Imperial College of Science, Technology and Medicine
Department of Computing

Multi-Sensor Fusion for Human-Robot Interaction in Crowded Environments

Stephen John McKeague

Supervised by Professor Guang-Zhong Yang

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Computing of Imperial College London, March 2015

Abstract

For challenges associated with the ageing population, robot assistants are becoming a promising solution. Human-Robot Interaction (HRI) allows a robot to understand the intention of humans in an environment and react accordingly. This thesis proposes HRI techniques to facilitate the transition of robots from lab-based research to real-world environments.

The HRI aspects addressed in this thesis are illustrated in the following scenario: an elderly person, engaged in conversation with friends, wishes to attract a robot's attention. This composite task consists of many problems. The robot must detect and track the subject in a crowded environment. To engage with the user, it must track their hand movement. Knowledge of the subject's gaze would ensure that the robot doesn't react to the wrong person. Understanding the subject's group participation would enable the robot to respect existing human-human interaction.

Many existing solutions to these problems are too constrained for natural HRI in crowded environments. Some require initial calibration or static backgrounds. Others deal poorly with occlusions, illumination changes, or real-time operation requirements. This work proposes algorithms that fuse multiple sensors to remove these restrictions and increase the accuracy over the state-of-the-art.

The main contributions of this thesis are:

1) A hand and body detection method, with a probabilistic algorithm for their real-time association when multiple users and hands are detected in crowded environments.

2) An RGB-D sensor-fusion hand tracker, which increases position and velocity accuracy by combining a depth-image based hand detector with Monte-Carlo updates using colour images.

3) A sensor-fusion gaze estimation system, combining IR and depth cameras on a mobile robot to give better accuracy than traditional visual methods, without the constraints of traditional IR techniques.

4) A group detection method, based on sociological concepts of static and dynamic interactions, which incorporates real-time gaze estimates to enhance detection accuracy.

Declaration of Originality

I hereby declare that the work presented in this this thesis is my own, except where appropriately referenced.

Stephen John McKeague

Dedication

For God so loved the world, that he gave his only begotten Son, that whosoever believeth in him should not perish, but have everlasting life

John 3:16

To my family, past and present

To Marta

Acknowledgements

First and foremost I would like to thank Professor Guang-Zhong Yang for giving me the opportunity of doing a Ph. D. under his supervision. I am also very grateful for his financial support and his trust in my studies.

I would like to offer my heartfelt gratitude to the members of the Grundy Educational Trust for their generous financial aid during my time at Imperial. To Jindong, who stayed awake till the early hours of the morning drafting my late conference submissions, thank you. Thanks, as well, to Javier, whose friendship, advice and knowledge I found invaluable. Indeed, the friendship I have been shown by all those in the Hamlyn Centre, including Piyamate, Ed, Charence and Alex, has made my time here a rewarding experience. To all the people at Calvary Chapel Westminster, I am very grateful for the kindness you have shown to me.

I wish to recognise the support and love, before and throughout my Ph. D., that I have been shown by my family and by my fiancée, Marta. Finally, I would be amiss in not making special mention of my mother, Janice McKeague, and father, Brendan Hugh McKeague, who have invested so much in me. Thank you.

List of Acronyms

HRI	Human-Robot Interaction
RGB	Red-Green-Blue
LED	Light-Emitting Diode
TOF	Time of Flight
SVM	Support Vector Machine
HMM	Hidden Markov Model
PCA	Principal Component Analysis
SVD	Singular Value Decomposition
BGS	Background Subtraction
ROS	Robot Operating System
RAM	Random-Access Memory
KB	Kilobyte
MB	Megabyte
GB	Gigabyte
MHz	Megahertz
GHz	Gigahertz

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Background

In the modern world, robots are used in a wide range of environments for a diverse number of applications. Currently, the most successful robotic applications are in industrial plants, laboratories and other highly structured environments. Autonomous mobile robots have not yet reached a level of maturity, where they can operate effectively in unconstrained, real-world environments. Many complications exist in developing mobile robots with autonomous operation in such settings; Human-robot interaction (HRI), robot navigation, and robot-environment interaction are all complex research topics with many unsolved difficulties. However, as mobile robot complexity increases, so does the range of applications that they can perform.

Human-robot interaction can be defined as the study of natural and effective communication between robots and humans. Through HRI techniques, a robot can understand the intention of humans in an environment and react accordingly. There are many research areas that compose this multidisciplinary field: people detection, body part localisation, face recognition, facial expression recognition, socially aware navigation, gesture recognition, audio recognition, human-activity detection, human attention detection and group detection, to name but a few. Many of these problem areas have been researched in the context of general computer vision. However, HRI requirements of a moving camera, potentially distant subjects, and computationally efficient algorithms, necessitate specific solutions that are discussed in this thesis.

For seamless interaction between autonomous mobile robots and humans, many problems need to be solved. Most notably, for the introduction of robots in real-world environments, computationally efficient sensing techniques must be developed so that HRI is unimpeded by crowds and dynamic environments. Crowded environments are challenging for many HRI solutions. To simplify the problem, many current techniques make assumptions, such as the number or placement of people within a scene, unoccluded poses, static cameras or requirements of worn devices for gaze estimation and audio recognition. Practical HRI systems do not have this luxury, as real-world demands are entirely variable in nature.

One of the major challenges that the modern world faces is in healthcare. Whilst the quality of healthcare solutions obviously improves over time, the continuing problem of global ageing populations [10] with improved survival rates due to medical advances, will place increasing strain on healthcare providers. This demographic shift will not only place greater demands on a smaller number of eligible healthcare providers, but will actually change the nature of the care that is provided to patients. Most notably, elderly patients and those with chronic diseases will require increasing support from assistive technology and patient monitoring. With the ever increasing demand on improving quality of life, new innovations in technology are required to help future populations receive adequate care.

The introduction of autonomous mobile robots into the healthcare environment could provide a way of alleviating future problems in global health [11]. Capable of performing simple, tedious tasks on a continuous basis, these robots could serve as a valuable tool in freeing up the working time of nurses. Alternatively, robots could provide a convenient way of providing patient monitoring and assistance in houses or nursing homes. The development of HRI solutions, capable of operating in crowded and dynamic environments, is necessary for autonomous mobile robots to operate in these healthcare environments.

To this end, the aim of this thesis is to provide mobile robots with sensing methods, which are capable of operating in crowded and dynamic environments, in order to facilitate HRI. Specifically, the problems of detection and tracking of people and their associated hands, human attention detection through gaze estimation, and group identification are addressed in the following work. To achieve the required robustness, accuracy and computational efficiency, information from multiple input sensors is used. Amongst them, depth cameras are employed, which produce images whose pixel values are not colour, but the distance to the closest object in the relevant direction. RGB-D cameras are those that produce both colour and depth images from the same device. Furthermore, when the Microsoft Kinect is referenced in this thesis, only the physical depth camera is being specified and not any gesture or pose recognition software that is supplied with the camera.

Most traditional HRI systems rely on information from colour cameras. Robot perception in colour images is complicated by issues such as lighting variances, depth ambiguity and differing skin colours. As such, many previous methods simplify the problem, with techniques such as background modelling, specific user initialisation poses or assumptions of unoccluded upright poses. This limits the potential adoption of such methods in an effective HRI framework, particularly one for use in crowded healthcare environments, elderly care, for wheelchair users or those requiring walking aids. However, through a combination of modern sensors, including RGB-D and infrared (IR) cameras, this thesis proposes techniques that are not subject to such constraints.

1.2 Thesis Structure

Chapter 2 reviews the current state-of-the-art in HRI techniques. Particular focus is given to HRI problems that overlap with the methods proposed in this thesis. A history of the significant developments in mobile robotics is first presented. Due to their importance to almost every HRI system, people and group detection methods are then reviewed, followed by the related problem of localising individual body parts. Hand detection is usually a prerequisite for classifying gestures, so recent gesture recognition systems are also presented. Finally, methods for human attention detection and gaze estimation are surveyed.

Chapter 3 presents a hand and body association algorithm designed for crowded and dynamic environments. A hand detection method is initially presented that uses a novel histogram descriptor, with geodesic distances from depth images used to filter background noise in crowded environments. A computationally efficient body detection method is then described, which also employs geodesic distances. Finally, a Bayesian algorithm for associating hands and bodies is presented, for when multiple people and hands are detected in a crowded environment. The method is evaluated in nine controlled environments, and three crowded environments, with results compared to two competing techniques.

Chapter 4 presents an RGB-D sensor-fusion algorithm for tracking hands in crowded and dynamic environments. Firstly the tracked hand state is formulated, together with the process of incorporating depth-image based hand detections and Monte-Carlo updates from colour images. Colour updates use dynamically learned, per-subject skin colour information. A method is then described to combine asynchronous depth and colour updates that have different computational speeds. Hand tracking accuracy is evaluated using nine controlled environments, and three crowded environments. Results are compared against the Kalman filter, when using three different depth-image based hand detectors.

Chapter 5 proposes a sensor-fusion gaze estimation system that uses IR and depth cameras on a mobile robot. The hardware setup is firstly described, including the IR LEDs and camera employed. Using this setup, an IR pupil detection method is presented. Resulting eye positions are used to generate a noisy gaze estimate using depth image information. Finally, a more accurate gaze estimation method is described that uses both IR and depth information. This is confirmed through experiments using a moving robot and multiple moving people, with results compared to the current state-of-the-art.

Chapter 6 presents a group detection algorithm that incorporates real-time gaze estimates to enhance the detection accuracy of static and dynamic interactions. A pairwise group detection process is first defined, with a subsequent recursive algorithm used to fully specify group membership. A novel set of features is extracted, based on sociological concepts of group structure. Group detection accuracy is evaluated using two approaches: a custom model-based method and a logistic regression classifier.

Chapter 7 concludes the work in this thesis, summarising the technical contributions and discussing its limitations. Potential future work is reviewed, with possible applications of the presented methods considered.

1.3 Thesis Contributions

The main technical contributions of this thesis can be summarised into the following aspects:

1. **A hand and body association algorithm designed for crowded and dynamic environments.**

   The proposed method combines a hand detector, robust to noise in crowded environments, with a computationally efficient body detector; a subsequent Bayesian hand and body association algorithm is presented for when multiple people and hands are detected in a crowded environment.

2. **An RGB-D sensor-fusion algorithm for tracking hands in crowded and dynamic environments.**

   The proposed method increases position and velocity accuracy by combining a depth-image based hand detector with Monte-Carlo updates using colour images.

3. **A sensor-fusion gaze estimation system, combining IR and depth cameras on a mobile robot to give better accuracy than traditional visual methods, without the constraints of traditional IR techniques.**

   The proposed calibration-free method uses IR eye detection with depth image analysis, to produce accurate gaze estimates of multiple dynamic people on a moving robot, at a longer range than traditional IR techniques.

4. **A group detection algorithm that uses gaze estimates to enhance the detection accuracy of static and dynamic interactions.**

   A novel set of features, based on sociological concepts of group structure, is used to classify groups based on tracked subjects' position, velocity and gaze estimates.

1.4 Publications

This thesis includes material from the following peer-reviewed publications:

1. S. McKeague, J. Liu, A. Vicente, and G.-Z. Yang, “Human gaze estimation using a sensor-fusion based long range infrared eye detector on a mobile robot,” in *Submission*, Feb. 2015

2. S. McKeague, J. Liu, and G.-Z. Yang, “Hand and body association in crowded environments for human-robot interaction,” in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2013, pp. 2161–2168

3. S. McKeague, J. Liu, and G.-Z. Yang, “An asynchronous rgb-d sensor fusion framework using monte-carlo methods for hand tracking on a mobile robot in crowded environments,” in *Social Robotics*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2013, vol. 8239, pp. 491–500

4. J. Correa, S. McKeague, J. Liu, and G.-Z. Yang, “A study of socially acceptable movement for assistive robots concerning personal and group workspaces,” in *Proccedings of the 7th Hamlyn Symposium on Medical Robotics*, 2014

5. C. Wong, S. McKeague, J. Correa, J. Liu, and G.-Z. Yang, “Enhanced classification of abnormal gait using bsn and depth,” in *Wearable and Implantable Body Sensor Networks (BSN), International Conference on*, May 2012

6. C. Wong, Z. Zhang, S. McKeague, and G.-Z. Yang, “Multi-person vision-based head detector for markerless human motion capture,” in *Wearable and Implantable Body Sensor Networks (BSN), International Conference on*, May 2013

7. J. Liu, J. Correa, S. McKeague, E. Johns, C. Wong, A. Vicente, and G.-Z. Yang, “A healthcare mobile robot with natural human robot interaction,” in *Proccedings of the 5th Hamlyn Symposium on Medical Robotics*, 2012

Chapter 2

Human-Robot Interaction - A Literature Review

2.1 Introduction

In 1920, science fiction author Karel Capek wrote a play, entitled *R. U. R*, which popularised the word “robot”. The term is derived from “Robota”, meaning forced labour in the author’s native Czech. The first electronic autonomous robots were created in Bristol between 1948 and 1949, and were called “Machina Speculatrix” by their inventor, W. G. Walter. These robots, shown in Figure 2.1a, were designed to move towards light sources, using analogue electronics and light sensors.

Between 1966 and 1972, research was conducted in Stanford University on the first digital robot that incorporated planning algorithms. Named “Shakey the robot”, and shown in Figure 2.1b, it was equipped with a camera, range finder and bump sensors. The robot was able to solve tasks requiring navigation, such as, “push the block off the platform”. In accomplishing this, the project introduced multiple techniques, such as the A* search algorithm, which are still used to this day. In the wake of these early successes, public interest in robotics grew rapidly.

