usually not accompanied by any additional information. (For an exception to this see the modification of gestures performed when demonstrating actions to young children in Section 8.2). Nevertheless, the recognition of manipulative gestures can also benefit from incorporating contextual information by considering the overall spatial setting and the temporal aspects. In general, there are three different options for the combination of the hand motion with its spatial context:

Body-centered approach: Similar to the use of the pointing direction to narrow down the object search space (see Section 6.1.1), the body or parts of the body can serve as reference point to which surrounding objects are associated. This will be detailed in Section 6.3.1.

Object-centered approach: The contrary approach focusses on the objects in a scene and based on these reference points it monitors for each object whether

a hand is gesturing in its vicinity or performs an interaction with the object. Approaches taking this direction will be covered in Section 6.3.2.

Fusion Approaches Combining Objects and Gestures: Treating both information sources, the objects and the gestures, in an integrated fashion results in a fusion approach that is especially well suited for coping with a broader variety of manipulative gestures. This fusion can be done in a modular way combining both information sources in parallel are reviewed in Section 6.3.3. Alternatively, a holistic fusion can be realized that does not require the extraction of the individual cues but considers the whole interaction scene. Such approaches are the covered in Section 6.3.4.

In addition to the spatial context, also the temporal context, i.e., the manipulative gestures that have been performed in the time span preceding the current gesture, can be incorporated. This is typically done implicitly in the recognition approaches (see Chapter 5). Consequently, the next Subsections focus on different ways to include the spatial context in order to understand manipulative gestures which is also termed 'action recognition' in the following.

6.3.1 Body-Centered Action Recognition

A large body of work applying a body-centered use of context for action recognition deals with whole body actions in office environments. Although in these settings the human actors do make use of their hands to perform actions like, e.g,. picking up a phone or opening a cabinet, the recognition does not focus on the detailed gesture but rather infers the action from the location of the human and some low-level features like intensity changes. An early example for this type of approach is the work of Ayers and Shah (1998) where a static camera observes an office environment. Here, a person is tracked based on detecting the face and/or neck with a simple skin color model. The way in which a person interacts with an object is defined in terms of intensity changes within the object's image area which is specified beforehand. By relating the tracked person to detected intensity changes in object areas and using a finite state model defining possible action sequences, the action recognition is performed. However, as such approaches use fixed camera positions with pre-specified object regions for action recognition, they are not applicable for mobile robots.

In order to enable the incorporation of context on a mobile robot, a body-centered action recognition needs to use the human body as reference point. Focussing on hand gestures, a logical choice is to use the hand as reference point. While for

pointing gestures a fixed eye-hand line was used (see Section 6.1.1) to search for the referenced object, manipulative gestures require a more flexible context handling. For example, while a mobile phone may be picked up from straight above, the handle of a coffee mug is usually grasped from the side. Figure 6.7 depicts a possible body-centered context search area where the size and orientation of the search area can vary for different actions (Fritsch et al., 2004).

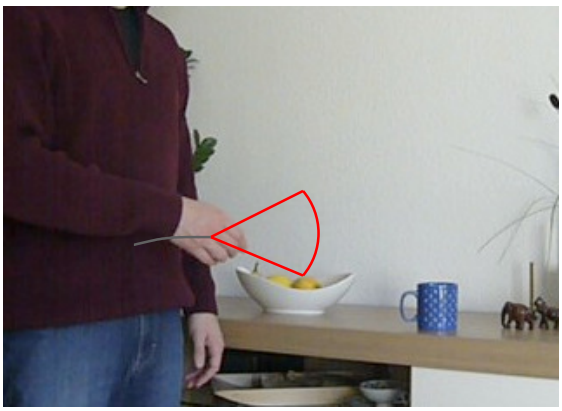


Figure 6.7: A body-centered context search area for action recognition.

Note that such a parametric modeling of the search area allows for a high flexibility of where to expect manipulated objects compared to pointing gestures where the search area was fixed. A drawback of such a body-centered approach is the necessity to specify for each action an appropriate search area, resulting in a large number of search areas to analyze if the hand trajectory itself does not exhibit a specific characteristic allowing to narrow down the potential actions. Moreover, if an action does not have any restrictions on the search area like, e.g., 'Pushing a button', then there is no need for a body-centered context definition. Consequently, body-centered action recognition is primarily useful for manipulative gestures that exhibit distinctive trajectories and approach objects in a characteristic way.

6.3.2 Object-Centered Action Recognition

Early work on object-centered action recognition by Moore et al. (1999) focussed on recognizing actions in office and kitchen environments in order to support the location and classification of objects. In the experiment setting a camera mounted at the ceiling allows to track the position of a human (head) and the acting hands based on skin color. Only if a tracked hand enters the vicinity of a potential object, the trajectory of the hand is analyzed with Hidden-Markov-Models (see

Section 5.4.1) trained offline. Note that these action models are bound to specific object types, and therefore provide *action-based* evidence for an unknown object. Other image processing steps are carried out to obtain *image-based* and *object-based* evidences for objects. All of these cues are fused in a Bayesian framework to recognize unknown objects, i.e., the gesture information is used primarily as an additional cue for object recognition.

More recent work by Li et al. (2007) aims at object-centered action recognition with a camera setting similar to the images available on a mobile robot. In such a setting, the hand trajectory is much more informative than when looking at the scene from above. However, at the same time the view-point dependency is stronger. They apply a more advanced probabilistic modeling using a dynamic Bayesian network (see Section 5.4.2) for each object and explicitly consider the camera view point in the recognition framework. Relying on an object-centered approach, the hand trajectory is analyzed with a particle filter (see Section 5.4.3) only if the hand is close to an object. Figure 6.8 depicts how such an object-centered context search area looks like. Different from Moore et al., this object-centered action recognition approach provides task recognition results, i.e., the manipulative actions that have been performed by the gesturing human.

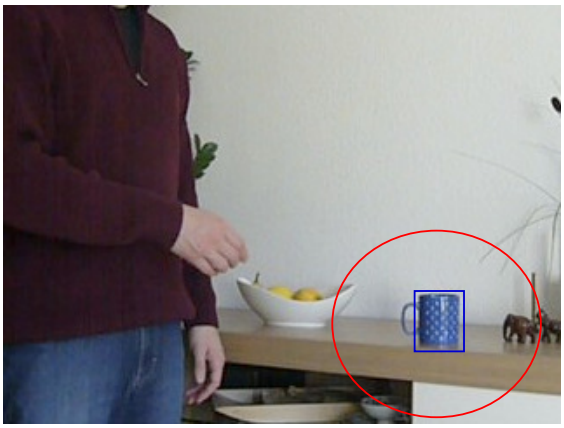


Figure 6.8: An object-centered context search area for action recognition.

A related approach was realized by Hahn et al. (2009) where the distance between known objects and the gesturing hand was analyzed by a polynomial classifier to recognize 'working actions' in an industrial environment. Together with a polynomial classifier to detect the 'transfer' between two working actions, this information of elementary actions was used as input for a Hidden-Markov-Model to recognize action sequences. An important aspect in this work is the possibility to differentiate

between known actions, i.e., the worker following the default sequence of actions, and unknown actions like, e.g., the worker scratching his head.

While the object-centered action recognition is intuitively appealing as different objects allow different manipulative gestures, an unsolved issue is the increase in complexity if the number of objects and trajectory models grows. Nevertheless, for restricted domains containing a controllable number of objects like, e.g., industrial environments, an object-centered approach offers a suitable way of combining object and gesture information to recognize manipulative actions.

6.3.3 Parallel Approaches Combining Objects and Gestures

One of the first approaches exploiting hand motions and objects in parallel is the work of Kuniyoshi and Inoue (1993) on qualitative recognition of assembly actions in a blocks world domain. This approach features an action model capturing the hand motion as well as an environment model representing the object context. The two models are related to each other by a hierarchical parallel automata that performs the recognition of manipulative gestures. This type of integration is hand-coded and specific to the task at hand, i.e., the parallel automata that processes the action and environment information to recognize assembly actions may not be applicable to other types of gestures. In addition, due to the limited vision capabilities at the time, the hand motion is not modeled by a detailed trajectory but only with the gross motion and the approaching/departing relation to an object. This restriction may be sufficient for block world assembly actions, but the motion information becomes crucial if there is a larger number of different objects that can be manipulated with several different gestures.

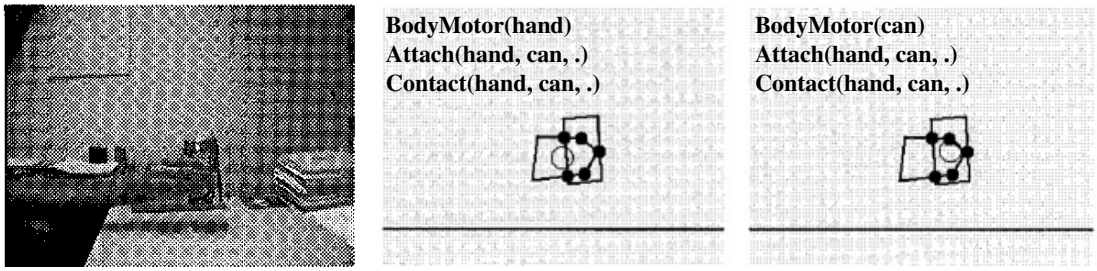
Another early approach tackling the understanding of manipulative gestures from a purely symbolic perspective is the work of Mann et al. (1997). In a roughly similar task domain as the blocks world, items on a table are manipulated (see Fig. 6.9(a)). After applying a simplified image processing to track all objects and hands in the scene, kinematic and dynamic properties are asserted to the individual objects. Objects in this approach are either hands or items on the table and all are treated in a unified computational theory as objects with different properties. For example, an object can be "attached" to another object, in "contact" with another object, or it can be a "BodyMotor" with self-induced movement (see Fig. 6.9(b)). This logical analysis of object relations results in a large number of possible interpretations where the one matching best has to be selected. The system therefore generates for each frame the most likely interpretation, but does not consider the hand motion, i.e., the temporal context. In addition, a major drawback is the huge increase in

possible interpretations for more complex scenes which is a typical challenge for approaches following the artificial intelligence paradigm.

A more recent approach modeling the actions with a description-based approach (see Section 5.3) is the work of Ryoo and Aggarwal (2007). Based on a rough segmentation of the scene, independent "primitive" processes are applied to extract the object type and object motion (see Fig. 6.10(a)). The final decision for each of the two types of information is taken by applying a naive Bayesian classifier that incorporates context information from the other information type and also top-down feedback from the overall semantic analysis. The integrated semantic analysis is performed using some form of grammar for action descriptions that has to be defined a priori using expert knowledge. In this grammar also hierarchical relations can be captured and an extension to handle temporal processes improves the applicability to real-world settings with varying temporal relationships. With these measures, the number of possible scene interpretations stays feasible compared to the generic approach of Mann et al. (see Fig. 6.10(a)).

While the logic-based and description-based approaches are appealing as they are more intuitive to understand, the modeling of manipulative gestures in a parallel non-symbolic approach incorporating objects and gestures is gaining interest recently (Gupta and Davis, 2007; Kjellström et al., 2008; Fleischer et al., 2009; Gehrig et al., 2011).

Such an approach that assumes a "flat" action structure without hierarchical relations is proposed by Gupta and Davis (2007). Here, a unified inference process

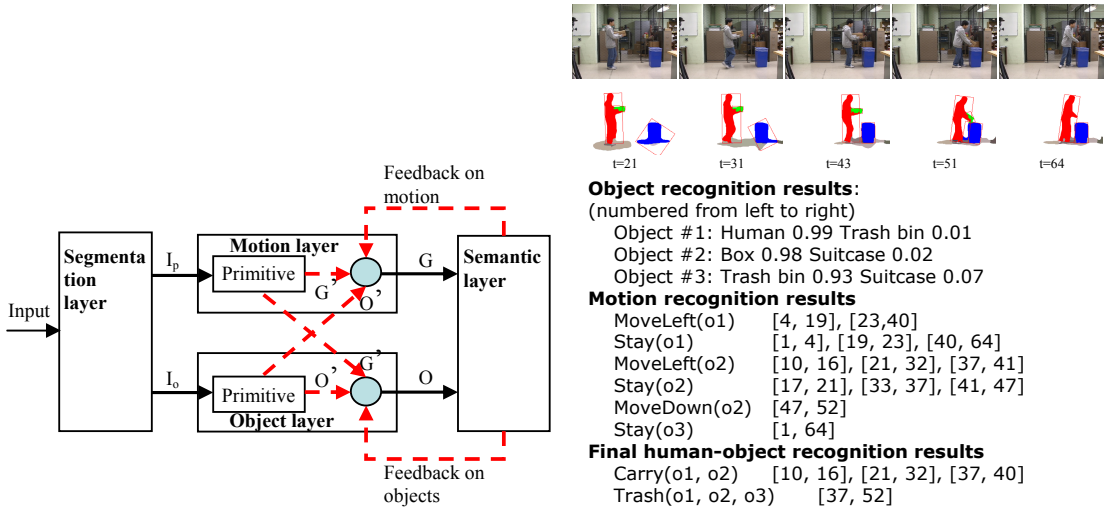


(a) (b)

Figure 6.9: (a) Input image and (b) visualization of two possible interpretations from the symbolic approach of Mann et al. (1997). A large open circle at the object center denotes a "BodyMotor"; the large disks at the vertices of the polygons denote "attached" objects; the small disks at the vertices of the polygons denote "contacting" objects. (images from Mann et al., 1997).

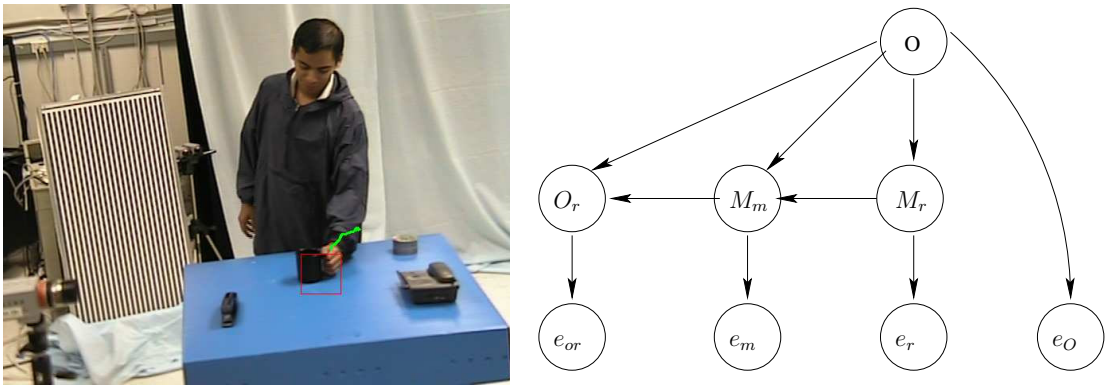
is realized that integrates object recognition and localization, action understanding, and perception of object reaction in a graphical model (see Fig. 6.11). Object evidence is acquired using standard object recognition methods and evidence for actions is obtained by applying HMMs to 2D hand tracking data. Through enforcing in the graphical model a global coherence between the different information sources this approach is able to realize a manipulative gesture understanding while improving the recognition performance of the individual perception processes.

Following a similar line of argument, Kjellström et al. (2008) propose the use of connected hierarchic conditional random fields (CHCRF) to model manipulative gestures. Here, the basic evidence is obtained by extracting hand position and hand shape as well as objects using shape descriptors. The object feature extraction is performed only in image regions near acting hands, i.e., essentially applying a hand-centered analysis similar to Section 6.3.1. However, for the actual manipulative gesture recognition, both feature types are used in parallel. Instead of applying a temporal model like, e.g., a HMM to the hand motion data, the time dimension is considered implicitly by creating feature vectors that contain the feature values for a certain time range. For both objects and actions a hierarchical conditional random field models the probability for a specific object and action. By connecting these outputs, the integration of both evidences is performed on the sequence level,



(a) (b)

Figure 6.10: (a) Parallel approach fusing motion and object information; (b) example segmentation results and the different levels of the semantic analysis (images from Ryoo and Aggarwal, 2007).



(a) (b)

Figure 6.11: (a) Image with motion and object information; (b) A graphical model incorporating evidences for objects (e_o), motions reaching to objects (e_{or}), manipulations (e_m) and object reactions (e_r) (images from Gupta and Davis, 2007).

i.e., not in every individual frame as in the previously outlined approaches.

The diversity of the outlined parallel approaches already shows that there is no single approach that can cope with all challenges in a superior way. Not surprisingly, recent approaches are characterized by incorporating probabilistic modeling techniques into the fusion process. This is sufficient for recognizing simple manipulative gestures and only for more complex actions a semantic modeling of temporal and hierarchical relations is needed.

6.3.4 Holistic Approaches to Action Recognition

Besides the explicit combination of hand motion and context information in a modular approach, also the implicit combination in a holistic approach can be used for action recognition. From the methods for holistic gesture recognition outlined in Section 5.1, those approaches incorporating not only the human motion but the overall scene are applicable to gesture understanding. Extending such holistic approaches to take advantage of the information contained in the scene is gaining interest recently (see, e.g., Ullah et al., 2010; Yu et al., 2010; Wuo et al., 2011).

Focussing on actions in movie scenes, the use of local spatio-temporal features in a bag-of-features approach (see Section 5.1) has been extended by Ullah et al. (2010) to handle non-local cues. Instead of separating the image sequences in fixed spatio-temporal grids for calculating the codeword histograms, different types of processing

are proposed to create the spatio-temporal separation. For example, in Fig. 6.12 a foreground-background region segmentation (bottom left) is used to separate the spatio-temporal features into two groups. Ullah et al. investigate different types of such semantic separations. The test data set is based on Hollywood movies and contains a wide range of actions. Only few of the actions actually involve the manipulation of objects (*AnswerPhone, DriveCar, Eat, GetOutCar*) as this was not in the focus of the investigation. Nevertheless, the semantic separation has been found to deliver an improved action recognition performance for nearly all actions compared to the standard grid, indicating the potential of this approach.