One of the first commercially available mobile robots was called “HERO 1”, shown in Fig-

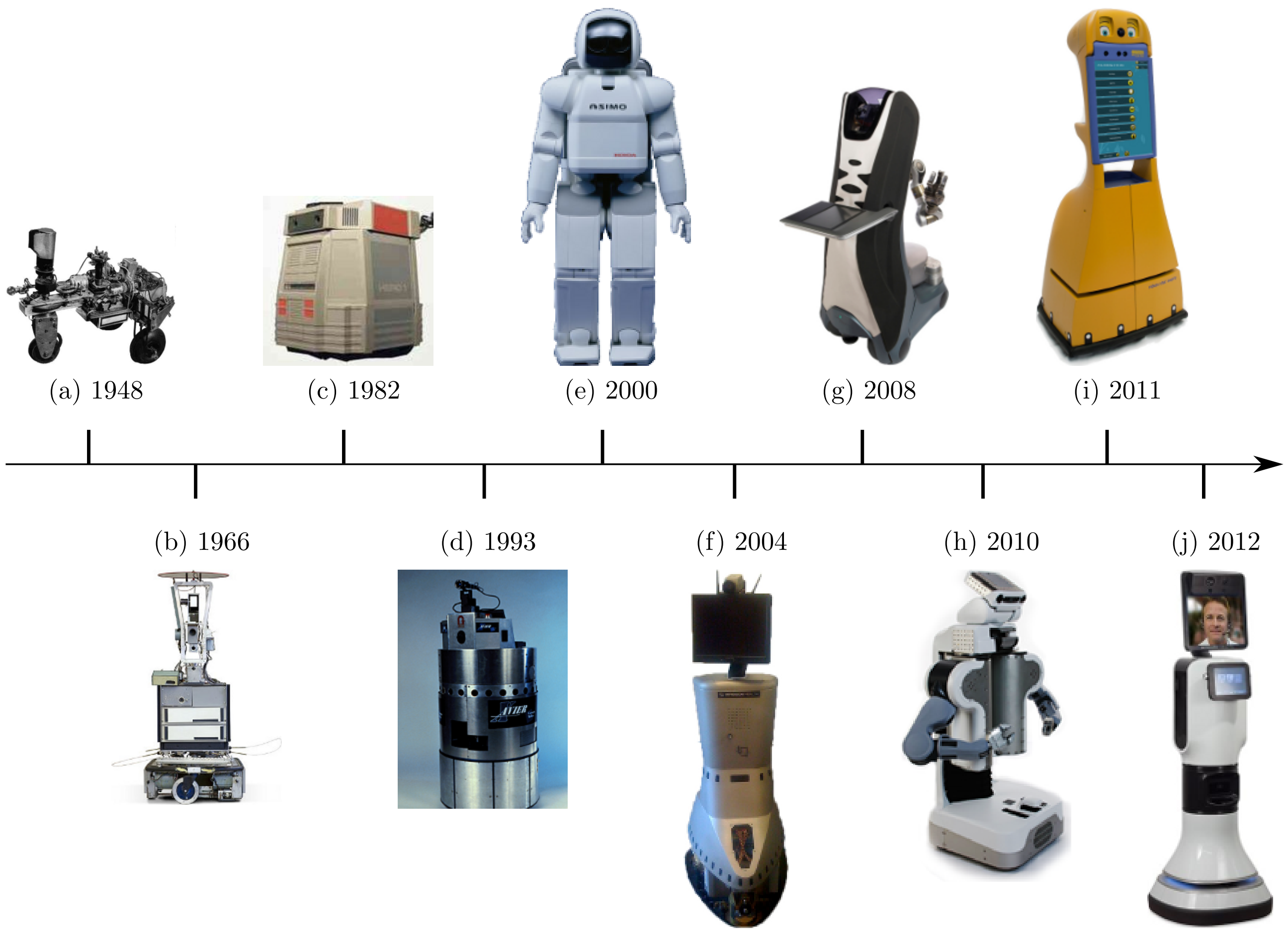


Figure 2.1: Timeline of major developments in the history of mobile robots. Details of the robots and their images sources are as follows: (a) is a "Machina Speculatrix" robot, designed by W. G. Walter in Bristol[a]. (b) is "Shakey the Robot", developed at Stanford University[b]. (c) is "HERO 1", a robot sold by Heathkit[c]. (d) is the "Xavier" robot, developed at Carnegie Mellon University [17]. (e) is the famous Honda "ASIMO" robot[d]. (f) is the "RP-7", made by InTouch Health [11]. (g) is the "Care-O-Bot 3" built by Fraunhofer IPA[e]. (h) is the "PR2" robot from Willow Garage[f]. (i) is the "Scitor G3" robot, manufactured by Metralabs [18]. (j) is the "RP-Vita", the updated model of the "RP-7" from InTouch health[g].

[a]http://www.extremenxt.com/walter.htm - Accessed 21/09/2014

[b]http://www.computerhistory.org/revolution/artificial-intelligence-robotics/13/289/1229?position=0 - Accessed 21/09/2014

[c]http://commons.wikimedia.org/wiki/File:Hero1.jpg - Accessed 21/09/2014

[d]http://world.honda.com/news/2000/c001120_5.html - Accessed 28/09/2014

[e]http://wiki.ros.org/Robots/Care-O-bot?action=AttachFile&do=view&target=Care-o-bot_3.jpg - Accessed 28/09/2014

[f]http://wiki.ros.org/Robots/PR2?action=AttachFile&do=view&target=pr2Image.png - Accessed 21/09/2014

[g]http://www.intouchhealth.com/products-and-services/products/rp-vita-robot/ - Accessed 28/09/2014

ure 2.1c, and sold by Heathkit in 1982. Designed for education purposes, the robot featured light, sonar, motion and sound sensors. Computation was performed using the Motorola 6808 processor, 4 kB of RAM and compact cassettes for data storage. The rapid increase in computation power through the 1980s and 1990s naturally led to increases in robot complexity.

In 1993, the “Xavier” robot was developed at Carnegie Mellon University. Shown in Figure 2.1d, it was the first mobile robot that could be teleoperated via the Web, and contained twin cameras for stereo vision, a range and sonar sensor, and bump sensors. With further developments in mobile robotic capabilities, the possible applications of teleoperation grew. Around 2004, InTouch Health released the “RP-7” remote presence robot, shown in Figure 2.1f. The robot was designed to allow clinicians to remotely see and interact with patients. To do this, the robot features two cameras, a microphone and infrared collision avoidance sensors. A video input jack on the back of the robot allows for the connection of an endoscopic or fluoroscopy camera. This allows a remote physician to directly view the same images as the medical team performing the examination. In 2012, InTouch Health released an updated model to the “RP-7”, called the “RP-Vita”, shown in Figure 2.1j.

In 2000, Honda released the famous “ASIMO” humanoid robot, which stands for “Advanced Step in Innovative Mobility”. “ASIMO”, shown in Figure 2.1e, is designed to be a multi-functional robot assistant, especially for those suffering from reduced mobility. Initially capable of walking at 1.6 kilometres per hour (km/h), newer models have demonstrated speeds of 2.7 km/h.

The “Care-O-Bot 3” is shown in Figure 2.1g and was released by Fraunhofer IPA in 2008. The robot is primarily designed to assist people in their homes. Targeted at supporting the independence of elderly and handicapped people, the “Care-O-Bot” operates autonomously, unlike robots such as the “RP-7”. It can perform fetch and carry tasks using its built-in tray, and can play music and games using its interactive touch screen. To facilitate this, the robot is equipped with stereo cameras, time of flight cameras and two laser range finders.

In 2010 the successful “PR2” robot, shown in Figure 2.1h was released by Willow Garage. The “PR2” serves as the company’s exemplar robot platform for their open source Robot

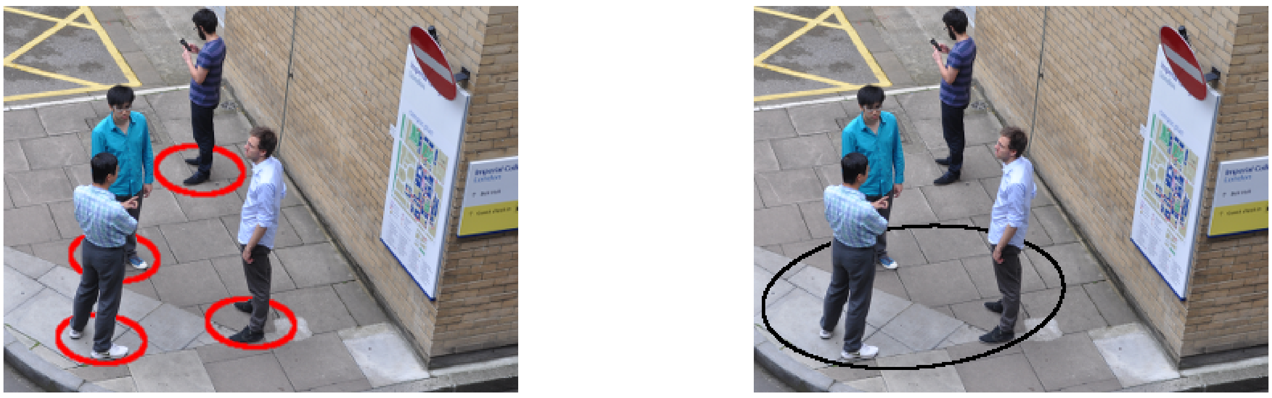
Operating System (ROS). More information on ROS is given in Appendix B, as it is the main software platform used in this thesis. The "PR2" has a large number of actuators and sensors, including multiple stereo cameras, multiple colour cameras, an inertial measurement unit, two accelerometers, two laser range finders, and two hand gripper pressure sensors.

With the continuing problem of global ageing populations [10], elderly care will be an increasingly important motivation for mobile robotic research. The holistic systems required for this application are, however, are still in their infancy. In one of the first long-term user studies of its kind, Schroeter et al. [18], under the "CompanionAble" project, combined the 2011 "Scitor G3" robot from Metralabs, shown in Figure 2.1i, and a smart environment to support elderly dementia patients. The system was evaluated in six user experience trials, where a patient and an informal care giver lived normally for two days in a test home with the system. The robot provided services such as appointment reminders, storing offered items and notification of missed calls. Questionnaires and interviews were used to evaluate user experience, acceptance and societal impact results. Whilst nearly all participants valued the robot for its embodied interaction possibilities, it was noted that neither speech recognition, nor people detection were robust enough.

The research in this thesis aims to extend the state-of-the-art HRI components of such holistic systems. The following section reviews existing literature in a variety of HRI topics, which strongly overlap with the subject matter of the methods proposed in this thesis. Techniques by which people and groups can be detected are described in Section 2.2. Methods for localising body parts, such as hands, are reviewed in Section 2.3. Gesture recognition techniques, for commanding and controlling robots, are discussed in Section 2.4. Finally, gaze estimation methods for human attention detection are described in Section 2.5.

2.2 People and Group Detection

Arguably the single most important precursor to human-robot interaction is the problem of detecting people, as shown in Figure 2.2. Solutions to this problem have applications in many



(a) People Detection (b) Group Detection

Figure 2.2: For any HRI task, one of the most common problems to be solved is that of detecting people, as shown by the hand-drawn red circles in (a). A related problem is that of segmenting detected people into interacting groups, as shown by the hand-drawn black circle in (b).

other areas of robotics, including navigation [19], human intention detection [20] and face recognition [21]. Advances in sensor quality, visual features and machine learning techniques have helped bring about mature solutions to this problem. Until the recent introduction of depth cameras, most approaches used a single monocular camera. The low cost of digital cameras, their ability to work in sunlight, and the intuitive appeal vision provides from a human perspective, have all helped to drive this research.

Human detection is a relatively difficult task from a computer vision perspective. The lack of explicit models in most methods leads to the use of machine learning techniques, where an implicit representation is learned from examples. Changing articulated pose, clothing, illumination and background can all further complicate the problem. Most practical vision systems thus make good use of features rather than pixel values directly, as a way of encoding ad hoc domain knowledge that would otherwise be difficult to learn from the raw training data. Table 2.1 lists some of the most relevant methods to this thesis, with expanded explanations given below.

A recent survey into monocular people detection [22] concluded that from a number of state of the art systems, the two most promising approaches were those that used Haar-like features [23–25] and those that used Histograms of Orientated Gradients [26, 27]. Reviewed methods were evaluated based on their computational efficiency and detection accuracy when tested on

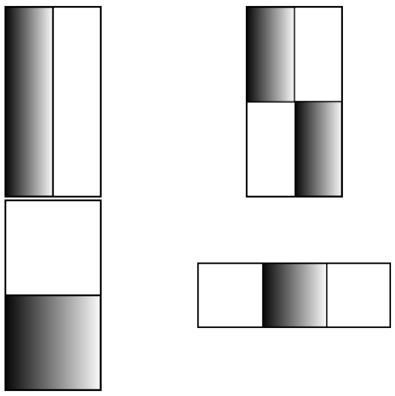
Table 2.1: Select summary of previous people detection methods.

Method	**Camera**	**Feature**	**Classifier**
Haritaoglu et al. [28]	Intensity	Curvature of silhouette vertices	Thresholding
Papageorgiou and Poggio [23]	Colour	Haar-Like	SVM
Viola et al. [24]	Intensity	Temporal Haar-Like	AdaBoost
Viola and Jones [25]	Intensity	Haar-Like	AdaBoost
Porikli [29]	Colour	Histogram	Nearest Neighbour
Dalal and Triggs [26]	Colour	HOG	SVM
Zhu et al. [27]	Colour	HOG	AdaBoost
Tuzel et al. [30]	Colour	Region Covariance	Nearest Neighbour
Tuzel et al. [31]	Colour	Region Covariance	LogitBoost
Chen et al. [32]	Colour	Shape, Colour & Temporal Statistics	AdaBoost
Mosberger and Andreasson [33]	Infrared	Local Descriptors	Random Forest
Choi et al. [34]	Depth	HOD	SVM

a large and varied test set of upright poses, more commonly known as a pedestrian data set.

Haar-like image features are so called because of their similarity to Haar wavelets. The value of a Haar-like feature is specified as the difference between the sum of the pixel values within two rectangular regions. These regions have the same size and shape, are horizontally or vertically adjacent, and are usually computed at a large number of locations and scales. Although very coarse, they are sensitive to the presence of edges, bars and other simple image structures. Within any image, the number of possible permutations of Haar-like features will be far larger than the number of pixels.

The first work to suggest the use of Haar-like features for object detection was by Papageorgiou and Poggio [23]. Contrary to later approaches [24, 25], the generation of Haar-like features was formulated as a two dimensional discrete wavelet transform at multiple scales. A descriptor vector was constructed from the several thousand resulting features for each sample. These vectors were then used to train a support vector machine (SVM) classifier. Object detection was performed by evaluating this SVM using a sliding window at multiple scales over a test image.


(a) Haar-like features

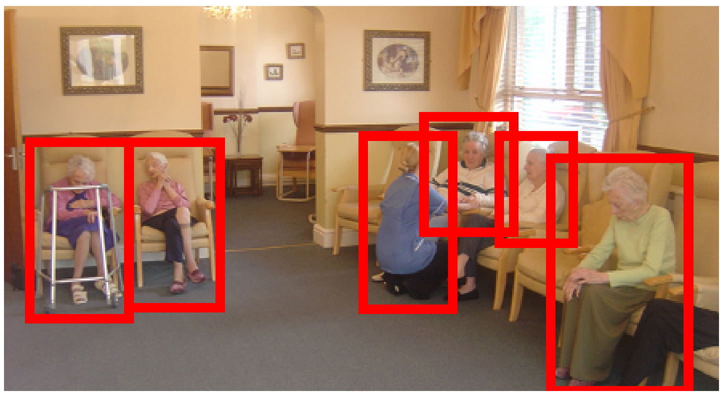

(b) People detection

Figure 2.3: (a) shows illustrations of certain Haar-like features. The value of each feature is equal to the difference of the sum of the pixel values in the black and white regions. (b) shows hand-drawn, illustratory results of a people detector trained using these Haar-like features. Image sourced from [1].

The results highlighted the applicability of this feature in capturing image detail consisting of clear patterns, as shown by the good results for face detection.

To increase the accuracy of human detection in low resolution video, Viola et al. extended the concept of Haar-like features to capture local motion information [24]. Instead of calculating the difference between adjacent regions in the same image, motion information was captured by calculating the difference between the same region in consecutive frames of a video stream. In this way, the Haar-like feature are computed in time, rather than position. Information about the direction of motion was additionally extracted by shifting the previous frame of the video stream in both the horizontal and vertical directions, and re-computing the features. When training an AdaBoost classifier [35] on a pedestrian detection dataset, the use of temporal features provided a greatly reduced false positive rate compared to static Haar-like features.

The AdaBoost algorithm combines a collection of weak classification functions to form a stronger classifier. A weak function is so called because it will not give good classification results by itself. With an input training set, the algorithm calls a simple learning algorithm repeatedly in a number of rounds. A set of weights are calculated over the training set, where

[1] http://www.thehavenresidentialhome.co.uk/images/caring-for-the-elderly.jpg - Accessed 01/10/2014

at each round the weights of incorrectly classified examples are increased, so that the simple learning algorithm focuses more on the harder examples in the training set. Every round, the simple learning algorithm will return the weak classification function that minimises the weighted number of misclassified training examples. Good general features are selected in the initial rounds by the algorithm whilst the hardest to classify training cases are dealt with by the features in later rounds.

The use of Haar-like features was further developed in the seminal Viola-Jones face detection framework [25]. As detailed in a recent survey [36], this framework still provides impressive results for forward orientated face detection. Three main extensions to the original object detection algorithm [23] were employed. Firstly, during initial feature generation, integral images were employed to increase the computational efficiency of the process. Secondly, an AdaBoost classifier, rather than an SVM, was employed to classify resulting descriptor vectors. Finally, the AdaBoost classifier was constructed in a cascading approach, allowing a reduction in the classification processing afforded to samples that are decisively negative.

The integral image is a commonly employed technique to speed up calculations involving summations. For any pixel in a greyscale image, i, its equivalent integral image, ii, value is the sum of the pixels above and to the left of it:

$$ii\left(x,y\right) = \sum_{x' \leq x, y' \leq y} i\left(x',y'\right) \tag{2.1}$$

Using an integral image, any rectangular sum can be computed in four array references, as shown in Figure 2.4. Any of the Haar-like features in Figure 2.3a can thus be computed in at most 9 operations. It is the extreme computational efficiency of the Haar-like feature that makes it a suitable for people and face detection, as they produce a very discriminative image representation when calculated in large numbers. However, many different techniques have made use of the integral image and its variants [37].

One such variant was presented by Porikli [29], who applied the technique to reduce the computational complexity of histogram generation. Porikli applied these “integral histograms” to

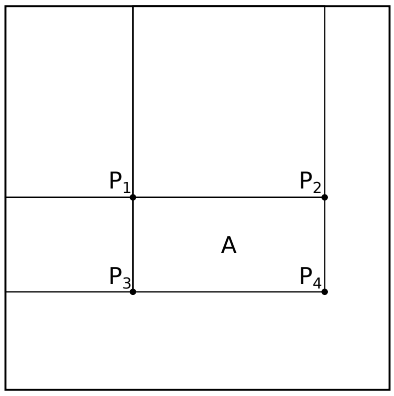


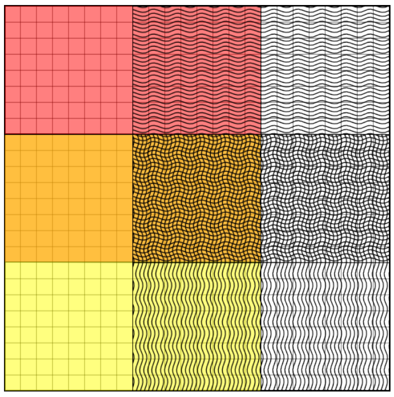
Figure 2.4: Illustration of an integral image. The summation of all pixel values in rectangle A is given by $P_4 + P_1 - (P_2 + P_3)$.

the problem of pedestrian tracking. With a template histogram of the person to track as input, colour histograms were generated for every possible search space in the image. The author claims that this process is over a thousand times quicker than traditional histogram generation. Tracking was shown to be continuous with a fast moving subject. This was contrasted with the mean-shift algorithm [38], which lost tracking under such a condition.

Histograms of Oriented Gradients (HOG) [26] are a widely used feature for human detection. The descriptors bear much resemblance to those used in the Scale-Invariant Feature Transform [39], using locally normalised histograms of gradient orientations. This descriptor is based on the idea that local object appearance and shape can be characterised by the distribution of local intensity gradients, even without precise knowledge of the corresponding gradient.

To generate the HOG descriptor, the image window to be characterised is divided into a number of small spatial regions called cells. For each cell, a histogram of gradient magnitudes and directions is accumulated over the pixels within it. Each cell is then grouped into larger spatial blocks, as shown in Figure 2.5a. A sum of the total histogram values is accumulated for each block and the result is used to normalise all the cells within each block. Presented results [26] emphasised the importance of strong local normalisation; multiple overlapping blocks should cover each cell in the entire image window

Zhu et al. [27] presented two modifications to the original HOG method. First, instead of using a SVM to classify descriptors, the authors used the AdaBoost algorithm, with the same cascade-


(a) HOG descriptor

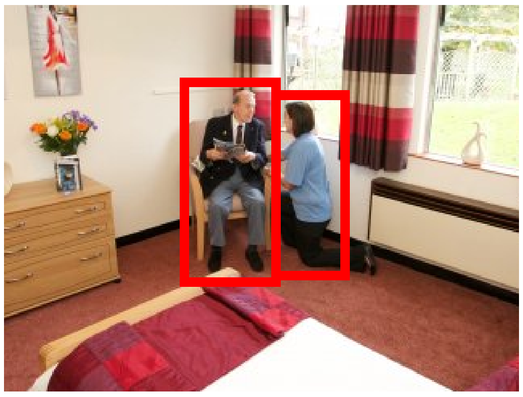

(b) People detection

Figure 2.5: (a) is an illustration of 4 overlapping HOG blocks. The smallest squares represent pixels. These are grouped into 9 cells in total, each containing a default of 8×8 pixels. One HOG block is of default size 2×2 cells, and either has a colour of red or yellow, or one of two patterns. (b) shows hand-drawn, illustratory results of a people detector trained using HOG features. Image sourced from [2].

of-rejectors approach proposed in [25]. At each AdaBoost training level, 250 random blocks were selected. For each block, a separate linear SVM was trained on all positive and negative samples. The weak classifier chosen at each level of the AdaBoost cascade corresponded to the SVM that minimised the weighted number of training examples. The authors additionally used the integral image technique to speed up calculation of the magnitude and orientation of the image pixels. Whilst these modifications resulted in a twenty times speed improvement, detection accuracy of the modified algorithm was slightly decreased.

A recent human detection method, shown to have superior results to the HOG approach, was presented by Tuzel et al. [30]. The authors suggested that regions of an image could be characterised by the covariance of a number of image statistics, such as intensity differentials. The integral image was used to reduce the time to calculate covariance matrices, so that it only depends on the square of the number of image statistics. Pedestrian detection was performed by again matching a template against a multi-scale sliding window over a test image; a detection was defined as the most similar search window to the template. In a later work [31], the authors extended this approach by incorporating covariance descriptors into a machine learning framework. A generalised version of the AdaBoost algorithm was employed for this task, with

[2]http://www.clsgroup.org.uk/?q=sites/default/files/styles/home_slide/adaptive-image/public/homes/images/web1_10.jpg&itok=FnkIFzzo - Accessed 01/10/2014

equations reformulated to operate in non-Euclidean space. Presented human detection results were shown to be superior to the HOG approach.

People detection failure in industrial environments can be disastrous. In these settings, Mosberger and Andreasson [33] used domain specific knowledge to increase detection robustness; knowing that all employees wear reflective safety vests, active IR illumination was used to obtain high intensity reflections. In alternating frames, IR illumination was disabled, giving a normal intensity image. Local STAR features were detected in the IR image and matched to the regular image. A random forest classifier was used to discard features not originating from the reflective vest, and a regressor was used to estimate the distance of reflective vest features from the camera.

As depth cameras become increasingly prevalent, so do the people detection solutions that use them. Choi et al. [34] proposed one such method. The depth image was initially downsampled and segmented into an initial number of regions, based on both distance and normals of neighbouring points. Based on heuristics such as height and width, regions not corresponding to training examples were discarded. Regions fulfilling all other criteria but size could be merged with larger regions. The Histogram of Oriented Depths descriptors [40] were calculated for each resulting region, and were classified as humans using SVMs.

With the development of more sophisticated people detection techniques, more complex problems can be tackled. Chen et al. introduced a crowd identification method that employs object classification techniques [32]. Background subtraction and temporal differences of pixels were used to segment objects within a scene. The resulting objects were classified as crowds using a combination of methods: temporal features classified by the AdaBoost algorithm, the object's self-similarity response, and analysis of its spatio-temporal energies.

Haritaoglu et al. [28] developed a system for detecting and tracking multiple people within a group called “Hydra”. A combination of background subtraction, region-based shape analysis and corner detection was used to segment individuals within a crowd. The resulting heads were then tracked with a dynamic template a second-order motion model. To recover from lost tracking due to camera occlusions, appearance models were constructed for each person.

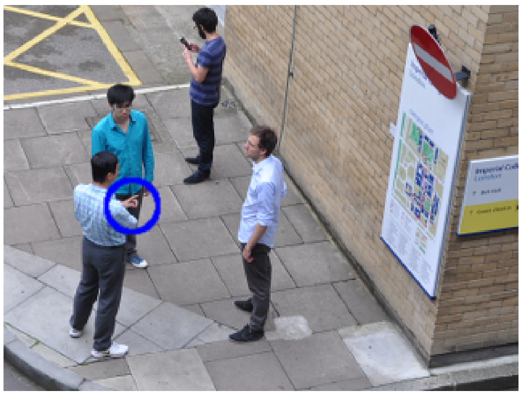


Figure 2.6: Solving the problem of hand detection, as shown by the hand-drawn blue circle, allows a robot to understand human intentions during HRI.

Ong et al. [20] proposed a holistic behaviour system for classifying human's intentions, and reacting to them if necessary. Separate particle filters were used to track people using two different sensors: laser-based leg detection and monocular head detection. With the leg detector's greater false-positive rate and the head detector's lack of depth information, a sensor fusion algorithm was used to combine both sources and compensate for each disadvantage. A behaviour inference method classified this 3D position and velocity estimate into one of eight categories: approaching, hesitating and unconcerned, to name a few.