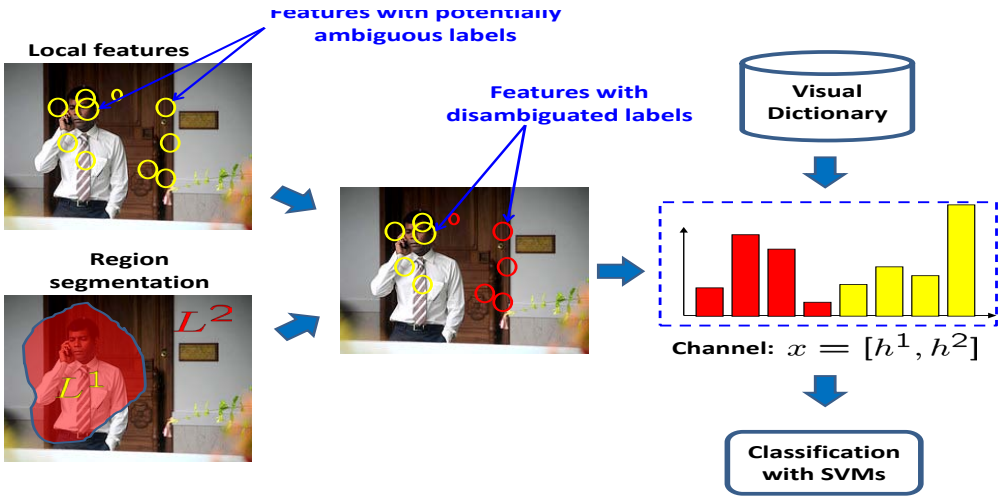


Figure 6.12: Spatio-temporal feature points separated into two groups based on region segmentation and schematic image of the associated recognition concept (image from Ullah et al., 2010).

The research on holistic action recognition is quite young, but first results are promising given the wide variety of settings like, e.g, in the movie scenes considered by Ullah et al. (2010). Through choosing the semantic region separation appropriately, this method may be able to reach sufficient recognition rates in the future. However, due to the holistic modeling, this type of approach will enable a robot only to understand what action was performed, but no detailed information can be obtained. This type of approach is therefore not suited if the robot is not only intended to understand the action but also to extract action details or to imitate the action.

6.4 Example: A Fusion Approach for Recognizing Manipulative Actions

In this Section an example of a fusion approach for the recognition of actions and action sequences is outlined (for a detailed description see Li, 2008). The spatial context considered in this example is body-centered (see Section 6.3.1) by considering objects that have a spatial relation to the acting hand. While for pointing gestures this spatial relationship is fixed (see the eye-hand line in Fig. 6.2 on page 142), the example algorithm incorporates the varying locations of different manipulative actions in the gesture modeling.

Incorporating temporal context for the recognition of manipulative actions is beneficial for the recognition process as a sequence of actions usually has some overall goal. For example, the sequential manipulative actions "take a pencil" and "move it to a notebook" should not only be recognized as two independent manipulative actions but also as an action sequence with the underlying human intention "to write". Recognizing such sequences of manipulative actions by incorporating the temporal context is demonstrated in this example with a probabilistic hierarchical approach. The probabilistic formulation makes it possible to hypothesize associated intentions that are often inferred naturally by human observers.

Section 6.4.1 will provide an overview of the system with the hierarchical manipulation model, the observation data, and the matching of observation data to the modeled actions. The process for inferring the intention during an ongoing sequence of human gestures is done by a particle filter and is described in Section 6.4.2. The experimental office environment for evaluating this approach and the obtained results are presented and analyzed in Section 6.4.3.

6.4.1 System Overview

The approach described here aims at recognizing manipulative hand gestures at different hierarchical abstraction levels. The chosen scenario is an office environment where a person sitting behind a table manipulates the objects on the table. The manipulative action sequences consisting of several individual actions are: *prepare tea*, *prepare coffee*, and *water plant*. The input to the system are 2D motion data of the hand (e.g., by detecting and tracking skin-colored regions as outlined in Section 4.4 on page 81) and object labels from object recognition. Figure 6.13 depicts an overview of the system architecture as well as the office setting (top left).

The starting point of recognizing action sequences are the individual actions. The manipulative primitives on the basic level are recognized by matching both hand

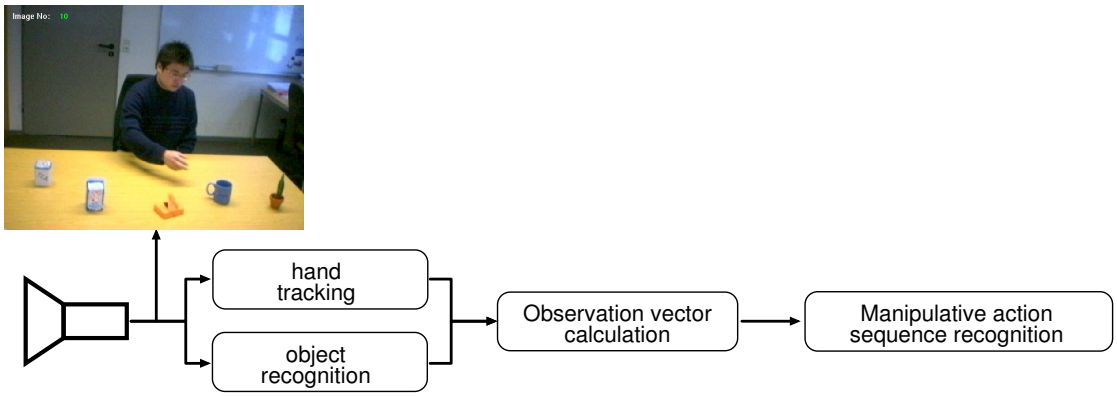


Figure 6.13: System architecture of the manipulative action recognition system.

trajectories and object context based on a particle filtering algorithm for trajectory recognition (see Section 5.4.3). This context-based algorithm for recognizing 'primitive' gestures is embedded into a hierarchical hidden Markov model (HHMM, see Section 5.4.2) which models the hierarchical structures in the manipulation of objects. Assuming a first order Markov process, a two time slices dynamic Bayesian network (DBN) with four layers can be used to model manipulative actions as depicted in Fig. 6.14. By combining such a lattice HHMM with a particle filtering algorithm it is possible to infer high-level intentions in manipulative actions by applying probabilistic processing strategies to low-level trajectory and object data.

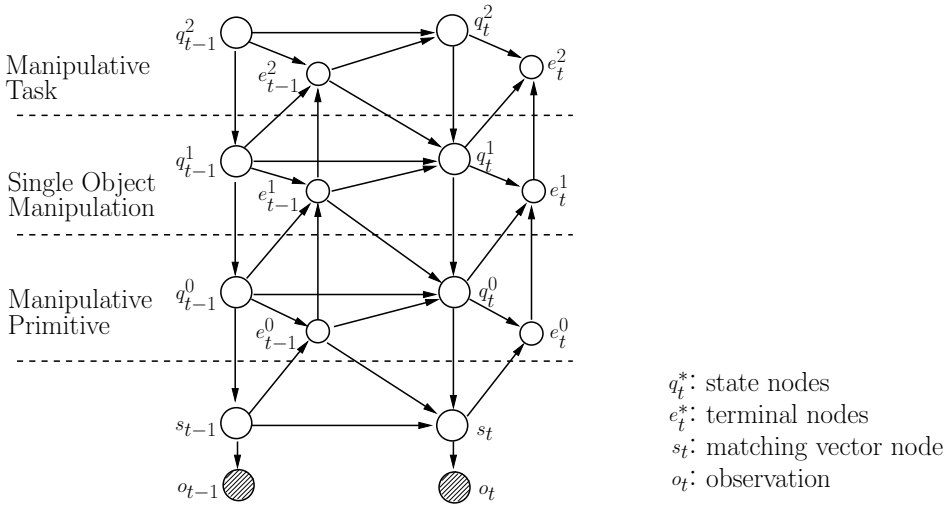


Figure 6.14: The DBN representation of the hierarchical structure for recognizing manipulative action sequences.

In the hierarchical manipulation model the shaded nodes at the bottom denote the observations o_t. The hidden states in the different hierarchical layers are represented by blank nodes. The arcs indicate the dependencies between the nodes. In the following Subsections, we will introduce the observations, the parameters needed for matching primitive motions, and the parameters for the hierarchical manipulation structure.

Observation of Hands and Objects

The observation o_t of manipulative gestures consists of two parts. The first part represents the hand and its motion. The hand motion observation vector o^{hand} includes hand position (h_x, h_y), hand velocity Δr, and change of direction $\Delta\gamma$:

$$o^{\text{hand}} = (h_x, h_y, \Delta r, \Delta\gamma) \tag{6.1}$$

Using Δr and $\Delta\gamma$ allows to represent the motion taking place in different directions and places in the image with one general model. This limits somewhat the constraints of a fixed camera position, but an appropriate view angle is still needed throughout the observation because of the lack of depth information in the 2D images.

The second part necessary for recognizing manipulative gestures is the object observation vector which consists of the information of the objects in the vicinity of the acting hand or currently in the hand. Assuming successful object recognition, the observation vector for an object contains its position (o_x, o_y), the distance r between the object and the operative hand as well as the angle $\Delta\psi$ relative to the direction of the hand motion. This vector is used during recognition to judge whether it is a relevant object in manipulation. In addition to these position-related features, the value of ID contains a unique identifier for each different object type in the scene. To represent the current contents of the hand, the observation vector also contains the hand content state o_{hc} that indicates whether the hand is empty ($o_{\text{hc}} = 0$) or contains this object ($o_{\text{hc}} = 1$). In order to assign the hand state correctly, a temporal reasoning has to be performed. Since a scene can contain several objects, the overall object observation vector is:

$$o^{\text{obj}} = \{o_1^{\text{obj}}, ..., o_i^{\text{obj}}, ..., o_L^{\text{obj}}\} \quad \text{with} \quad o_i^{\text{obj}} = (o_x, o_y, r, \Delta\psi, \text{ID}, o_{\text{hc}}) \tag{6.2}$$

Combining Eq. 6.1 and Eq. 6.2, the observation vector for each time step is:

$$\mathbf{o}_t = (o_t^{\text{hand}}, o_t^{\text{obj}}) \tag{6.3}$$

Note that in this model it is assumed that the acting hand can not hold more than one object at any time.

Matching Observations to Primitive Models

The function of the matching vector node s_t in Fig. 6.14 is the matching of the observation $\mathbf{o}_t$ with the manipulative primitive model q_t^0. The superscript 0 indicates that this is the most elementary discrete state. The manipulative primitives are typical hand motions in more complex manipulations. For example, 'touch a cup' consists of the primitives 'approach the cup' and 'withdraw the hand'. A manipulative primitive μ is modeled by a trajectory $q^{0(\mu)}$ of length T:

$$q^{0(\mu)} = \{(\mathbf{x}_0, \mathbf{c}_0) \dots (\mathbf{x}_t, \mathbf{c}_t) \dots (\mathbf{x}_T, \mathbf{c}_T)\}. \tag{6.4}$$

Extending the particle filtering approach for trajectory recognition (see Section 5.4.3) with context information, the model contains for each time step not only the motion vector $\mathbf{x}_t = (\Delta r_t, \Delta\gamma_t)$ of the hand but also the context vector $\mathbf{c}_t$ defining a search area and an object expectation for the (manipulated) object. This body-centered context search area is shown in Fig. 6.15 and consists of a search radius c_r and a direction range, limited by a start and end angle (c_α, c_β). This search area is compared with the object observation vector o_t^{obj} to judge whether an object is in the context area.

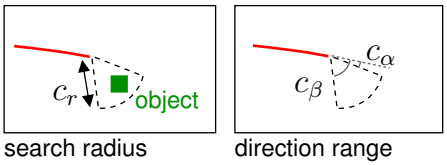


Figure 6.15: Object context

The object expectation (c_s, c_{hc}) has two parameters, c_s is used to hold the ID of the object expected in the search area and c_{hc} is the expected hand content. Thus, the complete context vector is:

$$\mathbf{c} = (c_r, c_\alpha, c_\beta, c_s, c_{\text{hc}}). \tag{6.5}$$

For comparison of the trajectory points $\mathbf{x}_i$ of a manipulative primitive $q^{0(\mu)}$ with the observed hand motion, the trajectory matching state $\mathbf{s}_t$ is used (see Section 5.4.3 on page 130 for details):

$$\mathbf{s}_t = (\mu_t, \phi_t, \alpha_t, \rho_t). \tag{6.6}$$

The comparison of the context vector $\mathbf{c}_i$ with the detected objects is part of the inferring process described in Section 6.4.2.

Hierarchical Manipulation Structure

Following the modeling of manipulative primitives in the bottom layer of the hierarchical model, this Subsection deals with the two other layers. The middle layer is the single object manipulation layer. Assuming that only one object can be the subject of a manipulation, manipulating another object must take place after manipulating the current object. This information is used to segment the whole observation sequence into manipulations of different objects. The top level is the manipulative task level. The states of this level are predefined intentions which are extracted from the sequence of single object manipulations. For example, the sequence of 'take a cup' and 'put tea into the cup' could be defined as the intention 'preparing tea'.

The model uses q_t^0, q_t^1, and q_t^2 to represent the state of the manipulative primitive, the single object manipulation, and the manipulative task at time t. They are called state nodes. On each level d, there is a terminal node e_t^d linking to the state node q_t^d. It is a binary node representing whether the current state ends at time t. If $e_t^d = 0$, the state q_t^d will continue. Alternatively, if $e_t^d = 1$, the state q_t^d ends here and a new state has to be initialized in the next time step. A terminal node equals one only when the value of the terminal node of the lower level is one because a state will not end when its sub-state is still in execution. Consequently, the states of the end nodes are inferred in a bottom-up sequence.

6.4.2 Inferring Process

The probability $Pr\{q_t^2|o_{1:t}\}$ of the top node state given the observation sequence represents the likelihood of the current manipulative task in execution. In order to achieve an online state estimation of different levels in the hierarchical representation, the concept of a belief state is applied, which is the joint distribution of all the current variables given the observation sequence $Pr\{q_t^{0:2}, e_t^{0:2}, s_t|o_{1:t}\}$. From this, the probability of the current action can be obtained by marginalization.

Updating the belief state is realized by a standard inferring process applying a particle filter. Here, an individual particle $\mathbf{b}_t^{(i)}$ is more complex as in previous formulations (see Section 4.2.2) as it includes not only the trajectory matching state $\mathbf{s}_t^{(i)}$ but also all hidden states:

$$\mathbf{b}_t^{(i)} = \{q_t^{0:2(i)}, e_t^{0:2(i)}, \mathbf{s}_t^{(i)}\}, \tag{6.7}$$

Its weight $\pi_t^{(i)}$ is the normalized probability of $Pr(\mathbf{b}_t^{(i)}|\mathbf{o}_{1:t})$. According to Bayes rule and the dependencies in the graphical model, the probability $Pr\{\mathbf{b}_t^{(i)}|\mathbf{o}_{1:t}\}$ has

the following factorization:

$$\begin{aligned} & Pr\{\mathbf{b}_t^{(i)}|\mathbf{o}_{1:t}\} \\ =& Pr\{\mathbf{o}_t|\mathbf{s}_t^i\}Pr\{e_t^{0:2(i)}|q_t^{0:2(i)},\mathbf{s}_t^{(i)}\} \\ & \int_j Pr\{q_t^{0:2(i)},\mathbf{s}_t^{(i)}|\mathbf{b}_{t-1}^{(j)}\}Pr\{\mathbf{b}_{t-1}^{(j)}|\mathbf{o}_{1:t-1}\} \end{aligned} \tag{6.8}$$

The first term of the right hand side of Eq. 6.8 is the probability of a matching between the state in the primitive model and the observation. Because of the two parts in the observation, hand trajectory and object context, $Pr\{\mathbf{o}_t|\mathbf{s}_t^i\}$ is calculated as:

$$Pr\{\mathbf{o}_t|\mathbf{s}_t^{(i)}\} = Pr\{o_t^{\text{hand}}|\mathbf{x}_t^{(i)}\}Pr\{\mathbf{o}_t^{\text{obj}}|\mathbf{c}_t^{(i)}\}. \tag{6.9}$$

The probability $Pr\{o_t^{\text{hand}}|\mathbf{x}_t^{(i)}\}$ represents the similarity of the trajectories and is calculated as described in Section 5.4.3 on page 130. The value $Pr\{o_t^{\text{obj}}|\mathbf{c}_t^{(i)}\}$ represents how good the observed object context fits the expectation. If the expected object is present the value $Pr\{o_t^{\text{obj}}|\mathbf{c}_t^{(i)}\} = 1.0$ is used and if the context area does not contain the correct object a smaller value $Pr\{o_t^{\text{obj}}|\mathbf{c}_t^{(i)}\} = 0.5$ is used. This leads to smaller weights $\pi_t^{(i)}$ of samples with a missing context so that these samples are selected less often in the resample procedure of particle filtering.

Recursively estimating the belief state of the HHMM is done with the typical three steps of particle filtering:

Select: Same as in standard particle filtering.

Predict: The states of the state nodes $q_t^{0:2(i)}$ of particle i with $i \in 1:N$ are predicted with a top-down sequence and sampled based on conditional probabilities $q_t^{d(i)} \propto Pr\{q_t^d|\hat{q}_{t-1}^{d(i)}, q_t^{d+1(i)}, \hat{e}_{t-1}^{d:d+1(i)}\}$. After the states of the state nodes are obtained, the state of the matching vector $s_t^{(i)}$ is sampled. When $\hat{e}_{t-1}^{0(i)} = 1$ which means the manipulative primitive ended at the last time step, the position indicator $\phi_t^{(i)}$ will be reinitialized as 0, which means the matching point is set to the beginning of the trajectory model of the next manipulative primitive. The scaling parameters $\alpha_t^{(i)}$ and $\rho_t^{(i)}$ are sampled randomly within a predefined limitation (see Section 6.4.3). When $\hat{e}_{t-1}^{0(i)} = 0$ which means the manipulative primitive continues, the position indicator goes one step further along the primitive model: $\phi_t^{(i)} = \hat{\phi}_{t-1}^{(i)} + (1 * \hat{\rho}^{(i)})$. The scaling parameters $\alpha_t^{(i)}$ and $\rho_t^{(i)}$ are predicted by adding Gaussian noise as prediction variance to their old values. When the states of the parameters in $s_t^{(i)}$ are obtained, the position indicator $\phi_t^{(i)}$ is used to determine $e_t^{0(i)}$, i.e., if the trajectory is nearly completed then $e_t^{0(i)} = 1$ and otherwise $e_t^{0(i)} = 0$. Then the states of

the other terminal nodes are predicted bottom-up and sampled conditioned on the current states of the state nodes and lower level terminal nodes: $e_t^{d(i)} \propto Pr\{e_t^d | q_t^{d-1:d(i)}, e_t^{d-1(i)}\}$, $d = 1, 2$.

Update: Same as in standard particle filtering.

Besides continuously estimating the belief state of the HHMM, the obtained probability values are also analyzed to determine which manipulative gestures have been performed and what object was involved in the gesture.

The classification of movements is achieved by calculating the *model end probability* $P_{\text{end},t}(q_t^d)$ as described by Eq. 5.4 on page 132. After recognizing a state, the objects that have been manipulated are determined by analyzing the context information in all samples that were used for calculating $P_{\text{end},t}(q_t^d)$. This is especially relevant for the single object manipulation and the manipulative primitive level.

On the single object manipulation level, there could be two objects appearing in the context, either in the hand or in the search area. Therefore, a main manipulative object MMO is defined that holds the ID of the expected object in the search area if there is no object in the hand. Otherwise, the object in the hand is the main object. The binary binding between a sample and an object does not provide information about the quality of the binding, but the sample weight represents how good the sample matches the observed gesture. Therefore, the *object probability* $P_{\text{obj},t}$ is calculated for every object o_i^{obj} based on the weights of the samples belonging to the single object manipulation model q^1 and containing this object in the context:

$$P_{\text{obj},t}(o_i^{\text{obj}}, q^1) = \sum_{n=1}^{N} \begin{cases} \pi_t^{(n)} & \text{if } \text{ID}_i^{\text{obj}} = \text{MMO} \\ & \wedge\ \text{MMO} \in q_t^1 \\ & \wedge\ q_t^1 \in \mathbf{b_t^{(n)}} \\ 0 & \text{else} \end{cases} \tag{6.10}$$

When a single object manipulation is recognized, the object with highest probability $P_{\text{obj},t}(o_i^{\text{obj}}, q^1)$ is selected as the single object manipulated. On the manipulative level, there is no main manipulative object. The object probability for the primitive model q^0 is similar to Eq. 6.10. The difference is that q^1 should be replaced by q^0 and MMO should be changed to the ID of the object in the hand c_{hc} or the object in the search area c_s according to the different binding requirement.

6.4.3 Evaluation Results

In order to evaluate the quality of the manipulative gesture recognition described in the previous Subsections, images of size 320x240 pixels depicting the table (see

Fig. 6.16) are recorded with a frame-rate of 15 images per second. The objects on the table are from left to right a tea can, milk, sugar, a cup, and a plant. For carrying out the experiment, the objects are always positioned at the same place on the table. This is necessary as the primitive models are based on the velocity and the relative angle of the hand motion and varying object positions would make the manipulative recognition more difficult. Note that for the experiment object recognition results were simulated.

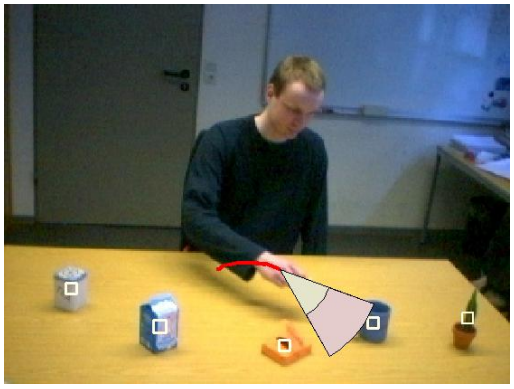


Figure 6.16: A snapshot of the office scenario depicting the hand trajectory (red), the current search context and the object positions.

The manipulation tasks carried out in the experiment and the hierarchical modeling of these tasks are listed in Tables 6.2 & 6.3. On the highest abstraction level, the tasks *prepare tea* and *water plant* can only be performed in one unique order. In *prepare coffee*, however, the demonstrator can choose different objects like milk or sugar and can use different orders for preparing his coffee resulting in a total of four possible sequences for *preparing coffee* as shown in Table 6.2.

Manipulation tasks	
Name	Sequence of single object manipulations
water plant	returnmove(cup)
prepare tea	puremove(cup) → returnmove(tea can)
prepare coffee	1) puremove(cup) → returnmove(milk) 2) puremove(cup) → approachmove(sugar) 3) puremove(cup) → returnmove(milk) → approachmove(sugar) 4) puremove(cup) → approachmove(sugar) → returnmove(milk)

Table 6.2: The definition of manipulation tasks at the highest level of the hierarchy.

Single object manipulations	
Name	Sequence of manipulative primitives
puremove	approach → move2emp → maniend
approachmove	approach → move2obj → maniend
returnmove	approach → move2obj → move2emp → maniend
Manipulative primitives	
Name	Manipulative Gesture
approach	hand approaches an object
move2obj	move an object to another object
move2emp	move an object to an empty place
maniend	hand draws back

Table 6.3: The definition of single object manipulations and manipulative primitives.

Each of the three manipulation tasks is performed 4-5 times by 8 persons resulting in 36 sequences for each task and a total of 108 sequences. The motion vector of each primitive model is generated by averaging over a training set of manually segmented motion data. Similarly, the context vector is defined manually by choosing for each time step an appropriate search area, if applicable. The initial, transition, and terminal probabilities are determined by analyzing the ground truth of the manipulations. For recognition a total of $N = 3000$ particles is used. The scaling factors α and ρ are between 0.8 and 1.2 with variance $\sigma = 0.1$.

Figure 6.17 displays a plot of the probabilities in the three hierarchy levels during recognition of a *prepare tea* task. Quantitative results of the recognition for the three tasks and the associated primitives are listed in Table 6.4. At the primitive level, an empty entry means the primitive is not used while a horizontal bar means the primitive is defined as part of the single object manipulation but it has not been used by the subjects during manipulation.

A comparison of the task recognition rates shows that *prepare tea* and *water plant* are successfully recognized most of the time with 94.4% and 97.2%, respectively. The recognition rate of the task *prepare coffee* is lower reaching only 86.1%. Looking at the recognition rates at the different levels, we find the reason are the low recognition rates of the *move2obj* and *maniend* primitives in the *approachmove* object manipulation. This is due to the camera's viewing angle, as the movement of the hand from the sugar to the cup in front of the body only generated a short hand trajectory in 2D that was difficult to recognize robustly.

The presented approach relies on 2D hand motion information and object recognition results. A direct consequence from the restriction to 2D data is the view-

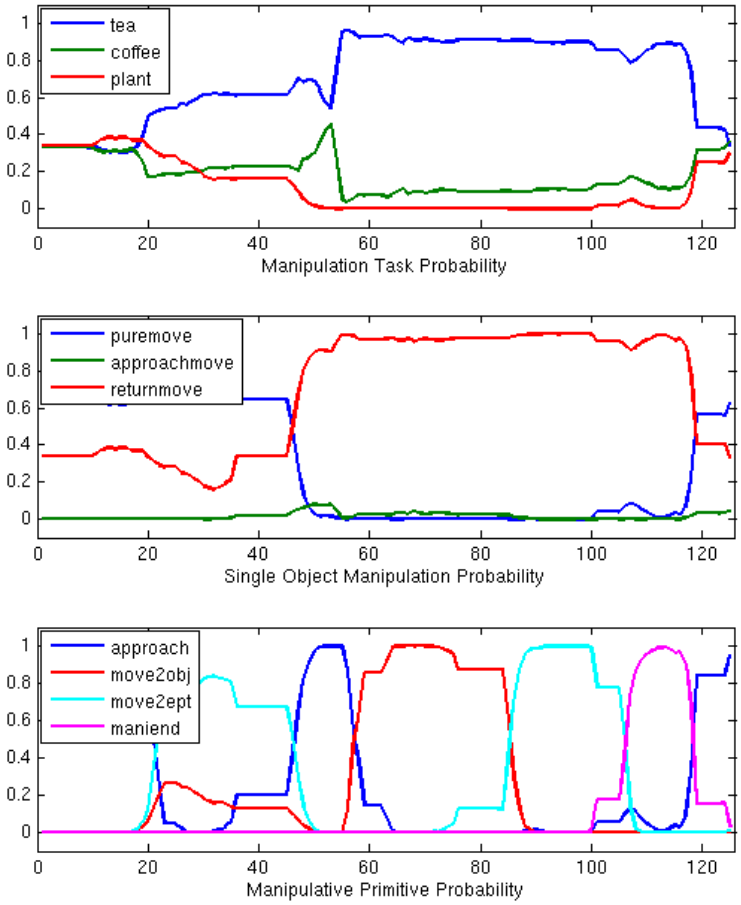


Figure 6.17: The recognition result from one video stream depicting a *prepare tea* task.

dependency of the manipulative primitives. This requirement of having 'typical' object positions to get similar trajectories is an obvious limitation. Using a 3D body tracking approach providing 3D hand motion information is one possible solution to this limitation. With a complete body tracking system, the representation of manipulative primitives could be extended by applying, e.g., body-centered trajectory models in order to achieve a more generic action recognition system. Obviously, not only the hand motion but also the object positions should be available in 3D to achieve a view independent action recognition.

Task Level						
tasks	prepare tea		prepare coffee			water plant
(%)	94.4		86.1			97.2
Single Object Manipulation Level						
Seq.	purem.	returnm.	purem.	approachm.	returnm.	returnm.
(%)	100	94.4	100	76.9	93.1	91.7
Primitive Level						
approach	100	91.7	100	97.2	96.5	100
move2obj		100		76.9	96.5	94.4
move2emp	100	100	97.2		89.6	80.5
maniend	–	91.7	–	73.6	90.9	97.2

Table 6.4: Recognition results for the different levels grouped by task.

6.5 Summary and Conclusion

In this Chapter different types of context and their incorporation into the gesture understanding process have been presented. For gestures performed by the human in an interactive setting with the intention to convey information to the robot, the human often provides additional information. This user-provided context helps in gesture understanding and is especially prominent in pointing gestures. Relevant types of such context incorporation have been detailed in this Chapter: 1) Posture information can be used to restrict the object search space using an eye-hand line; 2) Verbal information can complement the pointing gesture for specifying the type of pointing target (to an "object" or a "location") or the interpretation of the pointing gesture ("object is left of hand"). 3) Verbal information can specify object properties to support the understanding of a pointing gesture, i.e., the resolution of an object reference.

In order to give an example of the incorporation of such user-provided context, an approach for resolving object references based on a pointing gesture was shown. The two different processing streams were detailed for a) re-cognizing an already known object by employing an object recognizer selected from the verbally specified object name and b) finding an unknown object by visual processing parameterized using verbal information.

Besides the context specified explicitly by the user, also the overall setting can be used for understanding gestures. This situational context is especially relevant for understanding manipulative gestures that are usually not accompanied by any user-provided context. Incorporating situational context can be done in different ways. In a body-centered approach, the hand motion information is enriched with

the context to infer the manipulative gestures. In an object-centered approach, the focus is on the objects and any approaching hand is analyzed for whether it exhibits a motion that can be a manipulative gesture. While each of these types sets the focus on one kind of information, there are more recently a number of approaches aiming at a parallel fusion of both, objects and gestures. Here, the approaches range from symbolic rule-based approaches to probabilistic graphical models. Especially the graphical models offer the advantage of being learnable, i.e., they can be trained with data instead of the manual coding necessary for symbolic and rule-based approaches.

In order to give an example for the use of a graphical model for manipulative gesture understanding, an algorithmic solution for recognizing complex actions that consist of a sequence of more simple manipulative actions has been described. The approach embeds data from a body-centered gesture recognition method and object recognition results in a hierarchical hidden Markov model for modeling and recognizing higher level manipulative intentions. As representation a dynamic Bayesian network is used and the inference is done by a particle filter. The achieved recognition performance indicates that manipulative tasks can be successfully recognized by such a context-based hierarchical hidden Markov model. While the recognition is achieved by setting a threshold on the state end probability, the constant observation of this probability allows to also infer the current complex action. This ability to predict in a probabilistic fashion what might be currently ongoing is an important additional benefit that could be used in a top-down manner for steering the processing of the visual input data. While the example approach has not taken advantage of this information, this kind of processing can be expected to become more prominent in future gesture understanding approaches.

7 Robots Exhibiting Gesture Understanding Capabilities

In the previous Chapters the algorithmic aspects of gesture understanding methods have been covered with a focus on methods applicable to mobile robots. This Chapter is devoted to giving an overview over selected robots that have been equipped with such gesture understanding capabilities, allowing them to understand symbolic, pointing, and manipulative gestures.

In the last decade, a countless number of robots has been developed in research labs around the world, exhibiting different kinds of interaction capabilities. Due to the increasing availability of cheap color cameras and the growing computational power of standard PCs, vision-based understanding of gestures on mobile robots is becoming a standard part of the human-robot interface. In Section 7.1 some selected robots having gesture understanding capabilities in order to enable a more natural interaction with the human are presented. Going beyond these communicative aspects, Section 7.2 will cover robots that can understand and partially also reproduce manipulative gestures.

Obviously, many gesture understanding approaches presented in this book are not limited to applications on a mobile robot. It is rather that the constraints encountered on a robot like, e.g., the requirement of a single camera perspective and the huge variability of the background, pose strong challenges to the algorithms used. Therefore, the algorithms capable of operating on a mobile robot could also be applied in other settings. This includes scenarios where images of several cameras could be processed like, e.g., in intelligent rooms (see, e.g., Irie et al., 2004; Stiefelhagen et al., 2008). However, this Chapter will be limited to the review of gesture understanding approaches deployed on robots.

7.1 Robots Understanding Communicative Gestures

In this Section some robots that are targeted at understanding communicative gestures will be presented. A classical application of gesture recognition is the control of the robot with symbolic gestures like, e.g., a stretched out arm with open hand to indicate to the robot to 'stop' its current activity. Besides such a straightforward

robot motion control based on gestures without any context, there is a large number of robots nowadays that is capable of understanding pointing gestures. The next two Subsections will give examples of robots capable of understanding symbolic and pointing gestures.

7.1.1 Symbolic and Conventional Gestures

The first applications of gesture recognition for mobile robots date back to the early 90's. For example, Kortenkamp et al. (1996) presented a robot that recognized static arm gestures based on the configuration of the arm limbs and the head. The gestures looked somewhat like the gestures performed by airport workers to direct planes to their parking position.

Later works have had access to better cameras allowing to focus on the hand posture instead of the overall body posture. This resulted in recognition approaches for gestures that were also conventionally used in human-human interaction.

For example, the robot Robovie was equipped by Hasanuzzaman et al. (2007) with a system to learn and recognize static command gestures. The demonstrated hand postures included counting to three, pointing left and right, as well as thumb up and fist up.

Figure 7.1.1 depicts an example developed by Wang and Wang (2008), the mobile robot NTU PAL1 with the corresponding camera image showing a typical command gesture. The gesture vocabulary of this robot consists of three gestures, the open palm shown in the camera image, a fist, and the chinese gesture for 'six' with extended thumb and little finger while keeping the rest of the hand closed.

(a) NTU PAL1 robot

(b) Camera image of command gesture.

Figure 7.1: Controlling the motion of the mobile robot NTU PAL1 by command gestures (images from Wang and Wang, 2008).

Note that the recognition of symbolic and conventional gestures implemented for robot control is often a recognition of hand or body postures, i.e., the configuration is the relevant information and there is no temporal aspect involved. One reason for this is that the rejection of unknown gestures is a challenging task. During gesturing, a human may make many movements that may be accidentally recognized as gesture in a trajectory-based approach. If gesture recognition is used for controlling the robot, such errors would result in an unexpected robotic action. Consequently, characteristic postures are the preferred type of gestures for robot control. Symbolic gestures that are not directly intended for robot control but rather allow the robot to understand communicative interactions like, e.g., a waving or hand-shaking person are often realized together with the ability to recognize pointing gestures and are covered in the next Subsection.

7.1.2 Referential and Pointing Gestures

Following the initial approaches to recognize command gestures, later activities aimed at recognizing gestures that have no symbolic character on its own but are referring to another entity in the environment (see, e.g., Sakagami et al., 2002; Ghidary et al., 2002; Haasch et al., 2005; Stiefelhagen et al., 2007; Axenbeck et al., 2008; Martin et al., 2010; Breuer et al., 2011).

A famous robot equipped very early with the ability to recognize 3D pointing gestures is the humanoid robot ASIMO (Sakagami et al., 2002). Furthermore, this robot is also able to recognize symbolic gestures that are not defined by a certain hand posture but have a temporal aspect like, e.g, waving or hand-shaking.

A more recent example of a robot that can be directed to certain locations based on 3D pointing gestures is the robot HOROS (Martin et al., 2010) depicted in Fig. 7.2. This robot employs a holistic approach for pointing gesture recognition (see Section 5.1) that is explicitly triggered by verbal commands. The images depicted in Fig. 7.2(b) give a good impression of the large variability in the image data that is encountered by a robot in a real world setting and has to be handled by the gesture recognition algorithms.

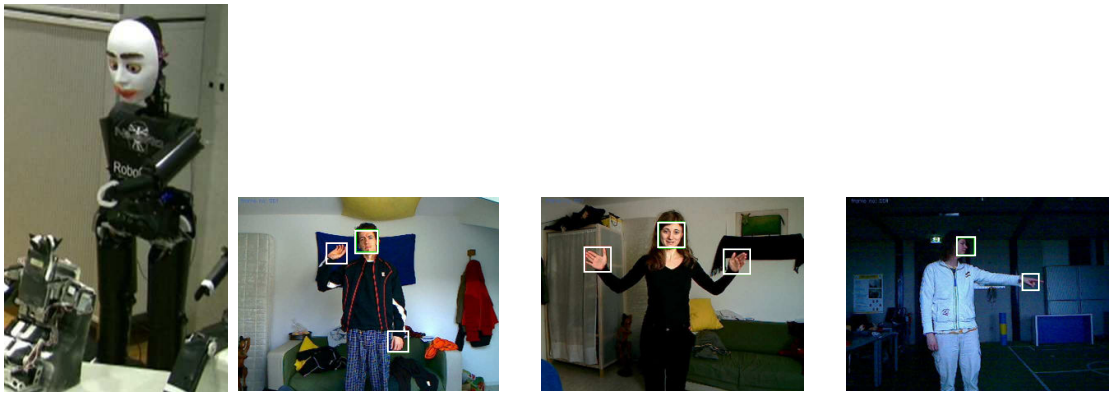
An example of a robot that also recognizes gestures performed with both hands is Robotinho depicted in Fig. 7.1.2. Axenbeck et al. (2008) employ appearance-based hand detectors (see Section 3.2.3), track the detection results with a Kalman filter (see Section 4.2.1) and employ HMMs (see Section 5.4.1) to recognize the trajectories. Besides recognizing one-handed gestures like waving and pointing, also two-handed gestures to indicate object sizes and a 'do not know' gesture. These gestures are symmetric, i.e., both hands perform a similar gesture and the

(a) HOROS (b) Camera images with pointing gestures.