2.3 Localising Body Parts

Effective HRI solutions need more knowledge than simply the presence or absence of human subjects. As shown in Figure 2.6, individual parts body parts, such as hands, must be detected if a robot is to understand human gestures, for example. Locating body parts has a diverse range of applications, including human activity recognition, human-computer interaction and robot-person following. Because of its importance for HRI, the main focus of this literature review is hand detection and tracking methods. However, there are two main ways of extracting hand information: explicitly detecting the hand and modelling it as a single point, or fitting a prior hand or body model to a subject. Section 2.3.1 will discuss methods that explicitly detect the hand and track its movement. Section 2.3.2 will review methods that detect body parts by fitting prior models to segmented subjects within a scene.

In contrast to people detection, most research in localising body parts has involved the use of more discriminative sensors than a simple monocular camera. This is due to the greatly increased difficulty of the problem. Tracking movement with very high degrees of freedom [41] and frequent self-occlusions [42] has necessitated the use of multiple cameras or other depth imaging techniques. The decreasing cost and increasing precision of 3D scanning devices employing Time-of-Flight (TOF) [43] or structured light techniques further solves the problems of spatial positioning and configuration of multiple cameras [44].

Depth images are invariant to skin colour, clothing, lighting and many other scene parameters. Depth values provide strong cues to distinguish a human subject from other objects. As a result, the region occupied by a human subject is easier to capture in depth, rather than colour images. Indeed, these facets have prompted the adoption of depth images in increasing numbers of computer vision problems. For example, Ruhnke et al. [45] recently applied the Harris corner detector [46] to depth images to find sets of interest points when constructing 3D object models from partial views.

2.3.1 Explicit Hand Detection and Tracking

A summary of the hand detection methods reviewed in this section is listed in Table 2.2; methods that additionally track specific hands over time are listed in Table 2.3. It can be noted that raw hand detection methods almost exclusively use depth cameras. Many methods employing colour cameras make assumptions of hand motion to simplify the problem. These algorithm's therefore explicitly track hands in consecutive frames.

Hand Tracking in Colour Images

As an exception to this statement, Dailey and Bo Bo [47] described a hand detection method using a colour camera. The authors applied Haar-like features to detect hands in crowded scenes, although the results had limited success. In contrast, Valibeik and Yang [56] detected hands with skin colour detection and a motion mask, applied to skin coloured areas. When the

Table 2.2: Select summary of previous hand detection methods.

Method	**Camera**	**Feature Description**	**Classifier**
Dailey and Bo Bo [47]	Intensity	Temporal Haar-like	AdaBoost
Plagemann et al. [1]	Depth	Image patch around keypoint at mesh extremities	"Joint Boost"
Shotton and Sharp [2]	Depth	Per-pixel depth differences	Random Forest
Ikemura and Fujiyoshi [48]	Depth	Depth histogram difference between two regions	AdaBoost
Li and Kulic [49]	Depth	Distance from mesh extremities to nearest edge in radial directions	Nearest Neighbour

Table 2.3: Select summary of previous hand tracking methods.

Method	**Camera**	**Summary**
Yang et al. [50]	Colour	Detect hands by segmenting moving areas and applying a skin colour mask. Track using similarity function
Bretzner et al. [51]	Colour	Detect hands with skin colour. Track with particle filter
Shan et al. [52]	Colour	Detect hands by segmenting moving areas and applying a skin colour mask. Track using particle filter and mean shift
Carbini et al. [53]	Depth & Colour	Detect hands as skin coloured regions within a set distance to a detected face. Track using colour and position probabilities.
Ghobadi et al. [54]	Depth	Segment hands with per-frame clustering using K-Means and EM
Nickel and Stiefelhagen [55]	Depth & Colour	Detect hands as connected skin colour regions. Track the highest probability candidate based on position and colour
Valibeik and Yang [56]	Colour	Detect hands by segmenting moving areas and applying a skin colour mask. Track with a Kalman filter
Chen et al. [57]	Colour	Single user's hands manually initialised. Tracked with particle filter

resulting hands were tracked with a Kalman filter, the results were more accurate.

A similar approach was adopted by Yang et al. [50] for extracting a single user's hand motion. Multiscale image segmentation was performed every frame, to isolate contiguous, homogeneous regions. Region matching between consecutive frames was achieved by minimising an error function that considers similarity between expected positions, area and average intensity. Motion between frames could then be estimated by calculating the affine transformation between matched regions. Neighbouring regions with similar motion were then merged, and a skin colour mask was applied. The largest elliptical region was defined as the head, with the proceeding two being classed as hands.

Particle filtering has been used by many methods to track hands in colour images. Chen et al. [57] and Bretzner et al. [51] described methods to track a user's hand with a single particle filter. Each method employs a different likelihood function: Chen et al. used multiple visual cues, including shape information, motion continuity and colour; Bretzner et al. used skin colour detection and scale-space extrema of blob and ridge features. Chen et al. required a set prior pose to initialise hand tracking. However, Bretzner et al. automatically tracked skin coloured regions of interest.

When tracking hands using particle filters, Shan et al. [52] proposed a solution to tackle the degeneracy problem, where only one sample has a non-negligible weight. Similar to previous methods, a colour probability image was formed by detecting skin colour regions in the image. A difference image of the current and previous frame was also calculated, and thresholded to include only moving regions. Sample weights in the particle filter were based on these two statistics. After samples were propagated with a constant motion model and reweighted, they were shifted to areas of high likelihood in the observation function with the mean shift algorithm [58].

Many of the techniques used by the previous colour image methods, such as continuous motion assumption, are not necessary in depth based methods. For example, Ghobadi et al. [54] proposed an early hand segmentation method using time-of-flight cameras. A per-frame clustering solution was employed, where pixels were initially segmented using K-Means. Using the generated cluster means, Expectation Maximisation was used to refine the resulting Gaussian mixture model. However, several authors proposed sensor fusion methods that combine colour techniques with depth information.

Carbini et al. [53] focused on combining colour techniques and depth information from computation stereo, to track the hands and heads of two people. Hand detection was performed by identifying skin coloured blobs, which were localised using a depth image. Anatomical constraints were used to filter unnatural hand-head combinations. Tracking results were then fused with speech to facilitate human-computer interaction with a large visual display.

Using the same sensors, Nickel and Stiefelhagen [55] proposed a method to recognise pointing gestures during HRI. Similar to Carbini et al., skin colour analysis was used to detect heads and hands, with their 3D positions estimated using computational stereo. A user's head and hand locations were tracked in time by evaluating their movement since the previous frame, anatomic likelihood, and skin colour probability.

Hand Detection in Depth Images

Most depth-based hand detection methods detect a number of features in an image, describe the features using a vector of information called a descriptor, and classify this descriptor using a machine learning classifier. Many feature descriptors have attempt to characterise objects by their shape [59,60]. One of the most successful is the "shape context", by Belongie et al. [61], which the authors used to measure the similarity of two shapes.

In the method, keypoints are generated as edges from the Canny edge detector [62]. The shape context descriptor is a coarse histogram of the relative coordinates of the edge locations in the shape. Histogram bins are of log-polar space, so that histogram entries closer the source edge are weighted more heavily. Point correspondences between two shapes can then be calculated using a cost matrix of the histogram differences between the two shapes' keypoints. The transformation that best aligns the two shapes can then be estimated.

The shape context has been used for a variety of applications. It was recently used by Kaloger-

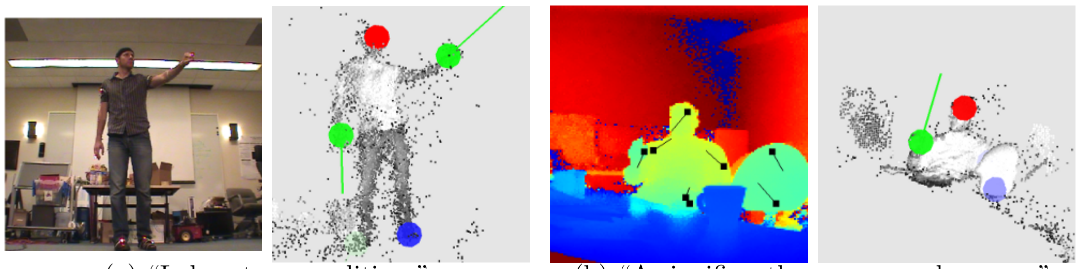

(a) "Laboratory conditions" (b) "A significantly more complex scene"

Figure 2.7: Images from the datasets Plagemann et al. used to evaluate their body part detector [1]. (a) shows the method being evaluated under "laboratory conditions", whilst (b) shows results from "a significantly more complex scene". Images are sourced with permission from [1].

akis et al. [63] to segment and label 3D meshes. In total, eight different shape descriptors were used, in conjunction with a conditional random field (CRF) [64], to classify all faces on a particular mesh. Anguelov et al. [65] described a similar method where, instead of a CRF, a Markov Random Field was used to address the problem of segmenting 3D scan data into objects and object classes.

A modified shape context descriptor was developed for head, hand and foot identification, by Li and Kulic [49]. Keypoints were detected at body endpoints using a hierarchical algorithm. The distance to the nearest edge was then calculated, in radial directions around the keypoint. The furthest distance denotes the keypoint direction. Starting at this direction, the rotation independent "local shape context" descriptor is the concatenation of all radial distances. Descriptors were classified as the nearest neighbour to ideal templates.

Plagemann et al. [1, 66] described a similar method for head, hand and foot detection, with keypoints as body end points. The centre of each mesh in the input point cloud was first found. The geodesic distance of each point in the mesh was then calculated. The "AGEX" (Accumulative Geodesic EXtrema) keypoints are the mesh points with the largest geodesic distances from the centroid, and all other AGEX points. A patch based descriptor was applied to each keypoint, to assign it to one of the three classes. Figure 2.7 shows images of the datasets Plagemann et al. used to evaluate their method.

One of the most successful body part detectors was developed by Shotton et al. [2]. The work

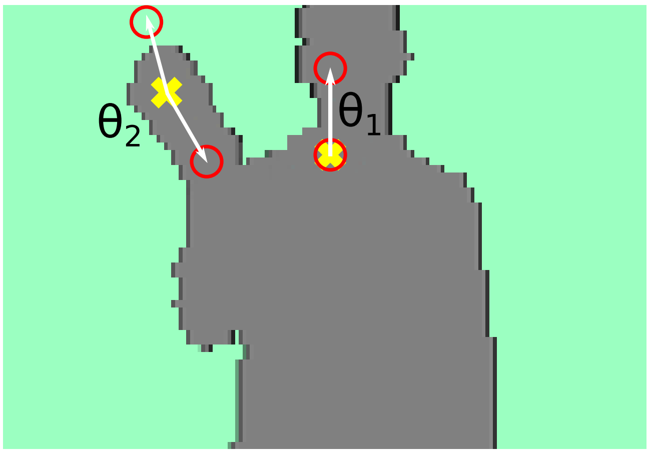


Figure 2.8: Illustration of the depth image descriptor from Shotton et al. [2]. A yellow cross indicates the pixel, $\mathbf{x}$, being classified. From Equation 2.2, the two offsets from each pixel, $\mathbf{x}$, are denoted in the figure as: $\theta = (\mathbf{u}, \mathbf{v})$. The offsets are illustrated with red circles.

details a thirty-one class body-part detector, using a per-pixel classification approach. A simple depth descriptor was used:

$$f(\mathbf{x}, \mathbf{u}, \mathbf{v}) = d_I\left(\mathbf{x} + \frac{\mathbf{u}}{d_I(\mathbf{x})}\right) - d_I\left(\mathbf{x} + \frac{\mathbf{v}}{d_I(\mathbf{x})}\right), \tag{2.2}$$

where $\mathbf{x}$ is a pixel in the depth image d_I, and $\mathbf{u}$ and $\mathbf{v}$ are pixel offsets. Many of these per-pixel depth differences, illustrated in Figure 2.8, are evaluated, with different $\mathbf{u}$ and $\mathbf{v}$ offsets, using a random forest classifier [67]. This algorithm is very fast, but is unsuitable for human-robot interaction as background subtraction must be performed as a pre-processing step.

Many body part localisation methods have used background subtraction (BGS) to simplify the problem and decrease computation. This does, however, make them unsuitable for HRI on mobile robots. As detailed in a recent survey [68], per-pixel GMMs are a widely used method of colour image BGS [69–71]. Störmer et al. [72] and Langmann et al. [73] extended BGS methods to depth cameras by generating GMMs for both the infrared and depth images of a TOF camera.

A similar descriptor to Shotton et al. was proposed by Ikemura and Fujiyoshi [48]. Rather than calculate the depth differences of two pixels, the authors constructed a descriptor that expresses the depth differences over two regions. Normalised histograms of depth values are computed over squares of 8×8 pixels. The descriptor is defined as the Bhattacharyya distance between histograms of two regions, and is termed the "relational depth similarity feature".

2.3.2 Model Fitting

Body Pose Estimation

With the advent of depth cameras, many recent methods have performed hand detection implicitly, finding many body part locations through fitting a full body model. All reviewed methods have assumed an upright pose, which will fail to track sitting users, or those in hospital beds. Additionally many methods required background subtraction, or clutter free environments to work. However, the added context of knowing all joint angles offsets justifies these constraints in some applications. A summary of the review body pose estimation methods is given in Table 2.4.

The shape context, introduced in Section 2.3.1, has also been applied to pose estimation methods. Mori and Malik [74] used it to detect human body configurations in a single colour image. A bipartite graph was formulated between shape contexts calculated at sampled edge points on the input image, and prototype test images. Edge weights represent the cost of matching sample points. Joint angles were calculated using the optimal point correspondence from the prototype with the lowest matching cost.