Figure 7.2: HOROS observing human interaction partners pointing to locations (images from Richarz et al., 2007 and Martin et al., 2010).

two-handed gesture can be recognized by requiring the individual hands to perform the same gesture in synchrony.

An early approach where a rather simple gesture recognition was combined with verbally specified object properties (see Section 6.1) was presented by Ghidary et al. (2002). Their robot recognizes pointing hand postures and the user can provide information on where a referenced object is positioned (left), the name of



(a) Robotinho (b) Images of recognized one- and two-handed gestures.

Figure 7.3: The robot Robotinho can recognize two-handed gestures including the indication of an object size (images from Axenbeck et al., 2008).

the object ('TV') and the rough size ('large'). Alternatively, instead of a single pointing posture the human can also indicate for large objects the two opposing corners of the object when he/she indicated this verbally beforehand (two points).

The robot Leonardo (Breazeal et al., 2004) is not a mobile robot but has been included here for its impressive appearance (see Fig. 7.4) and its multi-modal interaction capabilities. In this setting, a human can interact with saliently colored buttons arranged around the stationary robot. Leonardo recognizes deictic gestures in combination with speech and is therefore capable of resolving multi-modal object references. Specifically, it is possible to assign names to buttons by giving verbal information. However, the buttons are very simple objects with a salient color and learning the objects itself is not the primary focus. Instead, the focus is on building a socially intelligent robot that can learn to perform a simple task from natural human instruction, achieve a task in close collaboration with a human, and employ communication strategies like glancing or nodding to maintain a common ground. All these capabilities are in this very early system instance shown exemplarily with a strong focus on the buttons and interactions with these buttons. Note that as the system relies on an additional stereo camera that views the complete scene from the ceiling to recognize pointing gestures, it does not have the challenges faced by a mobile robot like, e.g., limited field of view.

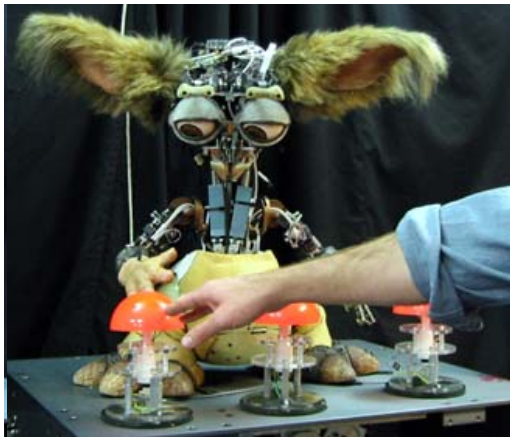
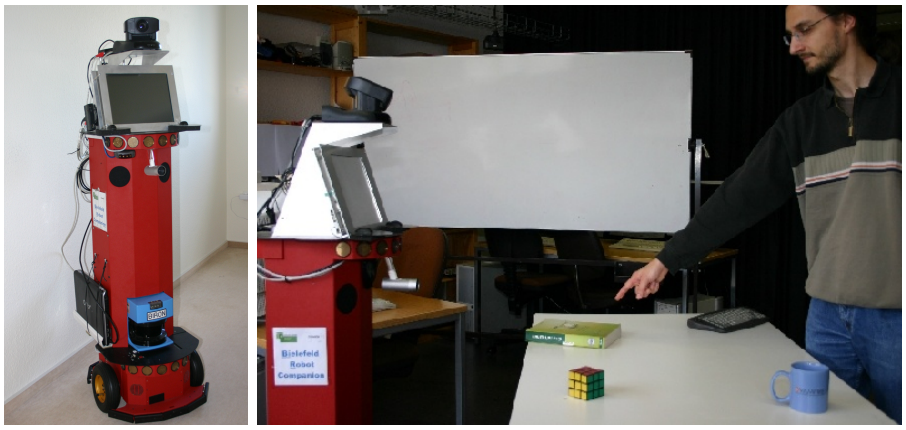


Figure 7.4: Interaction with Leo (image from Breazeal et al., 2004).

A more sophisticated incorporation of the user-provided verbal information was demonstrated by (Haasch et al., 2005) and has been presented in detail in Section 6.2. This system for resolving multi-modal object references has been implemented on the robot BIRON depicted in Fig. 7.5(a). The system allows a user to point to objects on a table and specify as additional information the color and

size of the referenced objects as shown in Fig. 7.5(b). While one camera observes the complete human in order to recognize pointing gestures, a pan-tilt camera is directed to the pointed-at location in order to identify the referenced object.



(a) BIRON (b) Example interaction with pointing gesture.

Figure 7.5: The robot BIRON understands multi-modal object references consisting of a pointing gesture and a verbally specified object color or size.

Although all of the robots outlined above show impressive gesture understanding capabilities that were partly realized already some 10 years ago, there are nearly no personal robots commercially available that possess any kind of communicative gesture understanding. This is partly due to the fact that there are only few personal robots commercially available today at all, but looked at it the other way around this low commercialization success of personal robots may be due to their cumbersome non-gestural interfaces restricting their usability. The reason for not providing commercial robots with advanced gestural understanding capabilities may be the low robustness of many research algorithms to real-world conditions. The problems encountered when applying robots with vision-based gesture recognition technology in ordinary households with arbitrary lighting conditions are manifold and have up to now not been solved in a sufficient quality to enable a natural human-robot interaction that meets the expectations of ordinary users.

7.2 Robots Understanding Manipulative Actions

The robots capable of understanding communicative gestures directly allow for a more natural human-robot interaction. If robots can also understand manipulative

actions, this type of gesture understanding can improve the interaction in a broader range of applications:

Monitoring behavior: A monitoring situation that is gaining increasing attention in the last years is the support and monitoring of elderly people. By keeping track of what actions a person has performed, a robot could, for example, remind the human about taking a glass of water with the daily medicine if this action did not happen in a certain time window.

Structuring communication: If the robot can reason about the current occupation of the human based on the recognized gestures like, e.g., the human holding a book for reading, the robot can choose not to interrupt the human now for some question but to wait until the human has finished his reading. In this sense, such manipulative actions would fall into the class of 'interactional gestures' (see Section 2.1.3).

Reproducing the action: If a robot can reproduce a manipulative action performed by the human, the effect on the interaction depends on the level of understanding. For example, if the robot can merely reproduce the observed movements, this can be used for interaction games (Bertsch and Hafner, 2009). In case the robot has a deeper understanding of the actions, i.e., what the intended effect of the movement is, the reproduction can also incorporate constraints like, e.g., obstacles in the path of the movement to be reproduced (Mühlig et al., 2010).

Research on *monitoring behavior* has not yet targeted robots but has focussed primarily on using several cameras installed at the ceiling of a room to observe the acting humans. The use of action understanding for *structuring communication* is obviously relevant for real robots, but has no dominant influence on research activities. The driving force for action understanding research is the quest for robots capable of *reproducing the action* that can be easily instructed by humans through demonstrating a new action to the robot. Research in this direction has been primarily pursued by the robotics community and is known under a variety of terms including *learning by watching* (Kuniyoshi et al., 1994), *programming by demonstration* (Ehrenmann et al., 1999) and *imitation learning* (Billard and Siegwart, 2004).

Here the term imitation learning will be used, which emphasizes the interactive learning aspects in a human-robot interaction. Many approaches focus on reproducing an observed motion and some examples will be given in Section 7.2.1. Example systems abstracting from the detailed movements and incorporating the

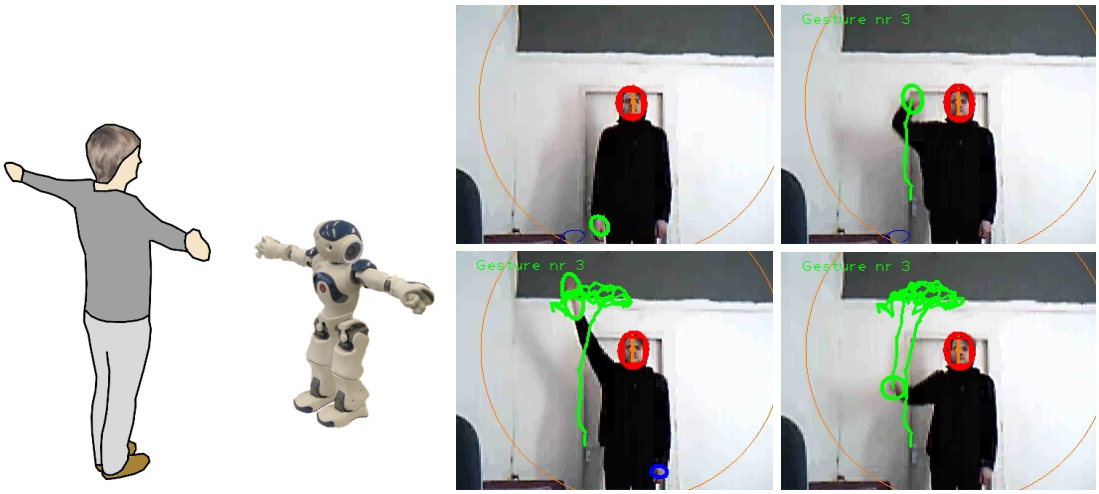
overall context like, e.g., the manipulated objects and obstacles in the environment, are covered in Section 7.2.2.

7.2.1 Imitating the Observed Motion

Approaches aiming at the pure imitation of the observed motion usually make the underlying assumption that the observed motion is meaningful, i.e., the human explicitly performs the action for the robot to observe it. Typically the action is repeated several times to extract the invariances from the demonstrations (see, e.g., Mühlig et al., 2009). Besides extracting the relevant features from the observed demonstrations, also the different embodiments of the demonstrating human and the reproducing robot have to be considered. An approach avoiding this challenge of imitation learning is to learn a demonstrated gesture from several demonstrations and then to manually code the respective gesture reproduction on the robot.

For example, Bertsch and Hafner (2009) demonstrated such an imitation interaction with the small humanoid robot NAO depicted in Fig. 7.6(a). The approach builds upon a recognition of the gestures for imitating them subsequently. A hand detection based on skin-color (see Section 3.2.2) and a subsequent tracking provide the basic features as depicted in Fig. 7.6(b). The trajectory is segmented into basic motion segments and simplified HMMs (see Section 5.4.1) are used to enable the recognition of 8 different gestures on-board the small mobile platform.

Instead of manually coding the reproduction of the gesture, the automatic repro-



(a) NAO (b) Images including tracking results

Figure 7.6: Interaction with Nao (images from Bertsch and Hafner, 2009).

duction is a major target of many recent approaches to imitation learning. Transferring the observed human motion to the control system of the robot's manipulators is the so-called motion retargeting problem. A recent example of such an approach by Dariush et al. (2009) allows the humanoid robot ASIMO to perform a similar motion as demonstrated by a human teacher. Here the recognition of the gesturing human is rather simple as 3D data from a time-of-flight camera is used to detect relevant key points of the human (shoulder, elbow, wrist, ...). This data is then retargeted to perform a similar motion on ASIMO as depicted in Fig. 7.7.

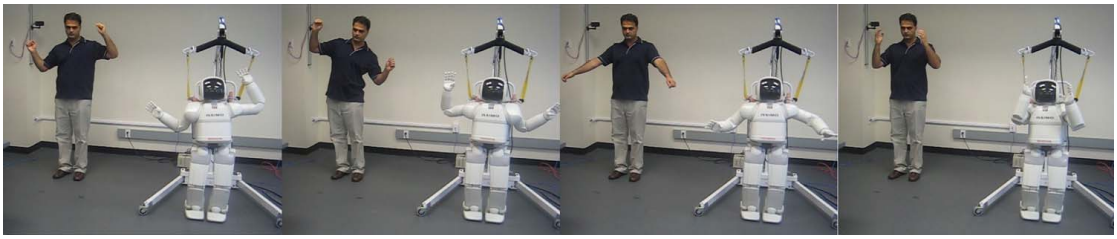


Figure 7.7: Motion retargeting on ASIMO (images from Dariush et al., 2009).

Obviously, for such a retargeting there is no understanding of a gesture needed, as the observed motions are just replicated without any deeper understanding of their target. The next Subsection will consider robots with a more advanced understanding and reproduction of observed gestures.

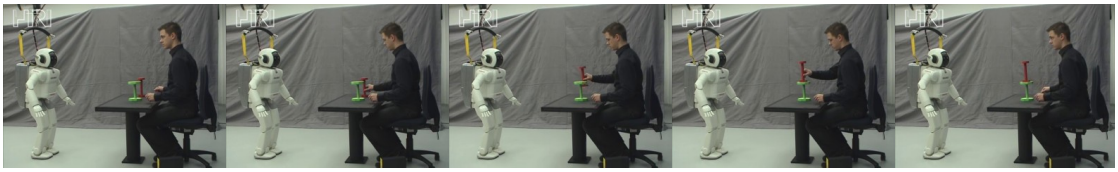
7.2.2 Understanding for Reacting to the Environment Manipulation

Inferring the intention of an acting human based on visually observing his acting is fundamentally different from approaches that aim at only imitating manipulative actions. As pointed out before, if the embodiment is different this requires a retargeting if the motion is to be imitated as a direct copy of the observed movement. However, also the differences in the environment need to be considered for a real understanding. For example, if a human is putting an object on top of another object, his movement alone is not specifying this gesture sufficiently. In other words, if the robot would just reproduce the observed movement, it will certainly not have a good success rate in safely placing two objects on each other. Obviously, the information of the manipulated objects need to be taken into account as well as other factors like, e.g., obstacles that require an adaptation of the robot's movement to achieve the target.

An approach that incorporates the object information into the imitation process in order to enable the robot ASIMO to learn object manipulations and reproduce them has been shown by Mühlig et al. (2010). In this approach, the hands and face of the human demonstrator are detected based on skin color (see Section 3.2.2) and subsequently tracked. An internal model of the human body in the form of a stick figure is used to derive the 3D posture information from the detected hands and face. By imposing constraints on the motion of the stick figure model, noisy or incomplete detection results (e.g., caused by occlusions) can be handled. This processing scheme is a simplified form of a model-based body tracker (see Section 4.5.2). Based on the body model, posture recognition is performed to enable the human teacher to provide commands to the robot that help to structure the learning process by providing start and end signals.

The main task of the body model, however, is to provide information on the acting hands. This information is combined with object information applying situational context in the form of an object-centered approach (see Section 6.3). From the analysis of the object motions in combination with the body model, ASIMO can interactively learn object manipulations and also reproduce them as depicted in Fig. 7.8. As the robot has built up an internal representation of the movement to be reproduced, it can also cope with additional constraints like the blocking of the movement path by an object or the reproduction of a one-handed manipulation with two hands.

Besides the depicted examples there are many more approaches that aim at



(a) Side view of human demonstrating a manipulative gesture to Asimo

(b) Front view of Asimo reproducing the observed manipulative gesture

Figure 7.8: Asimo learning to reproduce manipulative gestures (images from Mühlig et al., 2010).

realizing imitation learning on mobile robots. More information on the learning techniques itself can be found in the robotics community (see, e.g., Demiris and Billard, 2007; Billard et al., 2008; Kober and Peters, 2010).

7.3 Summary and Conclusion

The robots described in this Chapter already exhibit a variety of gestural understanding capabilities and give an impression of what the future of intelligent robots may be. With the research progress in the individual functionalities, the performance of such robots will become more and more human-like and the interaction quality will gradually improve. A very important point that must be made here, however, is the fact that all the interaction behaviors are programmed by experts. The described systems are able to learn factual knowledge of objects, but the learning process has been designed taking the human-human interaction as role model. This implementation of central learning processes for factual knowledge may be acceptable for core capabilities, but how about learning procedural knowledge? It is not possible to implement all procedural knowledge by experts, so an intelligent robot should have the capacity to learn this type of knowledge.

While 'learning' in general is a huge research question, the more specific question of how to learn to perform certain gestures is of crucial importance for a humanoid robot. For humans, learning a new gesture to manipulate some object is closely related to observing other humans performing the task and trying it out by oneself. If a robot has to be able to manipulate arbitrary objects like, e.g., a new coffee machine that was not existing when the robot was built, it has to have the capability to 'learn' how to make coffee with the unknown machine.

In the approaches for gesture understanding outlined in previous Chapters, the gesture recognition models were manually designed. For the recognition of pointing gestures this is not a problem, but for recognizing action sequences a robot should be able to autonomously learn to recognize and reproduce new actions. Recent results like the work of Mühlig et al. are very impressive, although the interaction setting relies on the human instructor to control the learning phase. Humans have the ability to learn new actions in a more natural interaction, so making a robot learn like a human will ease the teaching task and will also pose less restrictions on the type of object manipulations that can be learned. A typical learning scenario is the interaction between a mother and her child where the mother demonstrates to the child how to use objects. If a robot could learn to handle arbitrary objects in a way similar to how a child learns from its mother, a major obstacle towards building

truly intelligent interactive robots would be overcome. Insights from experiments aiming at revealing the mechanisms of such a social learning of gestures are the topic of the next Chapter.

8 Towards Learning of Gestures: Studies on Human Gesture Understanding

The algorithmic approaches for gesture recognition presented in this book rely on models for the gestures to be recognized. These models are either specified manually by the programmer or are constructed during a carefully controlled teaching phase with the human demonstrating the gesture repeatedly several times. In this Chapter, a different view on the acquisition of new gestures will be presented which aims at using children's learning processes as a role model. Following a review of the limits of classical learning approaches in Section 8.1, the modifications of actions performed by caregivers during action demonstration are introduced in Section 8.2. Experimental findings from studies investigating motionese are presented in Section 8.3. A study focussing on the complete interaction loop including a robot performing gesture reproduction is presented in Section 8.4. The chapter concludes with a summary in Section 8.5.

8.1 Limits of Classical Approaches to Learning

In many classical gesture recognition algorithms the models used for the analysis of the hand motion are designed by programmers. This means they are constructed manually before the recognition algorithms are integrated into complete systems like the interactive robots presented in the previous Chapter. As long as the different gestures to be recognized are known to the programmers beforehand, this method for constructing a gesture recognition system can be applied. Obviously, models for clearly defined hand motions like pointing and conventional gestures (see Section 2.1.3) can be defined in a recognition system a priori. Looking at the huge variety of different objects present in human environments, however, points out a basic problem of such predefined gesture models for handling manipulative gestures: it is impossible to cover all possible manipulative gestures in the design process in order to enable a robot to perform arbitrary object manipulations.

An obvious solution to this problem is to develop approaches that learn from human demonstrations how to perform manipulative gestures. This so-called *imitation learning* (see also Section 7.2) is guided primarily by five questions that need

to be decided on by a robot (Dautenhahn and Nehaniv, 2002): 'who to imitate', 'when to imitate', 'what to imitate', 'how to imitate', and 'how to evaluate' the performed imitation.

There has been intensive research on solving these questions by technical approaches (cf., e.g., Demiris and Billard, 2007; Billard et al., 2008; Dariush et al., 2009; Kober and Peters, 2010; Mühlig et al., 2010). However, most algorithmic solutions concentrate on the questions of 'what to imitate' and 'how to imitate'. The two questions are generally tackled by analyzing the observed hand trajectory, finding invariant features across several demonstrations ('what'), and enabling a robot to reproduce the demonstrated gestures ('how'). In order to simplify the task, these learning algorithms assume that the human demonstrator performs clearly separated manipulative gestures and that no other motions are performed during a teaching session.

Most current imitation learning approaches are, therefore, not representing natural interaction situations but are characterized by a very technical approach to imitation as they concentrate solely on extracting a trajectory model. This is in contrast to the first two questions of 'who to imitate' and 'when to imitate' that are strongly related to the social interaction between the demonstrator and the learner. In a scenario where one human demonstrates an object manipulation to another human, the human-human-interaction related to these two questions may take on the form of symbolic start/stop markers (e.g., verbal commands like "Look here" and "That's it"). Such an explicit verbal annotation of relevant trajectory parts used in the transfer of skills between adult humans, however, is radically different from how parents teach their children new skills. Especially in the interaction between a child and a caregiver, social aspects like, e.g., verbal and gestural 'attention getters' are of crucial importance as they allow the caregiver to guide the child's attention to demonstrated actions that the caregiver wants the child to learn. This guiding of attention is not limited to verbally denoting the start and end of a gesture, but includes modifications of the hand motion itself as research on adult-child interactions has shown (Brand et al., 2002). This so-called 'motionese' that adult demonstrators exhibit when demonstrating gestures to a child will be introduced in Section 8.2.

Making these findings on hand motion modifications accessible to technical systems aiming at gesture understanding requires to bridge the gap between developmental learning and computer science. Recent research by Vollmer (2011) indicates that imitation learning as it is performed by the child in interaction with its caregiver is a dynamic process. The interactive aspects are not captured when focussing only on the demonstrator's hand trajectory as it is done in most current imitation

learning approaches. In order to enable a technical system to benefit from 'motionese' modifications of the demonstrated hand motion, Section 8.3 introduces an algorithmic approach for measuring 'motionese', i.e., to detect 'when to imitate' an observed gesture.

Going beyond the hand motion itself, the interaction between demonstrator and learner is another domain which is not considered by current imitation learning approaches. While many classical approaches focus on trajectory information when answering 'what to imitate', the trajectory itself (i.e., the 'manner') may not be relevant but rather the effect (i.e., the 'goal') of the gesture like, e.g., pressing a button. Distinguishing between these two alternatives may be done based on analyzing the demonstrator's feedback on the robot's attempt to reproduce the action. Section 8.4 will introduce a human-robot interaction study aiming at investigating such an interactive learning setting.

Using feedback from the demonstrator after an imitation attempt is one way of answering the question 'how to evaluate'. Another alternative is to motivate the demonstrator to emphasize relevant action parts ('what to imitate') already *during* an action demonstration. For example, varying the simulated gaze of the robot learner can be used to communicate to the demonstrator what the robot currently 'understands'. The feedback of the demonstrator that is stimulated by the robot's comprehension display enables it to determine whether its current understanding of 'what to imitate' (manner or goal) is correct. Also such non-gestural aspects will be shortly covered in Section 8.4 before the Chapter concludes with a summary.

8.2 Modifications of Child-directed Motions: Motionese

In the research findings on developmental psychology there is a growing support for the idea that children are learning from input that is specifically designed for them. For example, adults speak differently when addressing children and this child-directed speech – *motherese* – has been characterized as directing the child's attention to relevant aspects of the speech signal (Fernald and Simon, 1984; Dominey and Dodane, 2004). While motherese is well known and has already been used as a cue in technical systems (see, e.g., Breazeal, 2004), more recent studies also found evidence for a modification in mothers' infant-directed physical actions that has been termed *motionese* (Brand et al., 2002; Gogate et al., 2000): *as part of 'motherese' [...] mothers' infant-directed actions [...] reveal distinctive characteristics that amplify or exaggerate meaning and structure within their bodily motions.*

(Brand et al., 2002, p.73).

Obviously, this action modification could also be used by a learning robot situated in a social context with a caregiver. It would enable a robot to automatically detect 'who','when', and 'what' to imitate based on the caregiver's action modification that seems to be a natural behavior for any adult demonstrating an action to a child. Consequently, it is necessary to understand in a first step how exactly adults modify their bodily motions and how this can be detected by a technical system.

In a study on motionese by Brand et al. (2002) it was observed that mothers reveal distinctive characteristics amplifying or exaggerating meaning and structure within their bodily motions in their infant-directed actions. These characteristics were identified using 8 intuitive categories: range of motion, rate, repetitiveness, proximity to partner, enthusiasm, interactiveness, punctuation, and simplification. The findings were obtained by analyzing videos with adult-child interactions. Each demonstration recorded on video was given a rating (0-4) for each of the eight categories by human coders. As this analysis was based on manual annotation, it has to be stressed that the definition of the categories was tailored for human coders and often a single parameter like, e.g., hand velocity was implicitly contained in several of the eight categories. In order to make motionese accessible to a robotic system for learning a new gesture, it is therefore necessary to first find a technically measurable equivalent to the rather qualitative ratings of the human coders in the studies by Brand et al.

8.3 Technical Analysis of Motionese

In order to investigate the technical analysis of motionese, an experiment similar to Brand et al. was carried out by Rohlfing et al. (2006) where the adult demonstrator was filmed with a camera. One of the tasks was the stacking of four cups into each other. In order to analyze the differences between Adult-Child Interaction (ACI) and Adult-Adult Interaction (AAI) the adults had to demonstrate the cup stacking to their child (see Fig. 8.1(a)) and to another adult (see Fig. 8.1(b)).

For the analysis the demonstrator's hand motion was extracted from the video using a vision-based 3D body tracking similar to the approach in Section 4.6. Note that the use of intrusive sensing methods outlined in Section 2.2.1 may make natural acting more difficult for the demonstrator and, more importantly, may distract the attention of the child looking at the demonstrator. Figure 8.2 depicts an image of the adult demonstrator before the start of the cup-stacking task together with example trajectory data of the hand indicating how it later performed the task.

(a) Adult-Child Interaction (ACI)

(b) Adult-Adult Interaction (AA)

(c) Adult-Robot Interaction (ARI)

Figure 8.1: The different settings of the studies analyzing motionese: (a) and (b) analyzed by Rohlfing et al. (2006); (c) analyzed by Vollmer et al. (2009) (images from Vollmer, 2011).

Based on tracking the hand, several action parameters can be derived from the hand motion. Consequently, visually observable modifications in actions can be analyzed in a quantitative manner by converting the categories used by the human coders in the study by Brand et al. (2002) to measurable parameters. However, there is no obvious conversion available as there are no direct one-to-one relations between the categories and the measured parameters. For example, 'punctuation' is

Figure 8.2: Example of cup-stacking task with hand trajectory overlaid. Each color of the trajectory represents the action of stacking a cup with corresponding color into the blue cup and the thin line represents moving the hand without any cup (image from Vollmer, 2011).

characterized by the acceleration of the hand and the number of pauses. However, pauses are also one aspect of 'simplification', as more pauses make the overall action sequence simpler.

Consequently, it is necessary to identify those parameters that can be extracted from the trajectory data of hands tracked in video streams and that are relevant for motionese. As a preprocessing step, the video stream is segmented into *motions* and *pauses* based on the hand velocity. In this way, only the parts of the trajectories that actually show a moving hand are analyzed. For a frame sequence segmented as a motion, the path traveled by the hand ($PathLength$) and the distance between start and end point ($DistanceTravelled$) are calculated. Additionally, the duration of the motion ($MotionDuration$) and the preceding pause ($PauseDuration$) are counted in video frames. For each motion, i.e., each sequence of frames containing a moving hand, the following parameters are calculated:

- *average velocity* $\bar{v} = \frac{\sum v}{MotionDuration}$
- *average acceleration* $\bar{a} = \frac{\sum \frac{dv}{dt}}{MotionDuration}$
- *roundness* $s = \frac{PathLength}{DistanceTravelled}$ is a measurement for the shape of the motion. It is defined as the path of the hand over the distance between the start and end position. A motion describing a curved path results in a high value, a straight move has a low value.
- *pace* $p = \frac{MotionDuration}{PauseDuration}$ indicates how long the motion is in relation to the preceding pause. If the pauses are relatively short, the motion has a high pace and vice versa.

Using these parameters, the trajectory data from the Adult-Child and Adult-Adult interactions has been analyzed by Rohlfing et al. (2006). In this initial study, significant differences have been found between how adults demonstrate a task to their children in comparison to the demonstration to another adult. In demonstrations to children, roundness was found to be significantly lower, there were more pauses, and there was a trend for a lower pace.

In a subsequent experiment by Vollmer et al. (2009), the same experiment has been repeated using as 'learner' a simulated robot face presented on a screen to investigate the Adult-Robot interaction (see Fig. 8.1(c)) in relation to the AAI and ACI conditions. In order to enable a comparison between all three conditions, the data from the first study was analyzed again in this more comprehensive study using improved analysis tools and evaluation methods. Besides confirming the findings

from the initial study, this also resulted in a better understanding of how adults interact with a robot learner.

Example results from this study are provided in Fig. 8.3 showing the differences in velocity and length of motion pauses for the different interaction partners. The motions in the adult-child interaction and adult-robot interaction reveal longer pauses between the single actions following each other and a lower velocity. These differences result in noticeable longer interaction sequences compared to adult-adult interactions. A much more detailed analysis of the results can be found in Vollmer et al. (2009).

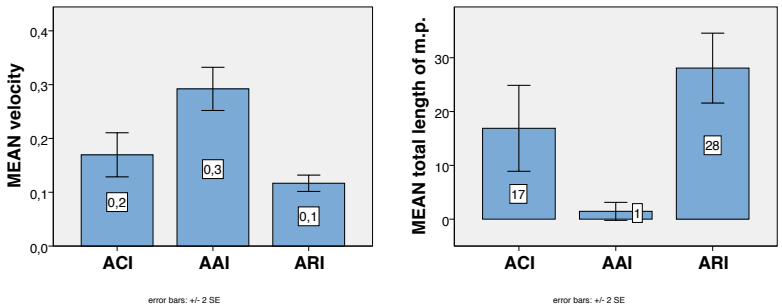


Figure 8.3: Mean values of velocity and length of motion pauses (m.p.) for the different interaction situations (image from Vollmer, 2011).

Summarizing the results, it was found that adult demonstrations towards a robot (ARI) exhibit even more motionese characteristics than the already significant differences found in demonstrations towards a child (ACI). These significant differences indicate the possibility to automatically identify a motion as being demonstrated by the adult with the intention to teach a 'learner' like a child or a robot. This has important consequences as it supports the idea of enabling social learning of an imitating robot, i.e., a technical system that can make use of the way how learning is done by a child. Importantly, this teaching of a robot does not require the human to have any knowledge of the technical system, as he/she can just treat it like a young child. The experiments have verified that humans indeed use child-directed motions (i.e., apply motionese) to highlight relevant parts of action demonstrations not only to children but also to robots.

8.4 Studying Gestural Interaction between Humans and a Robot

The previous Section has shown evidence that modifications in the demonstrator's hand motions can be used by a robot to learn new actions. In this Section the interaction between a human and a robot during such a kind of learning will be analyzed.

In most research approaches towards imitation learning, the interaction is unidirectional in that the demonstrator performs the action and subsequently the robot extracts the relevant information and reproduces the observed action. However, such a setting completely ignores the feedback that can be provided by the demonstrator when observing how the robot has imitated the demonstrated action. Another even more important aspect is the fact that most approaches to imitation learning aim at reproducing the motion while this may not always be the relevant aspect of an action. For example, when pushing a button it may be that the motion (i.e., 'manner') is less relevant than the effect (i.e., 'goal') of the gesture.

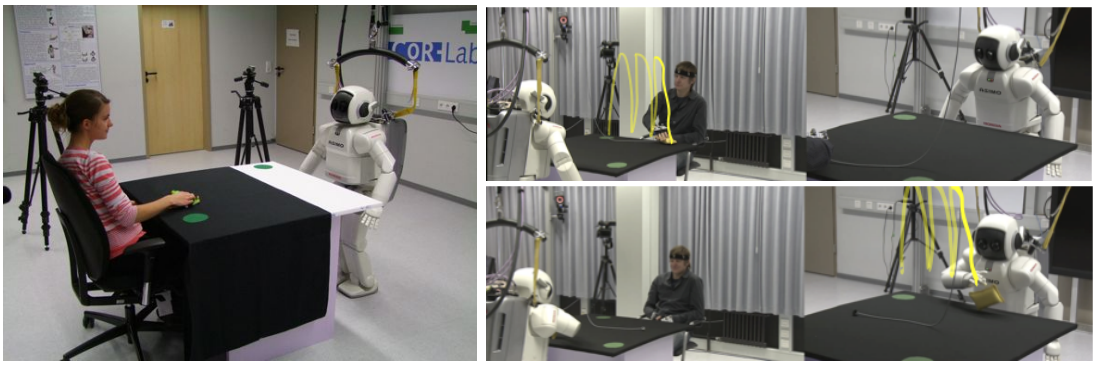
While in the motionese studies it was verified that a demonstrator emphasizes aspects of the hand motion for supporting the learning task, it was not distinguished between manner and goal. Furthermore, the influence of learner feedback on the adult's action demonstration was not considered. In order to understand how the feedback provided in an interactive setting shapes the demonstrator's action presentation and in turn the learning success of the robot, a comprehensive study was carried out by Vollmer (2011, Ch.7). Figure 8.4 shows the setup of this study where the human demonstrates actions to a real robot who, in turn, reproduces the actions.

In this study the human demonstrator was instructed to demonstrate to the robot a total of eight actions that differed in being either manner-crucial or goal-crucial. After each action demonstration, the robot reproduced the action. For each reproduction, before a new action demonstration started the behavior of the robot was randomly selected:

Manner-oriented imitation: the robot reproduced the observed trajectory as exactly as possible.

Goal-oriented emulation: the robot moved the object to the end position with a direct motion without considering the observed trajectory.

Due to this setup of the study, the reproduction was often not correct. This was intended to study how the demonstrator tries to 'correct' the learner by providing



(a) (b)

Figure 8.4: (a) Setting of the human-robot interaction study; (b) example demonstration and reproduction (images from Vollmer, 2011).

some kind of feedback. After each reproduction the human demonstrator could choose to show the action again or to go on with demonstrating a new action (see Fig. 8.5 for a visualization). In this way, the consequences of the robot's reproduction (imitating the motion or emulating the goal) on the demonstrators (next) action demonstration could be investigated.

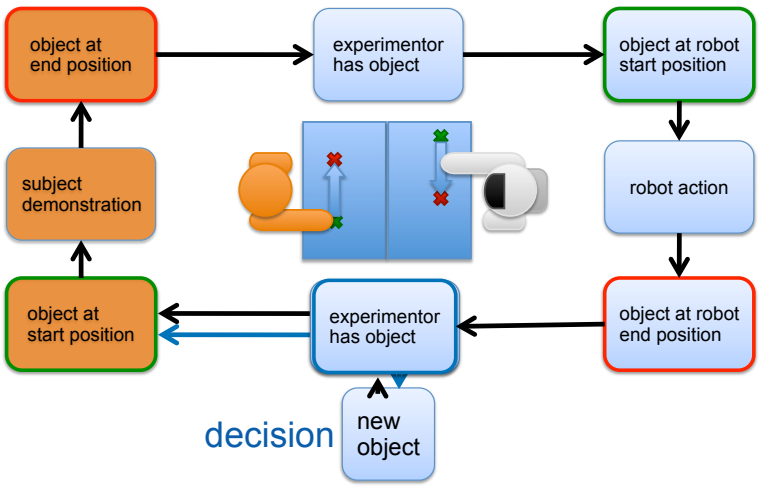


Figure 8.5: Sequence of steps in the interplay between human's demonstration and robot's reproduction (image from Vollmer, 2011).

For the initial demonstration of an action by the human subjects, which was not influenced by the feedback behavior, the motionese parameters were analyzed to evaluate possible differences in the two types of actions. As depicted in Fig. 8.6,

significant differences have been found: manner-crucial actions were demonstrated faster, with a higher pace, and with shorter motion pauses. These results indicate that the type of an action demonstration might already be inferred based only on the motionese parameters of the first demonstration.

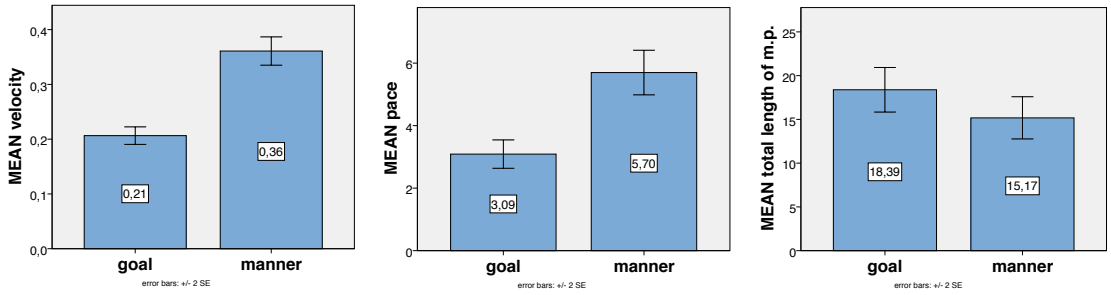


Figure 8.6: Differences in motionese parameters for goal-oriented and manner-oriented action demonstrations (images from Vollmer, 2011).

In the analysis of the effect of the robot's reproduction on the human's next demonstration, it was found that manner-oriented actions that the robot emulated by simply reaching the end position were repeated most often by the human demonstrator. The study therefore verified that the way in which the robot reproduces an action determines how often the human demonstrates the action. This coupling of the robot's feedback to the demonstrator's actions is an important characteristic of interactive learning situations that will enable a much more efficient learning of gestures by future robots that consider the insights from this study.