A sensor-fusion pose estimation algorithm called "VooDoo" was presented by Knoop et al. [75], that uses a time-of-flight camera in conjunction with an RGB camera. A cylindrical human body model was matched to an input body model using the Iterative Closest Point (ICP) algorithm. With a valid pose from a previous frame, points outside a global bounding box were first discarded. A bounding box for each body part was also generated, with outliers discarded. The closest model point to each input point was then found, constrained by sensor data such as 3D projections from image space face detection. Given these correspondences, ICP was used to find the optimal body pose.

Table 2.4: Select summary of previous full body pose estimation methods.

Method	**Camera**	**Summary**
Mori and Malik [74]	Intensity	Optimal training image found by matching shape contexts at edge points. Joint angles found from optimal point correspondence
Knoop et al. [75]	Depth & Colour	Body parts located using bounding boxes of the previous frame's pose. Cylindrical body model matched with ICP
Darby et al. [76]	Motion Capture	Use particle filtering to track body parts in a hierarchy of low dimensional spaces
Raskin et al. [77]	Motion Capture	Use particle filtering to track body parts in a hierarchy of low dimensional spaces
Zhu and Fujimura [78]	Depth	Using a correct pose from the previous frame, find keypoints and use inverse kinematics to calculate new pose
Siddiqui and Medioni [79]	Depth	Generate samples from the previous pose, guided by detected head and hand positions, and evaluate similarity of input and rendered depth images
Moutzouris et al. [80]	Depth	Using a correct pose from the previous frame, project point to low-dimensional space, sample, and evaluate high dimensional projections
Moutzouris et al. [81]	Depth	Extend [80] to find optimal pose estimates for a hierarchy of body parts

Zhu and Fujimura [78] proposed a body pose tracking method using a time-of-flight camera. Background subtraction was first performed. Major body parts were then detected with a deformable template consisting of a head, neck and trunk, and represented by simple shapes. Keypoints on the body were specified using the template. Using corresponding points from the optimal model in the previous frame, a pose estimate was calculated using inverse kinematics.

Siddiqui and Medioni [79] also guide pose estimation using body part detection. Simple depth thresholding was applied to the input depth image to isolate the subject. The head and hands were then located through edge detection and shape analysis. Based on the hand and head locations, samples from the previous optimal pose were generated. For each pose, represented by a sample, a depth image was rendered. Sample likelihood was based on the difference of the rendered and input depth images. A prior probability assigned zero probability to anatomically impossible poses. The optimal pose was given by the sample with the highest posterior probability.

Moutzouris et al. [80] described a single user 3D pose tracking method, using a multi-camera setup in a clutter free environment. Successful pose estimation in a particular frame required a correct pose from the previous frame. This pose was projected to a low dimension space, where multiple samples were taken and backprojected to the original, high dimensional space. Each of these backprojected samples constitutes a possible pose for the current frame. Using a cylindrical human model, the sample which has most volumetric overlap with the input 3D point cloud was chosen as the current most likely pose. Refinement of limb position was then performed by rotating limbs that lie sufficiently outside the 3D point cloud, until maximum overlap is achieved. The authors later extended this technique [81] by finding optimal pose estimates for a hierarchy of body parts, starting with the whole body, and progressing to limb components.

Many techniques that use a latent space to describe body poses, such as those by Moutzouris et al., have the problem that the space describes only those poses used in the training stage. If a different pose is found in the test set, accuracy will be reduced. Two similar solutions to this problem were described by Darby et al. [76] and Raskin et al. [77]. The authors enhanced the

Table 2.5: Select summary of previous hand pose estimation methods.

Method	**Target**	**Camera**	**Summary**
Wang and Popović [3]	Hands	Colour	Single coloured glove detected using GMM probabilities. Per-frame articulated pose estimated with nearest training neighbour
Sigalas et al. [82]	Arms	Colour	Hands detected with background subtraction and skin colour mask. Arm pose estimated with particle filter
Buehler et al. [83]	Arms	Colour	Detect single person with head and torso template. Estimate arm pose with per-pixel colour probability, HOG and anatomical likelihood and a term penalising large temporal changes
Oikonomidis et al. [84]	Hands	Depth & Colour	Track single hand as largest skin coloured region. Articulated pose estimated with optimisation function

generality of represented poses with a hierarchical dimensionality reduction method. In both methods, particle filtering was performed in the low dimensional space. Particle were weighted based on the similarity of the mapped data-space point to input motion capture data.

Hand Pose Estimation

Several accurate hand modelling techniques have been described recently, which calculate the articulated pose of either the hand and fingers, or arms and hands. They usually assume the availability of large hand silhouettes, which limit their use in HRI. However, these markerless approaches are an active area of research, which overlaps with the body pose methods in the previous subsection. A summary of the reviewed methods in this section is thus given in Table 2.5.

Wang and Popović [3] described a method of tracking global position and individual articulation of the fingers, using a monocular camera. The user was required to wear an ordinary cloth glove, shown in Figure 2.9, imprinted with a distinctive colour pattern. A training set of model finger

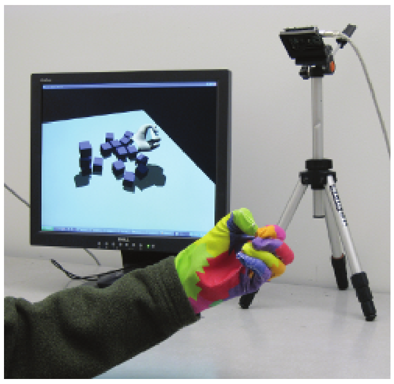


Figure 2.9: Image of the coloured glove used by Wang and Popović [3]. Image sourced with permission from [3].

configurations was firstly created, rendered into small raster images at various 3D orientations. The coloured glove was segmented from an input image using the mean-shift algorithm. The nearest neighbour training image was then found, using a distance metric that penalises the distance to the closest pixel of the same colour in both images.

Sigalas et al. [82] estimated hand and arm pose for gesture recognition. Background subtraction was performed, with a skin-colour mask applied to the resulting area. Each arm was separately tracked with a 4-dimensional particle filter. A multilayer perceptron was used to classify sequences of arm joint angles as one of five gestures. The result was used as input to a radial basis function network, to prevent invalid state transitions.

Buehler et al. [83] developed an alternate hand and arm pose estimation method, for sign language translation embedded into TV programmes. The shape and position of the head and torso was first estimated using a template matching algorithm. A sampling-based framework was used to generate the optimal hand-arm configuration. This was accomplished with a cost function that takes into account the colour probability of different body parts, anatomic likelihood and changes from the previous pose.

Oikonomidis et al. [84] described a method to find the optimal thirty-seven joint model that fits an input hand using the Kinect sensor. A large hand silhouette was extracted from the input image using skin colour detection. A hand model hypothesis was evaluated by initially rendering it as a depth map. This rendered depth map was then compared to the input depth map using a distance measure that penalises unnatural hand poses. An optimisation method was used to minimise this distance measure, using the solution from one frame to generate the initial optimisation estimation for the next frame.

2.4 Gesture Recognition

Gesture recognition can play an important part in an autonomous HRI system, allowing users to issue commands to control the robot. On a mobile robot, gesture recognition methods are necessarily more constrained that in many conventional situations; methods must not assume a static camera or large hand silhouette. Nonetheless, the introduction of depth cameras has led to an increase in HRI compatible methods. There are two main types of gesture recognition in literature: those that differentiate between static hand shapes; and those that analyse hand motion, usually modelling the hand as a point.

A review of vision-based gesture recognition methods [85], noted that there are four defining characteristics of human gestures: hand shape, position, orientation and movement. Hidden Markov Models [86–91] are an effective way of capturing the temporal dynamics of the gesture recognition process. This is corroborated by summary of the reviewed methods in Table 2.6. Listed methods include those that recognise hand gestures only, and those that recognise full body actions.

Dynamic Gesture Recognition

Eickeler et al. [93] presented a dynamic gesture recognition method using intensity images. Difference images of consecutive frames were first generated, with simple background subtraction applied. Moments were calculated from this image and used as descriptors for HMMs trained to recognise thirteen gestures, such as single and two handed waving. Two additional HMMs were trained on arbitrary movements to prevent transitional gestures from being misclassified and facilitate continuous gesture recognition.

Table 2.6: Select summary of previous gesture recognition methods.

Method	**Camera**	**Dynamics**	**Feature**	**Classifier**
Min et al. [92]	Intensity	Dynamic	Hand Position	HMM
Eickeler et al. [93]	Intensity	Dynamic	Image Moments	HMM
Iwai et al. [94]	Colour	Dynamic	Optical Flow Fields	HMM
Chen et al. [95]	Colour	Dynamic	Fourier Descriptor	HMM
Mori et al. [96]	Motion Capture	Dynamic	Body Joint Angles	Hierarchical HMMs
Huang and Trivedi [97]	Depth	Dynamic	3D Shape Context	HMM
Joslin et al. [98]	Colour	Dynamic	Hand Joint Angles	HMM
Holte et al. [99]	Depth	Dynamic	Spherical Harmonics	Nearest Neighbour
Li and Jarvis [100]	Depth	Static	Chamfer Transform	Nearest Neighbour
Gu et al. [101]	Depth	Dynamic	Body Joint Angles	HMM
Gu et al. [102]	Depth & Colour	Dynamic	Body Joint Angles	HMM
Priyal and Bora [103]	Colour	Static	Krawtchouk Moments	Nearest Neighbour
Neto et al. [104]	Glove Sensors	Static	Hand Joint Angles	ANN
Chen et al. [105]	Depth & Colour	Dynamic	Space-Time Subvolumes	ISA
Rui and Anandan [106]	Colour	Dynamic	Optical Flow Fields	Temporal Segmentation

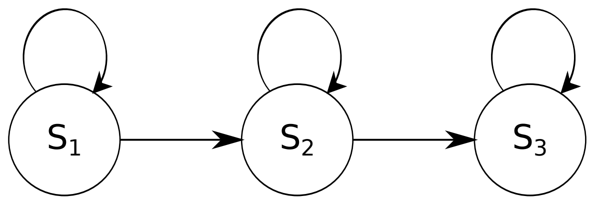
Another intensity image based method was proposed by Min et al. [92], who classified dynamic shape gestures, using discrete, left-right HMMs with four states. Similar to Schmidt et al. [107], the system was trained on twelve shapes such as circles, semi-circles and squares. As well as a self-transition and a transition to the proceeding state, a state could also transition to the state after the proceeding state; this is termed a "skip transition". HMM emissions are defined as the centre coordinate of a single 2D hand position.

Chen et al. [95] presented a dynamic gesture recognition system using colour images. To track a hand, motion information was extracted from a difference image, and was combined with background subtraction and a skin colour mask. The external boundary points of the hand were extracted with a contour following algorithm, and represented as a twenty-two dimensional Fourier descriptor [60]. Using this information, discrete, four state HMMs were trained to recognise twenty, single-handed sign language gestures.

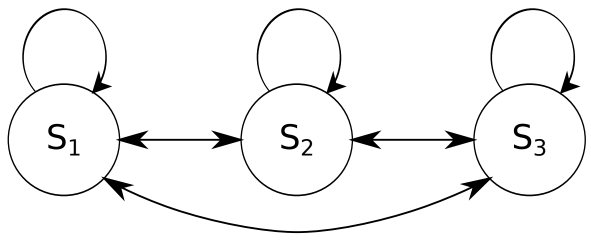
In contrast, Gu et al. [101] presented a dynamic gesture recognition system that uses a skeleton tracker running on the Microsoft Kinect RGB-D camera. A descriptor was used that consisted of four joint angles from the left arm: elbow yaw and roll, shoulder yaw and pitch. Because the authors only considered discrete HMMs [86], these joint angles were initially quantised using K-means clustering. Separate HMMs, with thirty states each, were trained for each of the five gestures considered: come, go, wave, rise up and sit down.

Joslin et al. [98] emphasised finger tracking, as well as hands, to classify dynamic gestures in colour images. To extract this information, markers had to be placed on the user's hand and fingers, allowing the 3D joint angles of the fingers to be calculated. Using twenty-six joint angles as emission variables, HMMs were trained to recognise give gestures, such as a grasping motion and a quotes sign.

Huang and Trivedi [97] proposed a 3D extension to the shape context, in order to classify gestures and actions in depth images. Background subtraction was first applied, with human-shaped blobs being tracked with a Kalman filter. Tracked humans were segmented into voxels - the 3D equivalent of 2D pixels. A single 3D shape context was centred on each tracked subject, with 3D bins that covered all possible human poses. The normalised number of voxels within



(a) Left-right HMM



(b) Fully-Connected HMM

Figure 2.10: Illustrations of two different types of HMMs. (a) depicts a left-right HMM, whilst (b) shows a fully-connected HMM. S_1, S_2 and S_3 represent the states of the HMM, and arrows indicate a non-zero transition probability between states.

each bin was used as a fifty-two dimension descriptor vector.

Using this descriptor, nine actions such as standing up and sitting down could be classified by discrete left-right HMMs, illustrated in Figure 2.10a. In contrast, most HMM implementations allow a transition from one state to any other, a type termed “fully-connected” and shown in Figure 2.10b.

The 3D shape context was used in another dynamic gesture recognition method, by Holte et al. [99]. Areas of motion in the depth image were first extracted using image differencing. The same 3D shape context as used by Huang and Trivedi [97] was then centred over the segmented body. A descriptor was then generated from the 3D shape context, using spherical harmonics. The system was evaluated on four, one and two-armed gestures, including clapping and waving. For each gesture class, the mean descriptor vector of the training examples was calculated. An input gesture was classified by finding the mean training vector with the lowest distance.