Besides the gesture-related feedback in the interaction study, also the gazing feedback of the robot was investigated. While a detailed analysis of the results is not relevant here (for an in-depth analysis see Vollmer (2011, Ch.7)), it should be noted that also the way in which the robot's gaze was controlled in the experiments influenced the human demonstration. Depending on whether the robot was set to imitate a motion or emulate a goal, the gaze was either following the hand (imitation) or jumping to the goal position (emulation). If the gaze followed the human hand during a manner-crucial action demonstration, actions were demonstrated slower as the robot was considered 'attentive'. If the robot, however, was in the emulation mode and its gaze jumped to the goal position, the demonstrators interpreted this as 'lack of attention' and tried to capture the robot's attention by, e.g., making a longer pause, in order to support the learning of the robot. These insights on how the robot's gaze influences the action demonstrations emphasize the importance of the robot's feedback behavior for gaining the most support from the human demonstrator during the gesture learning task.

8.5 Summary and Conclusion

This Chapter presented insights from studies into developmental learning going beyond current imitation learning settings that focus mainly on the reproduction of trajectory information. Results from the analysis of adult-child interactions and adult-robot interactions have shown that adults modify their behavior when demonstrating an action to a child and these modifications can also be observed in demonstrations to a robot. Consequently, it becomes possible to realize an imitation learning algorithm that takes advantage of this modified input for automatically detecting 'when to imitate'.

Going beyond the uni-directional analysis of the demonstrated hand motion, this Chapter also pointed to recent research on investigating the interactions between a demonstrator and a robot learner in an interactive setting. The study emphasized that for answering the question 'how to evaluate', the feedback from the demonstrator in a turn-taking setting can provide additional support. Furthermore, by controlling the robot's gaze during an action demonstration, it could be shown that this elicits action modifications by the demonstrator that can be used to distinguish between manner and goal in 'what to imitate'.

The studies analyzing 'motionese' and demonstrator feedback have only touched on the potential of imitation learning in a social setting. The findings suggest that imitation learning will also greatly benefit from incorporating additional cues like, e.g., speech (e.g., 'Nooooooow it's ready') to structure the action demonstration and to group several motions into one action. Although the research on incorporating insights from developmental learning into imitation learning is just beginning, the presented results confirm the large benefit these insights might provide for future gesture understanding algorithms. Instead of manually coding the gestures to be recognized, future systems might be able to extract the relevant information automatically. This will help research in intelligent robots to reduce the complexity of the 'acquisition problem', i.e., to reduce the requirements on built-in architectural constraints of learning mechanisms (Gergely, 2003).

9 Conclusion

In this book the individual algorithmic stages for hand gesture recognition and understanding have been covered together with some application examples and insights from developmental psychology. In this final Chapter, the presented material will be summarized and conclusions regarding the future development of the field of gesture understanding for intelligent robots will be drawn.

Out of the large field of human communication capabilities, this work has been restricted to cover research on human hand motions. The term 'gesture' is not limited to hand motions only but is also used in a broader meaning including, for instance, body posture and facial mimics. All these aspects contribute to equipping technical systems with more natural human-machine interfaces. However, especially for robots that interact with humans and that are intended to fulfill practical tasks, the understanding of *hand* gestures is a key capability for enabling the practical application of such robots.

Current interactive robots that are serving a practical purpose beyond entertainment are primarily being programmed by technical experts and have rigid human-machine interfaces like, e.g., pre-defined verbal commands or command gestures. The understanding of hand gestures is the next big step towards making the interaction with these robots more human-like, i.e., more natural. The approaches covered in this book have focussed on vision-based gesture understanding, considering that mobile robots have to carry all sensors on-board and should perceive not only the hand motions but also context information like, e.g., objects on a table.

Recognizing gestures based on a sequence of images is not a trivial task, and similar to other complex analysis problems solved with computer science methods the research community has mainly applied 'divide and conquer' approaches to this computer vision challenge. Consequently, for the analysis of hand motion, the gesture recognition problem has often been decomposed into detecting the hand, tracking it over an image sequence, and classifying the trajectory. Following this dominant view on vision-based gesture recognition, the organization of this book reflects this decomposition, grouping the respective research in the corresponding Chapters.

The wide range of approaches for hand detection covered in Chapter 3 emphasizes the difficulty of finding a generic solution to the detection problem despite of 25

years of research in computer vision for human motion analysis. Often the research scenarios are simplified so that the hand's color or shape can be successfully detected based on, for example, strong restrictions on the image background. An important trend of the last years is the use of machine learning approaches that process large amounts of low-level appearance features to generate automatically trained detection systems (see Section 3.2.3).

Through training the systems with a large number of examples for the appearance features, the color and shape information is implicitly encapsulated in the appearance models instead of the explicit modeling used in earlier approaches for color- and shape-based detection. This makes appearance-based approaches appealing as they allow the handling of arbitrary real world environments as long as the training data contains appropriate images of the settings. In terms of computational demands, the appearance-based approaches are well-suited for mobile robots, as modern machine-learning techniques can operate in real-time on standard PC hardware. A drawback of appearance-based approaches is that they cannot generalize well to untrained gestures. Furthermore, for more complex tasks the detection result is rather probabilistic and these ambiguities need to be handled in the subsequent tracking stage.

Chapter 4 has introduced standard tracking techniques for the incorporation of temporal information into the gesture recognition process. In early gesture recognition approaches that operated in simplified settings, a high quality of single image detection results was secured and the tracking task could be reduced to 'tracking by detection' by applying standard techniques like, e.g., Kalman filtering or particle filtering (see Section 4.2). In more realistic scenarios, however, a detection in every image is usually not possible. A successful approach to cope with this challenge has been to provide top-down feedback from the tracking stage to the detection stage and to adapt the detection algorithms with this top-down information.

Adaptive tracking approaches have initially been developed for use with classical detection methods focussing on the modification of models for visual features like hand shape or skin color (see Section 4.3). More recently, model-based approaches that internally maintain a sophisticated 3D model of the hand or the human for their detection in the 2D images are gaining increasing interest (see Section 4.5). Such adaptive model-based tracking approaches offer the potential benefit of coping with arbitrary motions that do not need to be modeled in advance as the visual models for detection are generated on-the-fly from the internal 3D models. However, the computational demands of model-based tracking approaches are rather high and an appropriate handling of occlusions is still an unsolved issue. Therefore, these adaptive tracking approaches are in direct competition with simple 'tracking by

detection' approaches that are extended to probabilistic versions to cope with the ambiguous detection results from appearance-based machine learning methods.

Especially for use on mobile robots, the Kinect® sensor has lowered the technical barrier significantly for robotics researchers to obtain trajectory data of a gesturing human. The high-quality depth information from the active Kinect® sensor is in limited form also available from a passive stereo camera and has shown its potential, while today no monocular approach achieves a similar performance at a comparable hardware cost. This may lead future vision-based hand detection and tracking research to focus on stereo-based approaches while monocular approaches may see a strong decline.

Based on a successful extraction of the hand trajectory provided by the detection and tracking stages, trajectory-based gesture recognition can be performed. In Chapter 5 relevant methods that have been developed over the years for time series processing and that are now applied successfully to the field of gesture recognition were reviewed. Due to their growing importance in the field, the focus was on probabilistic methods as they have proven to be well-suited to cope with the ambiguities present in the recognition task. Especially graphical models with their ability to represent hierarchical relationships are an important technique to tackle the recognition of more complex gestures consisting of several elementary hand motions. Although Hidden-Markov Models and Dynamic Bayesian Networks are applied heavily in many different approaches, there is a huge variability in the form of the graphical models and the level on which gestures are modeled. Despite the long research in this direction, this variety exemplifies that there is still no common modeling that has proven to be superior for a larger number of gesture recognition applications on robots.

With the techniques covered for detection, tracking, and recognition, the overall task of gesture recognition can be realized. However, the classical decomposition of the task into these steps has effected the nature of the developed recognition approaches. On the one hand, this decomposition has enabled to focus on the specific challenges in each of the individual areas and has created a wealth of approaches for some of the areas. On the other hand, this sequential view of the processing steps has emphasized bottom-up processing schemes which are primarily relevant for gestures that are independent of the context, i.e., where no top-down influences like, e.g., objects in the surroundings have to be considered. Such isolated bottom-up processing is, therefore, reasonable for symbolic and conventional gestures like, e.g., commands.

However, for pointing gestures and manipulative gestures that have a relation to their surroundings the classical bottom-up scheme becomes challenging. For such

gestures, the recognition of the hand motion alone is not enough as the context has to be incorporated into the recognition process to achieve gesture understanding. While research on *gesture recognition* has been very active since the 90's, the incorporation of context in more advanced algorithmic frameworks for *gesture understanding* is only recently gaining increased attention.

Chapter 6 has been devoted to introduce relevant types of context as well as algorithmic frameworks for the inclusion of context into the recognition process in order to achieve gesture understanding. Although initial work on gesture understanding has been conducted at the same time as the gesture recognition research has started, over the years there has been much less emphasis on the context processing because gesture recognition itself turned out to be a hard task. Consequently, progress on gesture understanding was limited and this research direction can be considered to be still in an early stage. However, there have been growing activities in this field in the last years and there is now a large diversity of algorithmic approaches that promise to advance gesture understanding substantially.

As the modular gesture understanding approaches typically build upon the existing methods for gesture recognition, the preceding Chapters have provided the algorithmic background of hand motion processing for introducing in Chapter 6 the methods for incorporating context. In addition to categorizing the different types of context, two approaches for incorporating user-provided context and situational context have served as instances to demonstrate the processing exemplarily. It should be noted that the main focus of gesture understanding research has been on modular approaches as they follow the 'divide and conquer' decomposition and are well suited for imitation learning, i.e., for the transfer of the observed gesture to a robot for reproduction. In contrast, holistic approaches to gesture understanding can only serve to improve the robot's reaction to an observed gesture but do not allow a skill transfer. Nevertheless, the very impressive results of recent holistic approaches to gesture understanding underline that there may be different routes to making robots more intelligent depending on whether the communication or manipulation skills should be enhanced.

From the many approaches on gesture recognition and gesture understanding, only a few ones have actually been implemented and tested on robots. Chapter 7 has provided an overview of robots exhibiting some kind of gesture understanding, but all of the presented robots are research prototypes that operate only in laboratory environments. Consequently, they may have a very limited ability to cope with real-world situations like, e.g., challenging lighting conditions or naive human users that may gesture quite differently from how the experienced researchers instruct 'their' systems. This rather challenging situation and the absence of commercially

successful personal robots are likely to be linked. As soon as the research community can implement robust gesture understanding systems providing natural human-robot interfaces, the demand for personal robots may rise.

Although the algorithms and systems presented in this book achieve impressive functionalities, they are often very brittle. In order to achieve robust performance in natural situations with partial occlusions, there is still substantial research to be carried out not only on the level of context incorporation, but also on the more elementary algorithms for detection and tracking. The versatility of probabilistic frameworks offers here the potential to introduce more top-down processing paths into the overall systems in order to use context information for improving also the elementary algorithms. An important open challenge for future systems will be the coherent view on the whole processing chain, i.e., to support each step of the algorithmic processing in a way that the overall recognition result achieves the best performance. Probabilistic frameworks are currently the most promising algorithmic means for this task, but there is still a long way to go to achieve gestural understanding on robots for a wider range of gestures.

Besides the algorithmic considerations about gesture understanding covered in this book, Chapter 8 has also shortly touched on recent research that focusses stronger on the interactive learning employed in adult-child interactions. Studies have shown that adults modify their behavior when demonstrating actions to children, and similar modifications have also been found in demonstrations towards a robot. Consequently, knowledge of such action modifications could be employed by a robot aiming at learning a new action from a human demonstrator.

Different from the carefully designed algorithms for gesture understanding systems, the insights from developmental learning offer the potential to let future robots learn simply by observing a human demonstrator. While today's technical approaches to imitation learning focus largely on trajectory reproduction, the range of human gestures is much wider than can be captured in the form of the trajectories. The different kinds of feedback that have been found in interactive learning settings between an adult and a humanoid robot give a first glimpse of the potential that insights from developmental learning provide for creating future gesture understanding systems. Combining the set of algorithmic methods presented in this book with the insights from developmental learning is an unsolved but promising research direction that can bring robots closer to exhibit advanced gestural understanding capabilities and, therefore, pave their way to become personal robots.

References

A. Agarwal and B. Triggs. Recovering 3D human pose from monocular images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(1):44–58, 2006.

J. K. Aggarwal and Q. Cai. Human motion analysis: A review. *Computer Vision and Image Understanding*, 73(3):428–440, 1999.

M. Ahad, J. Tan, H. Kim, and S. Ishikawa. Motion history image: its variants and applications. *Machine Vision and Applications*, pages 1–27, 2010. ISSN 0932-8092.

T. Ahmad, C. J. Taylor, A. Lanitis, and T. F. Cootes. Tracking and recognising hand gestures using statistical shape models. *Image and Vision Computing*, 15:345–352, 1997.

Y. Ahn, M. Kim, Y. Park, K. Choi, W. Park, H. Seo, and K. Jung. Implementation of 3D gesture recognition system based on neural network. In *Proceedings of the 9th WSEAS international conference on Applied informatics and communications*, pages 84–87. World Scientific and Engineering Academy and Society (WSEAS), 2009.

Aldebaran Robotics. Nao Interactive Robot, 2011. URL http://www.aldebaran-robotics.com/en/nao_robot_interactif.

J. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11): 832–843, 1983.

R. Arkin, M. Fujita, T. Takagi, and R. Hasegawa. An ethological and emotional basis for human-robot interaction. *Robotics and Autonomous Systems*, 42(3-4):191–201, 2003.

S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Trans. on Signal Processing*, 50(2):174–188, 2002.

T. Axenbeck, M. Bennewitz, S. Behnke, and W. Burgard. Recognizing complex, parameterized gestures from monocular image sequences. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 687–692. IEEE, 2008.

D. Ayers and M. Shah. Monitoring human behavior in an office environment. In *IEEE Workshop on Interpretation of Visual Motion, CVPR*, Santa Barbara, CA, June 1998.

G. Bailador, D. Roggen, G. Tröster, and G. Triviño. Real time gesture recognition using continuous time recurrent neural networks. In *2nd Int. Conf. on Body Area Networks (BodyNets)*, 2007.

Y. Bar-Shalom and T. Fortmann. *Tracking and Data Association*. Academic Press, Inc., San Diego, 1988.

C. Becker, S. Kopp, and I. Wachsmuth. Simulating the emotion dynamics of a multimodal conversational agent. In E. Andre, L. Dybkjaer, W. Minker, and P. Heisterkamp, editors, *Affective Dialogue Systems*, volume 3068 of *Lecture Notes in Computer Science*, pages 154–165. Springer Verlag, 2004.

C. Becker, H. Prendinger, M. Ishizuka, and I. Wachsmuth. Evaluating affective feedback of the 3D agent max in a competitive cards game. In *Int. Conf. on Affective Computing and Intelligent Interaction*, Lecture Notes in Computer Science, pages 466–473. Springer, 2005.

H. Bekel, I. Bax, G. Heidemann, and H. Ritter. Adaptive computer vision: Online learning for object recognition. In *Proceedings DAGM-Symposium*, pages 447–454. Springer, 2004.

S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.

F. Bertsch and V. Hafner. Real-time dynamic visual gesture recognition in human-robot interaction. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 447–453. IEEE, 2009.

A. Billard and R. Siegwart. Special issue on robot programming by demonstration. *Robotics and Autonomous Systems*, 47(2–3), 2004.

A. Billard, S. Calinon, R. Dillmann, and S. Schaal. Survey: Robot Programming by Demonstration. In *Handbook of Robotics*, volume chapter 59. MIT Press, 2008.

R. Bischoff and V. Graefe. Integrating vision, touch and natural language in the control of a situation-oriented behavior-based humanoid robot. In *IEEE International Conference on Systems, Man, and Cybernetics*, volume 2, pages 999–1004. IEEE, 1999.

M. J. Black and A. D. Jepson. A probabilistic framework for matching temporal trajectories: CONDENSATION-based recognition of gestures and expressions. *Lecture Notes in Computer Science*, 1406:909–924, 1998. ISSN 0302-9743.

A. Bobick and Y. Ivanov. Action recognition using probabilistic parsing. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 196–202, Santa Barbara, California, 1998.

A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.

M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 994–999, Puerto Rico, USA, 1997.

R. J. Brand, D. A. Baldwin, and L. A. Ashburn. Evidence for 'motionese': modifications in mothers' infant-directed action. *Developmental Science*, 5(1):72–83, 2002.

M. Bray, E. Koller-Meier, and L. V. Gool. Smart particle filtering for 3D hand tracking. In *Proc. IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pages 675–680, 2004.

C. Breazeal. *Designing sociable robots*. The MIT Press, 2004.

C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, H. Lee, J. Lieberman, A. Lockerd, and D. Chilongo. Humanoid robots as cooperative partners for people. *Int. Journal of Humanoid Robots*, 1(2):315–348, 2004.

T. Breuer, G. Giorgana Macedo, R. Hartanto, N. Hochgeschwender, D. Holz, F. Hegger, Z. Jin, C. Müller, J. Paulus, M. Reckhaus, et al. Johnny: An autonomous service robot for domestic environments. *Journal of Intelligent & Robotic Systems*, pages 1–28, 2011.

B. Burger, F. Lerasle, I. Ferrané, and A. Clodic. Mutual assistance between speech and vision for human-robot interaction. In *IEEE/RSJ Int. Conf. on Robots and Systems*, pages 4011–4016. IEEE, 2008.

L. Campbell, D. Becker, A. Azarbayejani, A. Bobick, and A. Pentland. Invariant features for 3D gesture recognition. In *Proc. IEEE Int. Conf. on Automatic Face and Gesture Recognition*, page 157, Killington, Vermont, USA, 1996. IEEE Computer Society.

T. Cham and J. Rehg. A multiple hypothesis approach to figure tracking. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 239–245, 1999.

C. Chang and R. Ansari. Kernel particle filter for visual tracking. *Signal Processing Letters*, 12 (3):242–245, 2005.

C. Chu and R. Nevatia. Real-time 3D body pose tracking from multiple 2D images. *Articulated Motion and Deformable Objects*, pages 42–52, 2008.

Y. Chuang, L. Chen, G. Zhao, and G. Chen. Hand posture recognition and tracking based on bag-of-words for human robot interaction. In *Proc. Int. Conf. on Robotics and Automation*, pages 538–543, 2011.

COGNIRON. The Cognitive Robot Companion, 2004. URL www.cogniron.org. (FP6-IST-002020).