Static Gesture Recognition

As opposed to the previously reviewed methods, which classified a gesture based on hand movement, Li and Jarvis [100] presented a method of recognising static hand gestures in depth images. The simple assumption that the hand is the closest object in the scene was used, along with depth thresholding, to isolate it. A set of eleven single-handed shapes, including finger pointing, was used for evaluation, with a training image provided for each. For each training image, a distance value for an input hand was calculated using Chamfer matching. The output gesture was defined as the nearest training neighbour.

Priyal and Bora [103] recently described a colour image based system for static hand gesture recognition. Hands were detected by isolating explicitly defined skin colour regions [108]. This region was separated into arm, palm and finger components using geometric properties of width and area continuity. The hand was orientated with respect to the detected fingers, and normalised to a set size. Krawtchouk moments were extracted from the resulting hand, as well as all training samples. A gesture was then classified as the training sample with the Krawtchouk moments that are most similar to those of the test sample. Ten single-handed gestures were evaluated, including a fist and a number of extended fingers.

If static hand gestures are issued consecutively, transitional hand shapes can be incorrectly classified as unintended gestures. Neto et al. proposed a solution to this problem [104] using a combination of two artificial neural networks (ANNs). Joint angles from glove sensors were fed into an ANN, which classifies the input as one of the system's trained gestures. A separate ANN was used to classify the glove sensors as one of a number of transitional gestures to be filtered. Only if no transitional gesture is detected by the second ANN, did the system allow a valid gesture from the first ANN to be output. The system was evaluated on ten single-handed gestures, including an open palm and pointing with the index finger.

The same problem of incorrectly classified transitional gestures was tackled by Iwai et al. [94], for dynamic gestures. Optical flow was performed on each colour input image. For all flow fields of a gesture example, the principal components were extracted with PCA. The most significant flow fields for all training samples were quantised with a clustering method. The observation symbols for the discrete HMMs used, were the nearest cluster for each test sample's principal components. The system was evaluated on seven instrument playing actions and seven Japanese sign language gestures, each set containing one and two handed gestures.

Action Recognition

Methods that classify whole body actions, such as walking or sitting down, share many similarities with hand gesture recognition techniques. For example, Chen et al. [105] tackled the problem of feature selection in action recognition using RGB-D cameras. This was accomplished with an unsupervised learning algorithm called Independent Subspace Analysis (ISA). It was trained on space-time subvolumes, which were randomly extracted from both depth and RGB data, and output the detected activity class. A second ISA was trained on larger subvolumes extracted from the RGB-D data. The output classes of these two separate classifiers were combined using an SVM variant. Evaluation was performed on two datasets; the first had six actions that include lifting pushing and waving, the second had ten actions that include carrying and throwing.

For recognition of action abstractions, Mori et al. [96] proposed a hierarchical HMM structure. A motion capture system was used to extract a thirty-six dimension vector of joint angles. This was used to recognise thirty specific actions, such as “lying on the right side” or “sitting in the Seiza position”. However, a small initial set of continuous HMMs were trained on more abstract actions, such as “lying” or “sitting”. If an input action was classified as the abstract class “lying”, it would be evaluated on a less abstract HMM level of actions, such as “lying on back” or “lying on side”. Only if the input action was then classified as “lying on side” would it then be evaluated on the most specific HMM level of actions.

A similar hierarchical human action recognition method was proposed by Gu et al. [102]. The method used the skeleton tracking from the Kinect sensor, and was evaluated in a breakfast scenario. In the lowest hierarchy level, discrete HMMs with ten states were used to classify six actions based on eight joint angles from the right arm. The resulting action and object were used as input to a mid-level Bayes classifier, which output a better estimate of the action, taking into account object context. In this way an action such as drink, would have less probability of being correct if it was observed with an object such as cereal. Finally, this action was used as input to high-level HMMs, which output a refined estimate of the action based on temporal context. In this way an action such as stir was less likely to be correct if a previous pour action

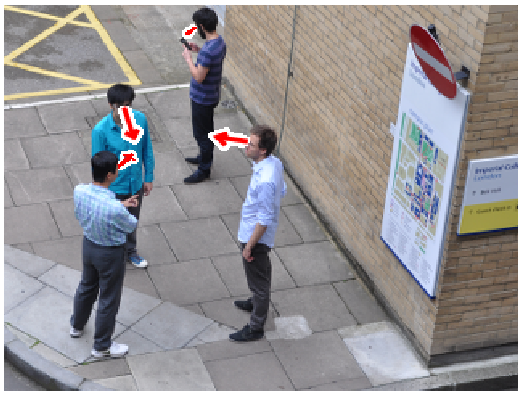


Figure 2.11: Gaze estimation techniques allow a robot to know where a detected person is looking, as illustrated by the hand-drawn red arrows on white background. This can be used in solving many other HRI problems, such as that of group detection or socially aware navigation.

was detected.

A solution to the similar problem of segmenting continuous human actions was presented by Rui and Anandan [106]. Background information was first subtracted, followed by dense optical flow calculation for all frames in the video to segment. Singular value decomposition (SVD) was performed on the resulting collection of flow fields, and the least significant component directions were discarded to reduce noise. Finally, the boundaries between actions in the clip were denoted by discontinuities in the temporal trajectories of the remaining components.

2.5 Gaze Estimation and Human Attention Detection

Gaze estimation techniques allow a robot to know where a detected person is looking, as shown in Figure 2.11. This has many applications in HRI: awareness of human-human interaction, context for deictic gestures and filtering audio commands not directed at a robot. Gaze estimation techniques can broadly be divided into methods that require infrared illumination and those that do not. Whilst infrared methods can allow for very accurate gaze estimates, working range restrictions have prevented the applicability of these methods to HRI. This problem is tackled in Chapter 5. A brief description of some of the reviewed gaze estimation methods in this section is given in Table 2.7, with corresponding advantages and disadvantages listed in Table 2.8.

Table 2.7: Brief description of some previous gaze estimation methods.

Method	**Summary**
Haro et al. [109]	Illuminate pupil with difference of on & off-axis LED images. Threshold result and track with Kalman filters.
Morimoto et al. [110]	Use two light sources at known positions to generate 3D glint positions and the corresponding gaze vector
Ji and Zhu [111]	Illuminate pupil with difference of on & off-axis LED images. Detect pupil and glint. Estimate gaze using ANN with pupil-glint positions
Yoo and Chung [112]	Attach LEDs to monitor corners to generate four glints. Illuminate pupil with image difference of monitor LEDs and an extra on-axis LED. Calibration is used to map glint image-coordinates to monitor coordinates
Nickel and Stiefelhagen [55]	Use [26] to track faces. Extract fixed-size intensity and disparity image. Classify orientation with ANNs
Benfold and Reid [113]	Using k-means clustering, classify pixels in fixed-size face image as hair, skin or background. Classify orientation with random ferns and a HMM
Czyzewski et al. [114]	Illuminate pupil with difference of on & off-axis LED images. Brightest point is glint. Estimate gaze based on glint position relative to pupil centre.
Cho et al. [115]	Locate monitor with template matching. Detect pupil with image-based method. Calibration is used to map pupil image-coordinates to monitor coordinates
Benfold and Reid [7]	Use [26] to track heads. Extract HOG descriptor and colour features. Classify with random ferns
Lee et al. [116]	Find pupil with image-based method. Generate glints with an infrared light. Use MLP for eye depth, used to calculate gaze
Sheikhi and Odobez [117]	Motion capture provides subject position and head angles. Continuous HMM created, where current state represents target observed
Cho and Kim [118]	User's face is tracked in 3D using computational stereo. Narrow-view camera on pan-tilt unit focuses in on tracked eye. Pupil detected with image-based method, with two glints from IR illumination. Calibration is used to map pupil-glint image coordinates to coordinates on a separate screen

Table 2.8: Advantages and disadvantages of previous gaze estimation methods. Calibration refers to instance or user-specific calibration, and not to one-time calibration, such as generating internal camera parameters.

Method	**Camera**	**Range**	**Calibration Required?**	**Multi-User?**
Haro et al. [109]	Infrared	Unspecified	No	Yes
Morimoto et al. [110]	Infrared	30 – 80 cm	No	No
Ji and Zhu [111]	Infrared	1.0 – 1.5 m	No	No
Yoo and Chung [112]	Infrared	40 – 60 cm	Yes	No
Nickel and Stiefelhagen [55]	Depth & Colour	Unspecified	No	Unspecified
Benfold and Reid [113]	Colour	Long	No	Yes
Czyzewski et al. [114]	Infrared	60 cm	No	No
Cho et al. [115]	Head-Mounted Camera	Unspecified	Yes	No
Benfold and Reid [7]	Colour	Long	No	Yes
Lee et al. [116]	Head-Mounted Camera	3 cm	Yes	No
Sheikhi and Odobez [117]	Motion Capture	Unspecified	No	Yes
Cho and Kim [118]	Infrared	1.4 – 3.0 m	Yes	No

For the problem of a robot estimating where a user is pointing, Nickel and Stiefelhagen [55] documented the increased accuracy when considering head orientation. With a similar method, Axenbeck et al. [119] used head orientation to enable a mobile robot to estimate the closest object that a user is looking towards. Both approaches first detect potential subjects using the Viola-Jones face detection algorithm [25] on input intensity images. Similarly, both approaches track a subject's head as a fixed-size bounding box. To estimate head orientation, Nickel and Stiefelhagen used a computational stereo system to extract pixel values from both intensity images and disparity images. Conversely, Axenbeck et al. extracted image features using only a monocular camera. In both cases, an artificial neural network was used to classify these input features, giving the resulting head orientation.

Mora and Odobez [120] recently used the Microsoft Kinect to estimate gaze. A subject's face was initially found using monocular face detection. This area was isolated within the depth image, and used to estimate head pose. Using a 3D mesh model, the eyes were isolated and their gazing direction estimated using an appearance-based model. Both the 3D mesh model and the eye appearance model required user-specific training data, which reduces the method's applicability to HRI. A somewhat similar method, combining a 3D head model and appearance-based gaze estimation, was proposed by Choi et al. [121]. However, the method by Mora and Odobez gives more competitive gaze estimates during unrestricted head motion.

The Microsoft Kinect was also used by Kim et al. [122] to estimate head pose. The algorithm works with a single user only, and requires a frontal RGB-D image of the face as training data. From this image, many synthetic head poses were generated, with corresponding RGB images rendered. These synthetic training images were then clustered, with each image projected into the locally optimum PCA subspace for that cluster. For a given input image, its optimal subspace was found by minimising reconstruction error. The input head pose was then given by the pose of the most similar training image within the optimal cluster. This method was only tested on synthetic data and so it is unknown how the system functions in real-world scenarios.

Synthetic head poses were also generated for training data in the head pose estimation method by Fanelli et al. [123]. Due to their suitability for large training datasets, random regression forests were used to detect input head poses. During training, a set of patches, with two rectangular areas within them, were randomly sampled from the training examples. The difference between the average depth values of these rectangular areas was used as the test feature for each node in a tree. A tree's leaf node stores a multivariate Gaussian distribution, which specifies the nose position and head rotation for an evaluated input image.

The head pose estimation method by Padeleris et al. [124] required only a single RGB-D image of the face as training data. To calculate head pose, a 3D model of the head was made from an input depth image, and transformed according to a process called particle swarm optimisation. Each transformed model was rendered as a depth image, and its values were compared to the training image. The transformation that gave the most similar rendered depth image to the training image was specified as the relative head pose. The method requires a graphics processing unit, and only works at 10 frames-per-second, which limits its use in HRI.

Much previous work on visual gaze estimation has been focused on its application to surveillance systems. In their most recent work [7], Benfold and Reid described a system to detect and track subjects' heads in order to estimate their gaze direction. The HOG descriptor [26] was used for head detection, whilst tracking was performed using a modified Kalman filter. Head poses were grouped into forty five degree classes and classified using randomised ferns. A combination of HOG and Colour Triplet Comparisons was found to be the most effective feature for head pose classification. The authors' previous paper [113] described a similar method for head pose classification, using lower resolution images. Random ferns were again used for pose classification, but a hidden Markov model was used to ensure temporal pose consistency.

Sheikhi and Odobez [117] used a humanoid Nao robot to estimate the subject of a person's attention. The method was motivated by the desire for humanoid robots to know who is speaking to whom in a particular scenario. Using a provided head pose, this "visual focus of attention" was evaluated using a continuous HMM, with a state per target, and the Gaussian mean set to the target head positions. The authors state that this obtains more temporally consistent results than previous solutions that use GMMs [125].

The use of infrared reflection to locate eye landmarks is a mature method of gaze tracking, with several different approaches. A common approach is to use an array of on-axis LEDs to reflect light from the ocular fundus through the pupil, and in alternate frames use off-axis LEDs to illuminate only the surrounding area. This reflected light from the ocular fundus is commonly known as the red-eye effect [126], and can be used to find centre of the pupil. Haro et al. [109] use this principal to track pupil centres in a monocular image. The authors did not filter visible light and so the difference of the on and off-axis images had significant noise. As a result, potential eyes were filtered using an adaptive threshold and appearance based modelling.

If a subject remains close to the camera, infrared lighting can be used to view corneal and lens reflections [127] in sufficiently large images of the eye. Usually, up to four main reflections, or Purkinje images, are potentially visible. A common method of gaze tracking involves using the pupil centre along with the first Purkinje image [110, 111, 114], commonly referred to as a glint. Several authors have used the relative positions of these two landmarks to coarsely track gaze on a graphic display [111, 114].