D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002. ISSN 0162-8828.

T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, Jan. 1995.

T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23:681–685, June 2001. ISSN 0162-8828.

A. Corradini. Dynamic time warping for off-line recognition of a small gesture vocabulary. In *Proceedings of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, Washington, DC, USA, 2001. IEEE Computer Society.

B. Dariush, M. Gienger, A. Arumbakkam, Y. Zhu, B. Jian, K. Fujimura, and C. Goerick. Online transfer of human motion to humanoids. *International Journal of Humanoid Robotics*, 6(2): 265–289, 2009.

T. Darrell and A. Pentland. Space-time gestures. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 335–340, 1993.

K. Dautenhahn and C. L. Nehaniv. The agent-based perspective on imitation. In *Imitation in animals and artifacts*, pages 1–40. MIT Press, 2002. ISBN 0-262-04203-7.

Y. Demiris and A. Billard. Special issue on robot learning by observation, demonstration, and imitation. *IEEE Trans. on Systems, Man, and Cybernetics - Part B: Cybernetics*, 37(2):254–255, 2007.

J. Deutscher and I. Reid. Articulated body motion capture by stochastic search. *International Journal of Computer Vision*, 61(2):185–205, 2005.

R. A. Dewey. Psychology: An introduction, 2011. URL http://www.intropsych.com/ch07_cognition/top-down_and_bottom-up_processing.html.

P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72. IEEE, 2005.

P. Dominey and C. Dodane. Indeterminacy in language acquisition: the role of child directed speech and joint attention. *Journal of Neurolinguistics*, 17:121–145, 2004.

A. Doucet and A. Johansen. *A tutorial on particle filtering and smoothing: Fifteen years later.* Oxford University Press, 2011.

D. Droeschel, J. Stückler, D. Holz, and S. Behnke. Towards joint attention for a domestic service robot–person awareness and gesture recognition using time-of-flight cameras. In *Proc. Int. Conf. on Robotics and Automation*, pages 1205–1210. IEEE, 2011.

D. Efron. *Gesture, Race and Culture.* The Hague: Mouton, 1941/1972.

M. Ehrenmann, P. Steinhaus, and R. Dillmann. A multisensor system for observation of user actions in programming by demonstration. In *IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, 1999.

P. Ekman and W. Friesen. The repertoire of nonverbal behavior: Categories, origins, usage, and coding. *Semiotica*, 1(1):49–98, 1969.

N. J. Emery. The eyes have it: the neuroethology, function and evolution of social gaze. *Neuroscience and Biobehavioral Reviews*, 24:581–604, 2000.

Encyclopedia Britannica. human eye. Encyclopedia Britannica Online, 2011. URL http://www.britannica.com/EBchecked/topic/1688997/human-eye. last access 2011.07.29.

A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly. Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding*, 108(1-2):52–73, 2007. ISSN 1077-3142.

A. Fernald and T. Simon. Expanded intonation contours in mothers' speech to newborns. *Developmental Psychology*, 20(1):104–113, 1984.

S. Fine, Y. Singer, and N. Tishby. The hierarchical hidden markov model: Analysis and applications. *Machine Learning*, 32(1):41–62, 1998.

F. Fleischer, A. Casile, and M. Giese. View-independent recognition of grasping actions with a cortex-inspired model. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 514–519, 2009.

T. W. Fong, I. Nourbakhsh, and K. Dautenhahn. A survey of socially interactive robots. *Robotics and Autonomous Systems*, 42(3–4):143–166, 2003.

Y. Freund and R. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

J. Fritsch. *Vision-based Recognition of Gestures with Context*. PhD thesis, *Faculty of Technology:* Bielefeld University, 2003. URL http://pub.uni-bielefeld.de/publication/2304982.

J. Fritsch, F. Lömker, M. Wienecke, and G. Sagerer. Detecting assembly actions by scene observation. In *Proceedings International Conference on Image Processing*, volume I, pages 212–215, Vancouver, Sep. 2000. IEEE.

J. Fritsch, S. Lang, M. Kleinehagenbrock, G. A. Fink, and G. Sagerer. Improving adaptive skin color segmentation by incorporating results from face detection. In *Proc. IEEE Int. Workshop on Robot-Human Interactive Communication*, pages 337–343, Berlin, Germany, September 2002.

J. Fritsch, M. Kleinehagenbrock, S. Lang, T. Plötz, G. A. Fink, and G. Sagerer. Multi-modal anchoring for human-robot-interaction. *Robotics and Autonomous Systems, Special issue on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*, 43(2–3):133–147, 2003.

J. Fritsch, N. Hofemann, and G. Sagerer. Combining sensory and symbolic data for manipulative gesture recognition. In *Proc. Int. Conf. on Pattern Recognition*, pages 930–933, Cambridge, United Kingdom, 2004. IEEE.

T. Fukuda, Y. Nakauchi, K. Noguchi, and T. Matsubara. Sequential human behavior recognition for cooking-support robots. *Journal of Robotics and Mechatronics*, 17(6):717, 2005.

B. Funt, K. Barnard, and L. Martin. Is machine colour constancy good enough? *Lecture Notes in Computer Science (ECCV 98)*, 1406:445–459, 1998.

D. Gavrila. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1):82–98, 1999.

D. Gehrig, H. Kuehne, A. Woerner, and T. Schultz. HMM-based human motion recognition with optical flow data. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 425–430, 2009.

D. Gehrig, P. Krauthausen, L. Rybok, H. Kuehne, U. Hanebeck, T. Schultz, and R. Stiefelhagen. Combined intention, activity, and motion recognition for a humanoid household robot. In *IEEE/RSJ Int. Conf. on Robots and Systems*, pages 4819–4825, 2011.

G. Gergely. What should a robot learn from an infant? Mechanisms of action interpretation and observational learning in infancy. In *Proc. Third Int. Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*, pages 13–24, Boston, MA, USA, 2003.

Z. Ghahramani. *An introduction to hidden Markov models and Bayesian networks*, pages 9–42. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2002.

S. S. Ghidary, Y. Nakata, H. Saito, M. Hattori, and T. Takamori. Multi-modal interaction of human and home robot in the context of room map generation. *Autonomous Robots*, 13(2): 169–184, 2002.

L. Gogate, L. Bahrick, and J. Watson. A study of multimodal motherese: The role of temporal synchrony between verbal labels and gestures. *Child Development*, 71:878 – 894, 2000.

R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001. ISBN 0201180758.

A. Gupta and L. Davis. Objects in action: An approach for combining action understanding and object perception. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

A. Gupta, P. Srinivasan, J. Shi, and L. Davis. Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 2004–2011, 2009.

A. Haasch. *Attention-controlled acquisition of a qualitative scene model for mobile robots*. Phd thesis, *Faculty of Technology:* Bielefeld University, 2007. URL http://pub.uni-bielefeld.de/publication/2304982.

A. Haasch, N. Hofemann, J. Fritsch, and G. Sagerer. A multi-modal object attention system for a mobile robot. In *IEEE/RSJ Int. Conf. on Robots and Systems*, pages 1499–1504, Edmonton, Alberta, Canada, August 2005. IEEE.

M. Hahn, L. Krüger, and C. Wöhler. Spatio-temporal 3D pose estimation and tracking of human body parts using the shape flow algorithm. In *Proc. Int. Conf. on Pattern Recognition*, pages 1–4. IEEE, 2008.

M. Hahn, L. Krüger, C. Wöhler, and F. Kummert. 3D action recognition in an industrial environment. In *Proc. 3rd Workshop on Human Centered Robot Systems*, volume 6 of *Cognitive Systems Monographs*, pages 141–150. Springer, 2009.

A. Hakeem, Y. Sheikh, and M. Shah. Case-e: A hierarchical event representation for the analysis of videos. In *Proc. of the national conference on artificial intelligence*, pages 263–268. AAAI Press; MIT Press, 2004.

B. Han, Y. Zhu, D. Comaniciu, and L. Davis. Kernel-based bayesian filtering for object tracking. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 227–234, 2005.

M. Hasanuzzaman, T. Zhang, V. Ampornaramveth, H. Gotoda, Y. Shirai, and H. Ueno. Adaptive visual gesture recognition for human-robot interaction using a knowledge-based software platform. *Robotics and Autonomous Systems*, 55(8):643 – 657, 2007. ISSN 0921-8890.

T. Heap and D. Hogg. Towards 3D hand tracking using a deformable model. In *Proc. IEEE Int. Conf. on Automatic Face and Gesture Recognition*, FG '96, pages 140–145, Washington, DC, USA, 1996. IEEE Computer Society. ISBN 0-8186-7713-9.

F. Hecht, P. Azad, and R. Dillmann. Markerless human motion tracking with a flexible model and appearance learning. In *Proc. Int. Conf. on Robotics and Automation*, pages 3173–3179, 2009.

F. Hegel, M. Lohse, and B. Wrede. Effects of visual appearance on the attribution of applications in social robotics. In *Proc. IEEE Int. Workshop on Robot-Human Interactive Communication*, pages 64–71. IEEE, 2009.

G. Heidemann, H. Bekel, I. Bax, and A. Saalbach. Hand gesture recognition: self-organising maps as a graphical user interface for the partitioning of large training data sets. In *Proc. Int. Conf. on Pattern Recognition*, volume 4, pages 487–490, 2004.

M. Hirose, Y. Haikawa, T. Takenaka, and K. Hirai. Development of humanoid robot ASIMO. In *IEEE/RSJ Int. Conf. on Robots and Systems, Workshop on Explorations towards Humanoid Robot Applications*, Maui, Hawaii, 2001.

N. Hofemann. *Videobasierte Handlungserkennung für die natürliche Mensch-Maschine-Interaktion*. Phd thesis, *Faculty of Technology:* Bielefeld University, Bielefeld, 2007. URL http://pub.uni-bielefeld.de/publication/2305486.

D. Hogg. Model-based vision: a program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983.

M. Holte, T. Moeslund, and P. Fihl. View invariant gesture recognition using the csem swissranger sr-2 camera. *International Journal of Intelligent Systems Technologies and Applications*, 5(3): 295–303, 2008.

S. Horimoto, D. Arita, and R. Taniguchi. Real-time hand shape recognition for human interface. In *Int. Conf. on Image Analysis and Processing*, 2003.

J. Ijsselmuiden and R. Stiefelhagen. Towards high-level human activity recognition through computer vision and temporal logic. *KI 2010: Advances in Artificial Intelligence*, pages 426–435, 2010.

K. Irie, N. Wakamura, and K. Umeda. Construction of an intelligent room based on gesture recognition: operation of electric appliances with hand gestures. In *IEEE/RSJ Int. Conf. on Robots and Systems*, volume 1, pages 193–198, 2004.

M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. *Lecture Notes in Computer Science*, 1064:343–356, 1996. ISSN 0302-9743.

M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998.

Y. Ivanov and A. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8):852–872, 2000.

M. Iwasawa, Y. Fukusato, E. Sato-Shimokawara, and T. Yamaguchi. Obtaining an object position using multimodal interaction for a service robot. In *Proc. IEEE Int. Workshop on Robot-Human Interactive Communication*, pages 1155–1160. IEEE, 2009.

H. F. J.-C. Terrillon, M. N. Shirazi and S. Akamatsu. Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images. In *Proc. IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pages 54–61, Grenoble, France, 2000.

M. Johnson-Robertson, J. Bohg, G. Skantze, J. Gustafson, R. Carlson, B. Rasolzadeh, and D. Kragic. Enhanced visual scene understanding through human-robot dialog. In *Proc. Int. Conf. on Robotics and Automation*, pages 3342–3348, 2011.

M. J. Jones and J. M. Rehg. Statistical color models with application to skin detection. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 274–280, Ft. Collins, CO, 1999.

P. Kakumanu, S. Makrogiannis, and N. Bourbakis. A survey of skin-color modeling and detection methods. *Pattern Recognition*, 40(3):1106–1122, 2007.

S.-P. Kang, M. Tordon, and J. Katupitiya. Curvature based hand shape recognition for a virtual wheelchair control interface. In *Proc. Int. Conf. on Robotics and Automation*, pages 2049–2054, 2004.

F. Kaplan and V. V. Hafner. The challenges of joint attention. In *Proc. 4th Int. Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic System*, pages 67–74, Lund University Cognitive Studies, 2004.

Kawada Industries. HRP-3 Promet, 2011. URL http://global.kawada.jp/mechatronics/hrp3.html.

D. Kawanaka, T. Okatani, and K. Deguchi. HHMM based recognition of human activity. *IEICE transactions on information and systems*, 89(7):2180–2185, 2006.

R. Kehl, M. Bray, and L. V. Gool. Markerless full body tracking by integrating multiple cues. In *ICCV Workshop on Modeling People and Human Interaction*, 2005.

A. Kendon. Studies in dyadic communication. *Studies in dyadic communication*, 7:177–216, 1972.

A. Kendon. *Gesture: Visible action as utterance*. Cambridge University Press, 2004.

A. Kendon and C. Mueller, editors. *Gesture*. John Benjamins Publishing Co., Netherlands, 2005.

H. Kim, S. Kim, and S. Park. Pointing gesture-based unknown object extraction for learning objects with robot. In *International Conference on Control, Automation and Systems*, pages 2156–2161. IEEE, 2008.

H. Kjellström, J. Romero, D. Martinez, and D. Kragic. Simultaneous visual recognition of manipulation actions and manipulated objects. In D. Forsyth, P. Torr, and A. Zisserman, editors, *Computer Vision ECCV 2008*, volume 5303 of *Lecture Notes in Computer Science*, pages 336–349. Springer Berlin / Heidelberg, 2008.

J. Kober and J. Peters. Imitation and reinforcement learning. *Robotics & Automation Magazine*, 17(2):55–62, 2010.

M. Kölsch and M. Turk. Robust hand detection. In *Proc. IEEE Int. Conf. on Automatic Face and Gesture Recognition*, 2004.

S. Kopp and I. Wachsmuth. Synthesizing multimodal utterances for conversational agents. *Computer animation and virtual worlds*, 15(1):39–52, 2004.

D. Kortenkamp, E. Huber, and R. P. Bonasso. Recognizing and interpreting gestures on a mobile robot. In *Proc. American Association Conf. of Artificial Intelligence*, pages 915–921, 1996.

A. Kranstedt and I. Wachsmuth. Incremental generation of multimodal deixis referring to objects. In *European Workshop on Natural Language Generation (ENLG)*, pages 75–82, 2005.

A. Kranstedt, A. Lücking, T. Pfeiffer, H. Rieser, and I. Wachsmuth. Deictic object reference in task-oriented dialogue. In G. Rickheit and I. Wachsmuth, editors, *Situated Communication*, volume 166, pages 135–189. Walter de Gruyter, Berlin, 2006.

H. Kruppa, M. A. Bauer, and B. Schiele. Skin patch detection in real-world images. In *DAGM Symposium*, pages 109–116, Zurich, Switzerland, September 2002.

Y. Kuniyoshi and H. Inoue. Qualitative recognition of ongoing human action sequences. In *Proc. Int. Joint Conf. on Artificial Intelligence*, pages 1600–1609, 1993.

Y. Kuniyoshi, M. Inaba, and H. Inoue. Learning by Watching: Extracting Reusable Task Knowledge from Visual Observation of Human Performance. *IEEE Trans. on Robotics and Automation*, 10(6):799–822, 1994.

I. Laptev and T. Lindeberg. Tracking of multi-state hand models using particle filtering and a hierarchy of multi-scale image features. In *IEEE Workshop on Scale-Space and Morphology*, Lecture Notes in Computer Science, pages 63–74, Vancouver, Canada, 2001. Springer Verlag.

I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

H. Li and M. Greenspan. Multi-scale gesture recognition from time-varying contours. In *Proc. Int. Conf. on Computer Vision*, volume 1, pages 236–243. IEEE, 2005.

Z. Li. *Vision-based Manipulative Gesture Recognition in a Human-Robot Interaction Scenario*. Phd thesis, *Faculty of Technology:* Bielefeld University, Bielefeld, 2008.

Z. Li, J. Fritsch, S. Wachsmuth, and G. Sagerer. An object-oriented approach using a top-down and bottom-up process for manipulative action recognition. In K. Franke, K.-R. Müller, B. Nickolay, and R. Schäfer, editors, *Lecture Notes in Computer Science*, volume 4174, pages 212–221, Heidelberg, Germany, 2006. Springer-Verlag.

Z. Li, S. Wachsmuth, J. Fritsch, and G. Sagerer. View-adaptive manipulative action recognition for robot companions. In *IEEE/RSJ Int. Conf. on Robots and Systems*, San Diego, CA, USA, October 2007.

A. Licsar and T. Sziranyi. User-adaptive hand gesture recognition system with interactive training. *Image and Vision Computing*, 23(12):1102–1114, Nov. 2005.

F. Lömker and G. Sagerer. A multimodal system for object learning. In L. V. Gool, editor, *Proceedings DAGM-Symposium*, Lecture Notes in Computer Science 2449, pages 490–497, Berlin, September 2002. Springer.

L. Lu, G. Hager, and L. Younes. A three tiered approach for articulated object action modeling and recognition. In *Advances in Neural Information Processing Systems*, pages 841–848, 2005.

J. MacCormick and M. Isard. Partitioned sampling, articulated objects, and interface-quality hand tracking. In *Proc. European Conference on Computer Vision*, volume 2, pages 3–19, 2000.

K. MacDorman and H. Ishiguro. The uncanny advantage of using androids in cognitive and social science research. *Interaction Studies*, 7(3):297–337, 2006.

R. Mann, A. Jepson, and J. Siskind. The computational perception of scene dynamics. *Computer Vision and Image Understanding*, 65(2):113–128, 1997.

C. Martin, F. Steege, and H. Gross. Estimation of pointing poses for visually instructing mobile robots under real world conditions. *Robotics and Autonomous Systems*, 58(2):174–185, 2010.

J. Maycock, J. Steffen, R. Haschke, and H. J. Ritter. Robust tracking of human hand postures for robot teaching. In *IEEE/RSJ Int. Conf. on Robots and Systems*, San Francisco, 2011. IEEE.

D. McNeill. *Hand and mind: What gestures reveal about thought.* University of Chicago Press, 1996.

MESA Imaging. Swissranger sr-3000, 2011. URL www.mesa-imaging.ch.

D. Miaw. Second skin : motion capture with actuated feedback for motor learning. Master's thesis, Massachusetts Institute of Technology. Dept. of Electrical Engineering and Computer Science., 2010.

Microsoft. Kinect sensor for xbox 360, 2011. URL www.xbox.com/de-DE/Kinect/.