To produce the glint, Czyzewski et al. [114] used two vertical off-axis arrays on either horizontal side of the camera, whilst Ji and Zhu [111] used a single circular LED array, with a large radius, centred on the camera axis. Due to the system's lack of calibration, the method by Czyzewski et al. required a static head. However, methods employing calibration usually suffer from being person and orientation dependent [111]. Ji and Zhu thus used generalised regression neural networks to allow head movement without experiment-specific calibration. Morimoto et al. [110] proposed a different calibration-free technique to estimate gaze under head motion, using the assumption of two light sources with constant known positions.

The use of two or more Purkinje images can aid in the gaze estimation process, but more hardware is usually required to generate the extra reflections. Lee et al. [116] used a head mounted camera with a single infrared light to estimate gaze using the first and fourth Purkinje images. As this light was off-axis, pupil detection was performed using circular edge detection. Eye depth was determined using a multi-layered perceptron and, after user-specific calibration, a user's gaze in 3D space could be calculated using a geometric transform.

A slightly more cumbersome two-camera wearable device was proposed by Cho et al. [115] to determine gaze location on a graphic display. One camera used a template matching method in order to locate the monitor. The other used circular edge detection with local thresholding to detect the pupil centre. User-dependent calibration allowed a geometric transformation to be defined, from the pupil position in image coordinates to monitor coordinates. To account for slippage of the wearable device after calibration, the authors described a method to alter the geometric transformation based on tracking corner points of the eye.

Using the pupil location and four reflection points on the eye, Yoo and Chung [112] also proposed a method of tracking gaze on a graphic display. An LED was attached to each monitor corner to produce the four eye reflections. Similarly to Cho et al., two cameras were used to allow for head movement, but they are statically positioned beside the monitor. One camera used skin detection to track a user's face. The other, on a pan-tilt base, analysed the user's eyes, and had an additional on-axis LED to detect the pupil centre. Using a simple calibration procedure, and the assumption that the four eye reflections are coplanar reflections of the monitor LEDs, a mapping function was found to transform the camera image plane to the monitor screen.

It has been noted that all methods of Purkinje image analysis have the same drawbacks of requiring a high resolution eye image [110, 112, 114, 116, 118], and frequently require restrictive calibration [112, 116, 118]. Cho and Kim [118] recently attempted to overcome these limitations with their long range gaze tracker. To detect infrared reflections at natural HRI distances, a narrow view camera on a pan and tilt unit was used. Two stereo cameras were used to track a subject's movement within a scene, and control the pan and tilt unit. Calibration was still required however, and only one user could be tracked at a time.

2.6 Conclusions

As stated in Chapter 1, human-robot interaction is a multidisciplinary field, composed of many research areas: people detection, body part localisation, face recognition, facial expression recognition, socially aware navigation, gesture recognition, audio recognition, human-activity detection, human attention detection and group detection, to name but a few. This chapter has reviewed some of the aspects most relevant to the methods proposed in this thesis.

A review of the significant developments in the history of autonomous mobile robots was first given in Section 2.1. Methods of people and group detection were then discussed in Section 2.2. The localising of body parts was detailed in Section 2.3. Specific attention was given to methods of detecting and tracking hands as a single point, as well as articulated pose estimation methods for both hands and full bodies. Section 2.4 described Methods by which gestures can be recognised from the movement of tracked body parts. Finally, gaze estimation techniques, using both vision and infrared imaging, for human-attention detection, were reviewed in Section 2.5.

Reviewed techniques used a multitude of sensors: intensity and colour images, depth and infrared cameras, motion capture systems and wearable devices. Many methods of localising important features in images were presented, along with corresponding descriptors. Method restrictions, such as requiring calibration or background subtraction, were stated where applicable. Machine learning techniques for classifying these descriptors were discussed, along with their application to reviewed methods.

This chapter has reviewed the existing literature for the problems addressed in this thesis. These problems are illustrated in the following example: an elderly person, engaged in conversation with friends, wishes to attract a robot's attention. This composite task consists of many problems. The robot must detect and track the subject in a crowded environment. To engage with the user, it must track their hand movement. Knowledge of the subject's gaze would ensure that the robot doesn't react to the wrong person. Finally, understanding the subject's group participation would enable the robot to respect existing human-human interaction.

Currently, most successful people and group tracking solutions are very computationally expensive, restricting their use on mobile robot hardware. Crowded environments pose problems for most explicit hand detection techniques. On the other hand, model-based pose estimation methods assume an upright pose, prohibiting sitting users, those in wheelchairs and subjects with consistent lower body occlusions. Traditional visual gaze estimation methods do not have the required granularity for many HRI tasks. However, existing infrared methods mostly have too short a range to be used on a mobile robot. In the following chapters, novel solutions to these HRI problems are described that either relax the constraints of existing methods, improve their accuracy, or decrease their computational expense.

Whilst contemporary mobile robots vary in size depending on their target applications, those designed for HRI commonly have a height somewhat under that of the average human. In HRI, this limits the number of people in the robot's view. This thesis defines the issue of crowds in indoor human environments as commonly occurring clusters of around five people. Additionally, the following methods assume that interacting subjects are either standing or sitting in an indoor environment. Whilst some methods may operate accurately outside these assumptions, this is the foreseen to be the most common scenario where the presented methods could be employed. Unusual test cases, such as users with hair completely occluding their faces, are not considered.

Chapter 3

Hand and Body Association in Crowded Environments for Human-Robot Interaction

3.1 Introduction

Human-robot interaction (HRI) solutions provide an important means by which people can naturally command and control robots in an environment. As discussed in Chapter 1, HRI is increasingly important in areas such as healthcare, where robots have the potential to assist the elderly and vulnerable. Many useful HRI tasks, including attention detection and gesture recognition, require an accurate understanding of the motion of people, and their associated hands, in an environment. However, many existing solutions for hand and body association are not robust enough for use in real-world environments. The results of the 2012 ChaLearn Gesture Challenge [128] illustrate this point: none of the best ranked methods explicitly tracked

The work in this chapter was partially presented in S. McKeague, J. Liu, and G.-Z. Yang, "Hand and body association in crowded environments for human-robot interaction," in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2013, pp. 2161–2168 either people or hands. There is a clear need for robust methods to tackle this problem. The contribution of this chapter is thus to provide a hand and body association algorithm, designed for crowded and dynamic environments.

Until recently, most body detection methods used colour cameras, as shown in Section 2.2. Whilst these methods can provide good detection accuracy, they are usually computationally expensive, due to the abundance of locations and scales in the image that could contain a body. Detecting individual body parts, such as hands, is an arguably harder task. For many HRI tasks, both problems need to be tackled simultaneously, and in real time. In this case, the computational expense of traditional colour-based body detection methods warrants alternative solutions to the problem.

Many existing methods of hand and body tracking do not perform well in crowded and dynamic environments [1]. Frequently, body part detectors simplify the task with assumptions of body pose and occlusions [1, 49, 74]. However, these constraints cannot be assumed in real-world environments. Much contemporary research has focused on the problem of full-body pose estimation. As discussed in Section 2.3.2, many well-known methods require a static background [2] or require a user to adopt a specific pose during tracking initialisation [129, 130]. Methods with such restrictions are unable to be incorporated into an effective HRI framework, particularly one used in healthcare applications.

The hand-body association framework described in this chapter has three major components, each representing a novelty of the proposed method. A method of detecting hands in crowded environments is first introduced. Using depth camera information, geodesic distances are employed to isolate points local to hand, regardless of their Euclidean distance to points in other regions. Secondly, a computationally efficient, probabilistic method of identifying body clusters in a cluttered scene is described. This facilitates the subsequent association of tracked bodies and hands, based on a Bayesian framework, with increased robustness.

For HRI purposes, the proposed method can handle rapid tracking initialisation, varying background illumination, differing skin tones and clothing, and multiple hypothesis considerations. It uses depth images to increase robustness in crowded environments and operates in real-time.

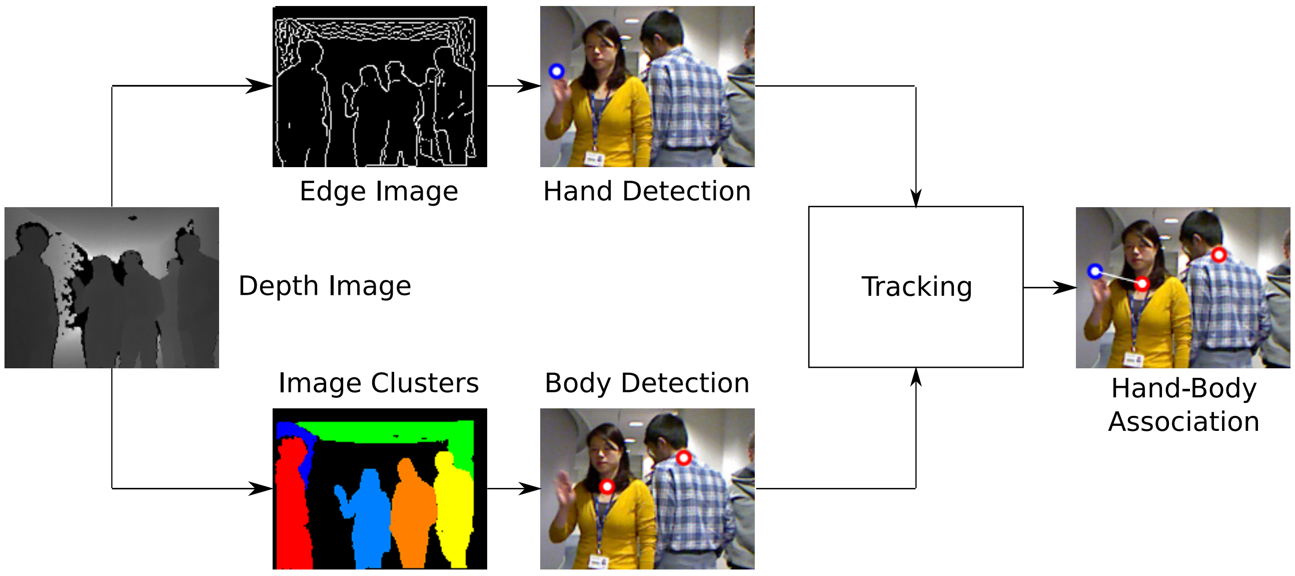


Figure 3.1: Illustration of the hand-body association system structure.

Accuracy is evaluated using a range of parameters, and compared to two existing methods: the shape context descriptor [61], and the body part detection method proposed by Plagemann et al [1]. No existing gesture-based dataset could be found, where environments were crowded enough to thoroughly test the proposed method. Suitable datasets were thus created, manually annotated, and made publicly available. Results for these crowded datasets are presented for all three methods, with improved performance exhibited by the proposed method.

3.2 Method

The three major hand-body association components will be described in this section. A hand detection method for crowded environments is proposed in Section 3.2.1. An associated body detection method is described in Section 3.2.2. Finally, the algorithm for probabilistically associating tracked hands and bodies is described in Section 3.2.3. An overview of how the separate components are combined in the complete hand-body association framework is shown in Figure 3.1.

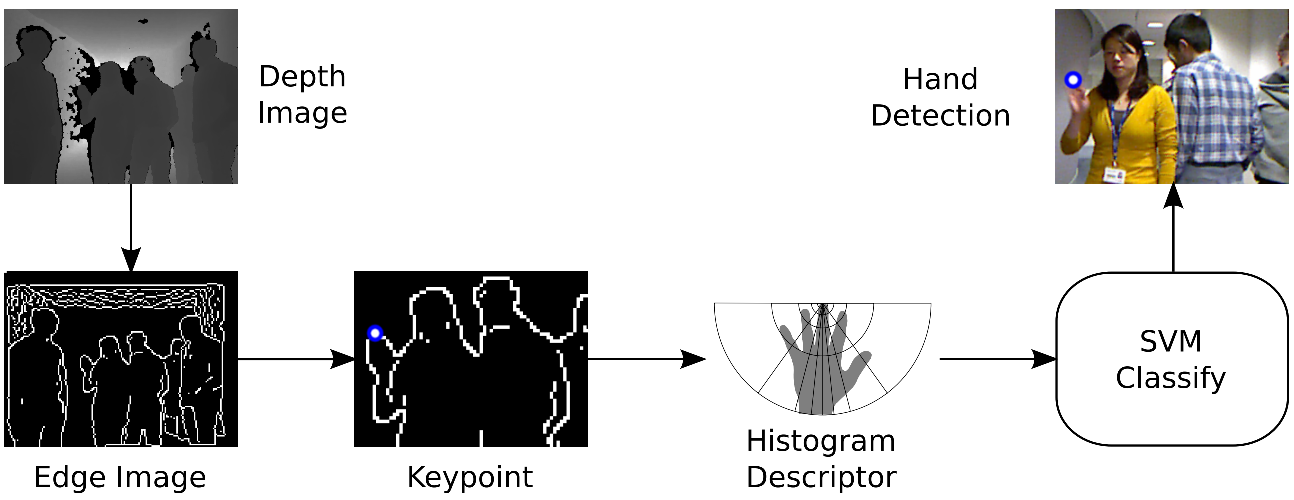


Figure 3.2: Illustration of the proposed method for hand detection in crowded environments.

3.2.1 Hand Detection

An overview of the proposed method for hand detection in crowded environments in shown in Figure 3.2. Edge detection is performed on an input depth image, with a novel descriptor generated for every edge point. A Support Vector Machine (SVM) is then used to identify descriptors belonging to hands.

As noted in Section 2.3.1, the shape context is a successful image descriptor [61, 63]. The proposed hand detection descriptor, shown in Figure 3.3, is similarly a histogram, though it is optimised for crowded environments using depth images. To reduce the hand detection search space, Canny edge detection is initially performed in the depth image. The resulting edges are defined as hand keypoints, to be characterised by the proposed descriptor.