S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, 37(3):311–324, 2007.

P. Modler and T. Myatt. Recognition of separate hand gestures by time-delay neural networks based on multi-state spectral image patterns from cyclic hand movements. In *Proc. IEEE Int. Conf. Systems, Man and Cybernetics SMC 2008*, pages 1539–1544, 2008.

T. B. Moeslund and E. Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231–268, 2001.

T. B. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2):90–126, 2006. ISSN 1077-3142.

M. A. Moni and A. B. M. S. Ali. HMM based hand gesture recognition: A review on techniques and approaches. *Computer Science and Information Technology, International Conference on*, 0:433–437, 2009.

D. Moore and I. Essa. Recognizing multitasked activities from video using stochastic context-free grammar. In *Proceedings of the National Conference on Artificial Intelligence*, pages 770–776. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2002.

D. Moore, I. Essa, and M. Hayes. Exploiting human actions and object context for recognition tasks. In *Proc. Int. Conf. on Computer Vision*, volume 1, pages 80–86, Corfu, 1999.

J.-M. Morel, A. B. Petro, and C. Sbert. Fast implementation of color constancy algorithms. In R. Eschbach, G. G. Marcu, S. Tominaga, and A. Rizzi, editors, *Color Imaging XIV: Displaying, Processing, Hardcopy, and Applications*, volume 7241. SPIE, 2009.

D. Morris, P. Collett, P. Marsh, and M. O'Shaughnessy. *Gestures: Their Origins and Distribution.* Jonathan Cape, London, 1979. ISBN 0224015702.

M. Mühlig, M. Gienger, S. Hellbach, J. Steil, and C. Goerick. Task-level imitation learning using variance-based movement optimization. In *Proc. Int. Conf. on Robotics and Automation*, pages 1177–1184. IEEE, 2009.

M. Mühlig, M. Gienger, and J. Steil. Human-robot interaction for learning and adaptation of object movements. In *IEEE/RSJ Int. Conf. on Robots and Systems*, pages 4901–4907. IEEE, 2010.

P. Natarajan and R. Nevatia. Coupled hidden semi markov models for activity recognition. In *IEEE Workshop on Motion and Video Computing (WMVC)*. IEEE Computer Society, 2007.

C. P. Nehaniv. Classifying types of gesture and inferring intent. In *Proceedings of the Symposium on Robot Companions: Hard problems and Open Challenges in Robot-Human Interaction AISB'05*, pages 74–81, Hatfield, UK, 2005.

R. Nevatia, T. Zhao, and S. Hongeng. Hierarchical language-based representation of events in video streams. In *Computer Vision and Pattern Recognition Workshop, 2003. CVPRW'03. Conference on*, volume 4, pages 39–39. IEEE, 2003.

K. Nickel and R. Stiefelhagen. Visual recognition of pointing gestures for human-robot interaction. *Image and Vision Computing*, 25(12):1875–1884, 2007.

J. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision*, 79(3):299–318, 2008.

C. Nölker and H. Ritter. GREFIT: Visual recognition of hand postures. In A. Braffort, R. Gherbi, S. Gibet, J. Richardson, and D. Teil, editors, *Gesture-Based Communication in Human-Computer Interaction: Proc. International Gesture Workshop, GW '99, France*, pages 61–72. Springer Verlag, LNAI 1739, 1999.

N. Oliver, A. Pentland, and F. Berard. Lafter: a real-time face and lips tracker with facial expression recognition. *Pattern Recognition*, 33:1369–1382, 2000.

M. Pardowitz, S. Knoop, R. Dillmann, and R. D. Zöllner. Incremental learning of tasks from user demonstrations, past experiences, and vocal comments. *IEEE Trans. on Systems, Man, and Cybernetics - Part B: Cybernetics*, 37, 2007.

V. Pavlovic, R. Sharma, and T. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(7):677–695, 1997.

T. Pfeiffer, M. Latoschik, and I. Wachsmuth. Conversational pointing gestures for virtual reality interaction: Implications from an empirical study. In *Proceedings of the IEEE Virtual Reality 2008*, pages 281–282, 2008.

S. L. Phung, A. Bouzerdoum, and D. Chai. Skin segmentation using color pixel classification: Analysis and comparison. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(1): 148–154, 2005. ISSN 0162-8828.

PMDTec. Pmd [vision], 2011. URL www.pmdtec.de.

G. Pons-Moll, L. Leal-Taixe, T. Truong, and B. Rosenhahn. Efficient and robust shape matching for model based human motion capture. In *Proceedings DAGM-Symposium*, pages 416–425. Springer, 2011.

R. Poppe. Vision-based human motion analysis: An overview. *Computer Vision and Image Understanding*, 108(1-2):4–18, 2007. ISSN 1077-3142.

L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286, 1989.

Y. Raja, S. J. McKenna, and S. Gong. Colour model selection and adaptation in dynamic scenes. In *Proc. European Conference on Computer Vision*, pages 460–474, 1998.

D. Ramanan and D. A. Forsyth. Finding and tracking people from the bottom up. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 467–474, 2003.

J. Rehg and T. Kanade. Digiteyes: Vision-based hand tracking for human-computer interaction. In *Proc. of the Workshop on Motion of Non-Rigid and Articulated Bodies*, pages 16–24, 1994.

J. Richarz, A. Scheidig, C. Martin, S. Müller, and H. Gross. A monocular pointing pose estimator for gestural instruction of a mobile robot. *International Journal of Advanced Robotic Systems*, 4(1):139–150, 2007.

H. Rieser. Pointing in dialogue. *Catalog*, 4:93–101, 2004.

H. Rieser and M. Poesio. Interactive gesture in dialogue: a ptt model. In *Proceedings of the SIGDIAL 2009 Conference: The 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 87–96. Association for Computational Linguistics, 2009.

K. J. Rohlfing, J. Fritsch, B. Wrede, and T. Jungmann. How can multimodal cues from child-directed interaction reduce learning complexity in robots? *Advanced Robotics*, 20(10):1183–1199, 2006.

A. Roussos, T. S., P. V., and P. Maragos. Affine-invariant modeling of shape-appearance images applied on sign language handshape classification. In *Proceedings International Conference on Image Processing*, 2010.

M. Ryoo and J. Aggarwal. Recognition of composite human activities through context-free grammar based representation. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2006.

M. Ryoo and J. Aggarwal. Hierarchical recognition of human activities interacting with objects. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

A. Sadeghipour and S. Kopp. Embodied gesture processing: Motor-based integration of perception and action in social artificial agents. *Cognitive Computation*, 3(3):419–435, 2011.

Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura. The intelligent asimo: system overview and integration. In *IEEE/RSJ Int. Conf. on Robots and Systems*, volume 3, pages 2478 – 2483, 2002.

H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978.

J. Saldien, K. Goris, B. Vanderborght, J. Vanderfaeillie, and D. Lefeber. Expressing emotions with the social robot probo. *Int J Soc Robot*, 2(4):377–389, 2010.

Y. Sato, Y. Kobayashi, and H. Koike. Fast tracking of hands and fingertips in infrared images for augmented desk interface. In *Proc. IEEE Int. Conf. on Automatic Face and Gesture Recognition*, page 462, 2000.

B. Schauerte, J. Richarz, and G. A. Fink. Saliency-based identification and recognition of pointed-at objects. In *IEEE/RSJ Int. Conf. on Robots and Systems*, pages 4638–4643, 2010.

J. Schmidt. *Vision-based Posture Detection and Tracking for Interactive Scenarios*. Phd thesis, *Faculty of Technology:* Bielefeld University, Bielefeld, 2009.

J. Schmidt and M. Castrillon. Automatic initialization for body tracking - using appearance to learn a model for tracking human upper body motions. In *3rd International Conference on Computer Vision Theory and Applications (VISAPP)*, 2008.

J. Schmidt, B. Kwolek, and J. Fritsch. Kernel particle filter for real-time 3D body tracking in monocularcolor images. In *Proc. IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pages 567–572, Southampton, UK, April 2006. IEEE.

J. Schmüdderich, H. Brandl, B. Bolder, M. Heracles, H. Janssen, I. Mikhailova, and C. Goerick. Organizing multimodal perception for autonomous learning and interactive systems. In *IEEE-RAS Int. Conf. on Humanoid Robots*, pages 312–319, 2008.

C. Schüldt, I. Laptev, and B. Caputo. Recognizing human actions: A local svm approach. In *Proc. Int. Conf. on Pattern Recognition*, volume 3, pages 32–36. IEEE, 2004.

C. Shan, T. Tan, and Y. Wei. Real-time hand tracking using a mean shift embedded particle filter. *Pattern Recognition*, 40(7):1958 – 1970, 2007. ISSN 0031-3203.

T. Shibata, K. Wada, Y. Ikeda, and S. Sabanovic. Cross-cultural studies on subjective evaluation of a seal robot. *Advanced Robotics*, 23(4):443–458, 2009.

N. Shimada, K. Kimura, and Y. Shirai. Real-time 3D hand posture estimation based on 2D appearance retrieval using monocular camera. In *IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, pages 23–30, 2001.

H.-K. Shin, S.-W. Lee, and S.-W. Lee. Real-time gesture recognition using 3D motion history model. In *Advances in Intelligent Computing*, volume 3644 of *Lecture Notes in Computer Science*, pages 888–898. Springer Berlin / Heidelberg, 2005.

J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-Time human pose recognition in parts from a single depth image. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, 2011.

H. Sidenbladh. *Probabilistic Tracking and Reconstruction of 3D Human Motion in Monocular Video Sequences*. PhD thesis, KTH Sweden, 2001.

H. Sidenbladh, M. Black, and D. Fleet. Stochastic tracking of 3D human figures using 2D image motion. In *Proc. European Conference on Computer Vision*, pages 702–718, 2000.

L. Sigal, S. Sclaroff, and V. Athitsos. Estimation and prediction of evolving color distributions for skin segmentation under varying illumination. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 152–159, 2000.

L. Sigal, S. Bhatia, S. Roth, M. J. Black, and M. Isard. Tracking loose-limbed people. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 421–428, 2004.

M. Sigalas, H. Baltzakis, and P. Trahanias. Gesture recognition based on arm tracking for human-robot interaction. In *IEEE/RSJ Int. Conf. on Robots and Systems*, pages 5424–5429, 2010.

R. Singh, B. Seth, and U. B. Desai. A real-time framework for vision based human robot interaction. In *IEEE/RSJ Int. Conf. on Robots and Systems*, pages 5831–5836, 2006.

D. Skočaj, M. Kristan, A. Vrečko, M. Mahnič, M. Janicek, G.-J. M. Kruijff, M. Hanheide, N. Hawes, T. Keller, M. Zillich, and Z. Zhou. A system for interactive learning in dialogue with a tutor. In *IEEE/RSJ Int. Conf. on Robots and Systems*, 2011.

A. Sloman and M. Croucher. Why robots will have emotions. In *In Proc 7th Int. Joint Conference on AI*, pages 197–202, 1981.

C. Sminchisescu and B. Triggs. Mapping minima and transitions of visual models. *International Journal of Computer Vision*, 61(1), 2005.

M. Soriano, B. Martinkauppi, S. Huovinen, and M. Laaksonen. Skin detection in video under changing illumination conditions. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 839–842, 2000.

T. Starner and A. Pentland. Visual recognition of american sign language using hidden markov models. In *International Workshop on Automatic Face and Gesture Recognition*, pages 189–194, Zurich, Switzerland, 1995.

T. Starner, J. Weaver, and A. Pentland. Real-time american sign language recognition using desk and wearable computer based video. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(12):1371–1375, 1998.

B. Stenger, P. R. S. Mendonca, and R. Cipolla. Model-based 3D tracking of an articulated hand. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, volume 2, 2001.

R. Stiefelhagen, H. Ekenel, C. Fugen, P. Gieselmann, H. Holzapfel, F. Kraft, K. Nickel, M. Voit, and A. Waibel. Enabling multimodal human–robot interaction for the karlsruhe humanoid robot. *IEEE Transactions on Robotics*, 23(5):840–851, 2007.

R. Stiefelhagen, K. Bernardin, H. Ekenel, and M. Voit. Tracking identities and attention in smart environments-contributions and progress in the CHIL project. In *Proc. IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pages 1–8. IEEE, 2008.

M. Störring, H. J. Andersen, and E. Granum. Skin colour detection under changing lighting conditions. In H. Araújo and J. Dias, editors, *SIRS'99 Proc. 7th Int. Symposium on Intelligent Robotic Systems*, pages 187–195, July 1999.

M. Störring, H. J. Andersen, and E. Granum. Physics-based modelling of human skin colour under mixed illuminants. *Robotics and Autonomous Systems*, 35(3-4):131–142, 2001.

H.-I. Suk, B.-K. Sin, and S.-W. Lee. Hand gesture recognition based on dynamic bayesian network framework. *Pattern Recognition*, 43:3059–3072, September 2010.

G. Ten Holt, M. Reinders, and E. Hendriks. Multi-dimensional dynamic time warping for gesture recognition. In *Proc. of the conference of the Advanced School for Computing and Imaging (ASCI 2007)*, 2007.

J. Triesch and C. von der Malsburg. A system for person-independent hand posture recognition against complex backgrounds. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23 (12):1449–1453, 2001.

W. Tsai, C. Lin, S. Yang, M. Sheu, and C. Su. Adaptive motion gesture segmentation. In *The 2008 International Conference on Embedded Software and Systems Symposia (ICESS2008)*, pages 386–391. IEEE, 2008.

Z. Tu and A. Yuille. Shape matching and recognition: Using generative models and informative features. In *Proc. European Conference on Computer Vision*, pages 195–209, 2004.

M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuro Science*, 3(1): 71–86, 1991.

M. M. Ullah, S. N. Parizi, and I. Laptev. Improving bag-of-features action recognition with non-local cues. In *Proc. British Machine Vision Conference*, pages 95.1–11, 2010. ISBN 1-901725-38-3.

A. Veeraraghavan, R. Chellappa, and A. K. Roy-Chowdhury. The function space of an activity. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 959–968, 2006.

P. Viola and M. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

C. Vogler and D. Metaxas. A framework for recognizing the simultaneous aspects of american sign language. *Computer Vision and Image Understanding*, 81(3):358–384, 2001.

A. Vollmer. *Measurement and Analysis of Interactive Behavior in Tutoring Action with Children and Robots*. Phd thesis, Bielefeld University, Bielefeld, 2011.

A.-L. Vollmer, K. Lohan, K. Fischer, Y. Nagai, K. Pitsch, J. Fritsch, K. Rohlfing, and B. Wrede. People modify their tutoring behavior in robot-directed interaction for action learning. In *Proc. Int. Conf. on Development and Learning*, volume 8, Shanghai, China, 2009. IEEE.

C. von Hardenberg and F. Berard. Bare-hand human-computer interaction. In *ACM workshop on Perceptive User Interfaces (PUI 2001)*, Orlando, Florida, 2001.

M. Walters, D. Syrdal, K. Dautenhahn, R. Te Boekhorst, and K. Koay. Avoiding the uncanny valley: robot appearance, personality and consistency of behavior in an attention-seeking home scenario for a robot companion. *Autonomous Robots*, 24(2):159–178, 2008.

C. Wang and K. Wang. Hand posture recognition using adaboost with sift for human robot interaction. *Recent Progress in Robotics: Viable Robotic Service to Human*, pages 317–329, 2008.

L. Wang, W. Hu, and T. Tan. Recent developments in human motion analysis. *Pattern Recognition*, 36:585–601, 2003.

R. Y. Wang and J. Popović. Real-time hand-tracking with a color glove. *ACM Trans. Graph.*, 28:63:1–63:8, July 2009. ISSN 0730-0301.

W.-H. A. Wang and C.-L. Tung. Dynamic gesture recognition based on dynamic bayesian networks. *WSEAS Transactions on Business and Economics*, 4(11), 2007.

V. Willert, J. Eggert, J. Adamy, and E. Koerner. Non-gaussian velocity distributions integrated over space, time and scales. *IEEE Trans. on Systems, Man, and Cybernetics - Part B: Cybernetics*, 36(03):482–493, 2006.

C. Wöhler. *3D computer vision: efficient methods and applications*, chapter 7: Applications to safe human-robot interaction. Springer-Verlag Berlin Heidelberg, 2009.

S. Wrede, M. Hanheide, C. Bauckhage, and G. Sagerer. An Active Memory as a Model for Information Fusion. In *Proc. 7th Int. Conf. on Information Fusion*, pages 198–205, 2004.

C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Analysis and Machine Vision*, 19(7):780–785, 1997.

Y. Wu and T. S. Huang. Vision-based gesture recognition: A review. In *3rd Gesture Workshop*, pages 103–115, Gif-sur-Yvette, France, 1999.

Y. Wu, J. Lin, and T. S. Huang. Analyzing and capturing articulated hand motion in image sequences. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(12):1910–1922, 2005. ISSN 0162-8828.

W. Wundt. *The language of gesture.* The Hague: Mouton, 1900/1973.

X. Wuo, D. Xu, L. Duan, and J. Luo. Action recognition using context and appearance distribution features. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 489–496, 2011.

M. Yamamoto, H. Mitomi, F. Fujiwara, and T. Sato. Bayesian classification of task-oriented actions based on stochastic context-free grammar. In *Proc. IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pages 317–322. IEEE, 2006.

J. Yang, W. Lu, and A. Waibel. Skin-color modeling and adaptation. *Lecture Notes in Computer Science*, 1352:687–694, 1998.

M.-H. Yang and N. Ahuja. Recognizing hand gesture using motion trajectories. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, volume 1, 1999.

T. Yu, T. Kim, and R. Cipolla. Real-time action recognition by spatiotemporal semantic and structural forest. In *Proc. British Machine Vision Conference*, pages 52.1–12, 2010.

B. Zarit, B. Super, and F.K.H.Quek. Comparison of five color models in skin pixel classification. In *Proc. of Int. Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, pages 58–63, Corfu, Greece, 1999.

C. Zhu and W. Sheng. Online hand gesture recognition using neural network based segmentation. In *IEEE/RSJ Int. Conf. on Robots and Systems*, pages 2415–2420. IEEE, 2009.

C. Zhu and W. Sheng. Realtime recognition of complex daily activities using dynamic bayesian network. In *IEEE/RSJ Int. Conf. on Robots and Systems*, pages 3395–3401, 2011.

J. Ziegler, K. Nickel, and R. Stiefelhagen. Tracking of the articulated upper body on multi-view stereo image sequences. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 774–781. IEEE, 2006.