The shape context uses log-polar sampling to ensure that the highest concentration of sample points lies near the keypoint being analysed [131]. The proposed descriptor extends log-polar sampling to angular binning, as well as distance, to focus on local regions most indicative of a hand. As shown in Figure 3.3, this is achieved with a keypoint direction that points towards the centre of the hand. The hand centre is defined to be the mean position of the depth pixels that comprise it. Additionally, areas in the opposite direction to the hand can be ignored, due to their lack of information.

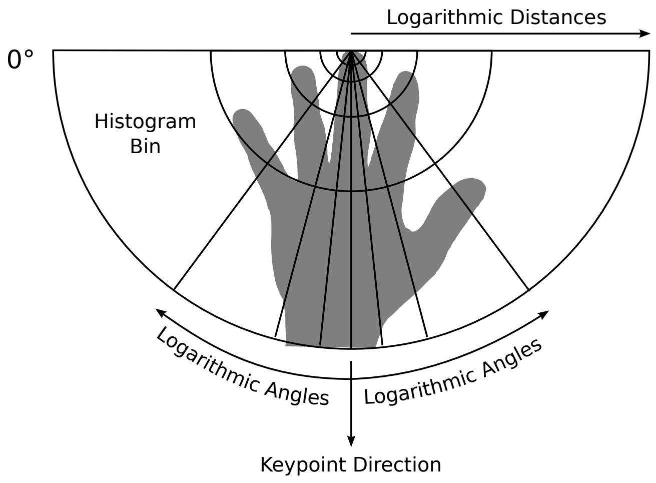


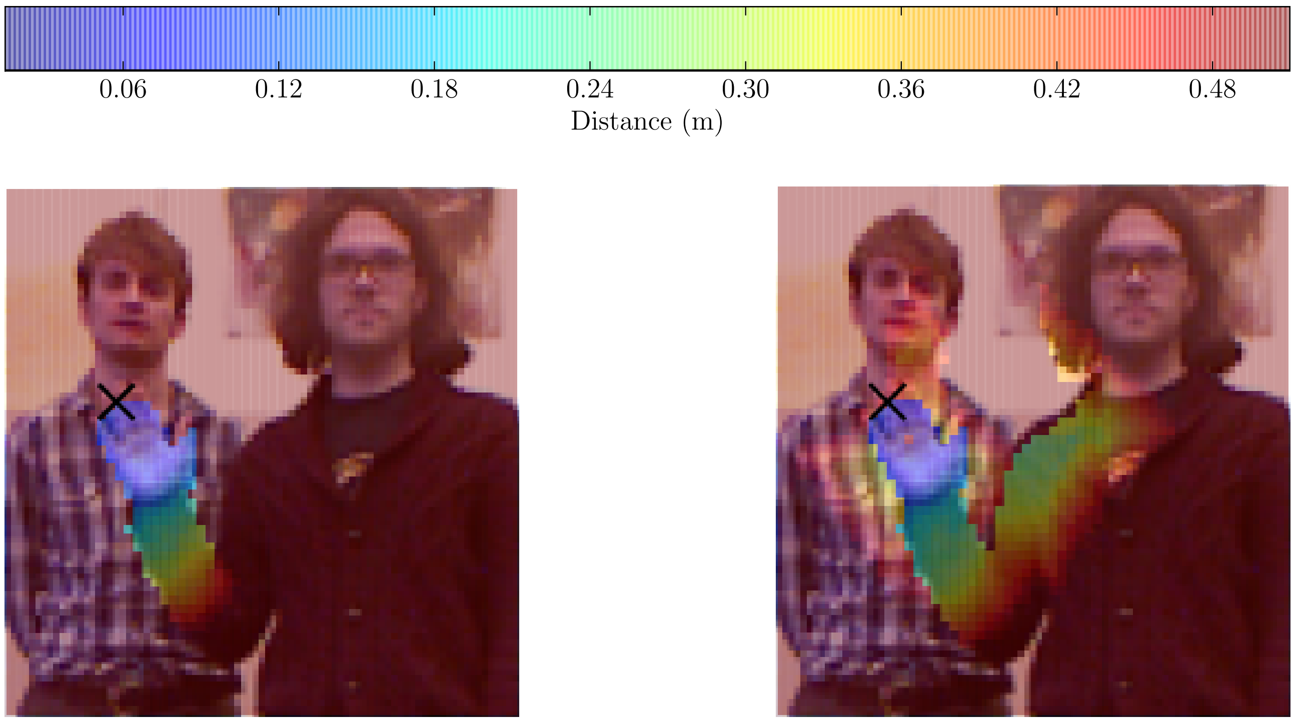
Figure 3.3: Illustration of the proposed hand descriptor histogram, based on log-polar sampling.

Using Euclidean distance to identify points that are local to a keypoint can lead to problems in crowded environments. As shown in Figure 3.4, unconnected background, other people and other body parts will frequently have a smaller Euclidean distance than points on the forearm. Geodesic distances are thus used to ensure that “distance” from a keypoint corresponds to importance to the descriptor. Dijkstra’s algorithm [132] is used to optimally compute geodesic distances.

Dijkstra’s Algorithm

The geodesic distance between two pixels in a depth image is the shortest cumulative distance between them, when traversing paths of neighbouring pixels on the same mesh. Being a graph search algorithm, the problem can be formulated in terms of the depth image as follows: nodes in the graph represent local pixels on the same mesh as the keypoint; two nodes are connected if they represent neighbouring pixels.

Two parameters govern the structure of this graph. The first, min_d, is the minimum Euclidean distance, above which neighbouring pixels are considered belonging to different meshes. Only neighbouring pixels separated by less than min_d are processed. The other parameter, max_d, is the maximum geodesic distance that pixels can have from the keypoint. Pixels with a greater distance than max_d are not processed. min_d can be equated to the low hysteresis threshold of



(a) Geodesic Distance

(b) L1 Distance

Figure 3.4: Heat map showing the advantage of using geodesic distances to highlight important regions of a hand in crowded environments. Both geodesic and L1 distances within 0.5 m are shown, from a point on the fingertips indicated by a black cross.

the Canny edge detector. max_d should be chosen through experimentation. Figure 3.4a shows the pixel regions that would lie within different values of max_d.

Before the graph is generated, the number of depth pixels with a geodesic distance to the keypoint less than max_d, is not known. Thus, graph nodes are not initially known. To compensate, a slight modification of Dijkstra's algorithm is made: neighbouring pixels with a Euclidean distance of less than min_d are iteratively added as graph nodes, if their geodesic distance from the keypoint is less than max_d. The pseudocode for this algorithm is given in Algorithm 1.

The performance of this algorithm depends heavily on the data structures $\mathcal{Q}$ and $\mathcal{R}$. If $\mathcal{Q}$ is an unsorted list, the complexity of the algorithm is given by $O\left(E + V^2\right)$, where E is the total number of edges and V is the number of nodes analysed in the graph. This arises from the iterative search over V nodes, with $DELETE - MIN_{(\mathcal{Q})}$ taking $O\left(V\right)$ each iteration. The $INSERT_{(\mathcal{Q})}$ function is simply $O\left(1\right)$ complexity, and is executed E times.

If $\mathcal{Q}$ is represented with a binary heap then the complexity is reduced to $O\left(\left(E + V\right)\log\left(V\right)\right)$.

Algorithm 1 Pseudocode of a modified version of Dijkstra's algorithm, where depth image pixels are dynamically added as nodes to visit, $\mathcal{Q}$, if their Euclidean distance to an existing node is less than min_d and their geodesic distance from the keypoint is less than max_d.

```
Require: $\mathcal{Q} = \{(\mathbf{u}_0, s_0), (\mathbf{u}_1, s_1) \ldots\}$:  ▷ an ordered list of nodes to visit, consisting of a node
   and ordered on an accumulated distance value
Require: $\mathcal{R} = \{(\mathbf{u}_0, d_0), (\mathbf{u}_1, d_1) \ldots\}$:  ▷ a map holding the results, consisting of a node and
   an accumulated distance value
Require: $\mathbf{k}$:  ▷ the keypoint for which the descriptor is to be generated
Require: $DELETE - MIN_{(\mathcal{Q})}()$:  ▷ pops the node from $\mathcal{Q}$ with the lowest distance value

  $\mathcal{Q} = \{(\mathbf{k}, 0)\}$
  $\mathcal{R} = \{(\mathbf{k}, 0)\}$
  while $\mathcal{Q} \neq \emptyset$ do
      $(\mathbf{u}, s) \leftarrow DELETE - MIN_{(\mathcal{Q})}()$
      $d \leftarrow \mathcal{R}[\mathbf{u}]$
      if $s \leq d$ then
          for all $\mathbf{v}$ = neighbours of $\mathbf{u}$ do
              $w$ = distance from $\mathbf{u}$ to $\mathbf{v}$
              if $w < min_d$ then
                  $d_v = d + w$
                  if $d_v < max_d$ then
                      if $\mathcal{R}[\mathbf{v}] == \emptyset$ or $d_v < \mathcal{R}[\mathbf{v}]$ then
                          $\mathcal{R}[\mathbf{v}] = d_v$
                          $INSERT_{(\mathcal{Q})}(\mathbf{v}, d_v)$
                      end if
                  end if
              end if
          end for
      end if
  end while
```

This is due to the $O(\log(V))$ cost for $DELETE - MIN_{(\mathcal{Q})}$, which was previously $O(V)$. However, for each analysed node's neighbour, $INSERT_{(\mathcal{Q})}$ now takes $O(\log(V))$ operations.

A Fibonacci heap, which has a $DELETE - MIN_{(\mathcal{Q})}$ complexity of $O(\log(V))$ and $INSERT_{(\mathcal{Q})}$ complexity of $O(1)$, could be used instead of the binary heap to give $O(E + V\log(V))$. However, the efficiency of the binary heap memory structure (a simple array) means that it actually performs faster in any implementation, despite its theoretical higher complexity [132]. Additionally, the most efficient implementation of $\mathcal{R}$ was found to be a static array. Even though the size has to be initialised to the depth camera resolution, in practise this was significantly faster than any map or list data structure. Note also, that because a pixel has 8 neighbours in this context, $E = 8V$.

With an algorithm defined to generate the geodesic distance of all points local to a keypoint, construction of the hand descriptor in Figure 3.3 can be detailed.

Descriptor Generation

For each keypoint, $\mathbf{k} = [x, y, z]^T$, in a depth image's corresponding point-cloud, $\mathcal{P}$, the set of local points, $\mathcal{P}_k$, are first calculated:

$$\mathcal{P}_k = \{\mathbf{x} \mid \mathbf{x} \in \mathcal{P}, g(\mathbf{x}, \mathbf{k}) < max_d\}, \tag{3.1}$$

where $g(\mathbf{x}, \mathbf{k})$ is the geodesic distance of point $\mathbf{x} = [x, y, z]^T$ from $\mathbf{k}$.

Using these local points, a 2D keypoint direction vector, $\mathbf{k}^d$, can be calculated. This vector points towards the mean of the points in $\mathcal{P}_k$:

$$\mathbf{k}^d = \mathbf{P}\left(\frac{1}{N}\sum_{\mathbf{x}\in\mathcal{P}_k}\mathbf{x} - \mathbf{k}\right),$$

where N is the number of points in $\mathcal{P}_k$ and $\mathbf{P} = [\mathbf{I}_2, \mathbf{0}_{2,1}]$ is a 2×3 matrix that projects a 3D vector onto a 2D plane parallel to the camera image plane. $\mathbf{0}_{2,1}$ is a 2×1 zero matrix.

For each local point, $\mathbf{x} \in \mathcal{P}_k$, a 2D direction vector from its keypoint, $\mathbf{k}$, is also calculated. The unnormalised vector is defined as:

$$\mathbf{x}^d = \mathbf{P}\left(\mathbf{x} - \mathbf{k}\right).$$

As shown in Figure 3.3, the normalised keypoint direction vector, $\hat{\mathbf{k}}^d$, lies at an angle of $\frac{\pi}{2}$ relative to the descriptor. Additionally, points situated more than $\frac{\pi}{2}$ radians from the keypoint direction are discarded. For the remaining points, the angular difference, θ_{xk}, between the descriptor and $\hat{\mathbf{x}}^d$ is then calculated:

$$\theta_{xk} = \cos^{-1}\left(\hat{\mathbf{x}}^d \cdot \left(\mathbf{R}\hat{\mathbf{k}}^d\right)\right) \qquad \text{if } \hat{\mathbf{x}}^d \cdot \hat{\mathbf{k}}^d >= 0$$
$$\mathbf{R} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix},$$

where $\hat{\mathbf{x}}^d$ is the normalised 2D direction vector of local point $\mathbf{x}$, and $\mathbf{R}\hat{\mathbf{k}}^d$ gives the vector from the keypoint to the 0° line of the descriptor, illustrated in Figure 3.3.

Knowing both the geodesic distance, $g\left(\mathbf{x}, \mathbf{k}\right)$, and angular difference, θ_{xk}, of all local points, the keypoint's histogram can be calculated. Rather than assign appropriate bins by taking the logarithm of these values, it is more computationally efficient to calculate the static histogram bin boundaries.

The logarithmic distance boundary, b_n^d, of bin number n, out of a total of N^d, is given as:

$$b_n^d = min_d \left(\frac{max_d}{min_d}\right)^{\frac{n}{N^d-1}}, \qquad n = 0 \ldots N^d - 1.$$

As defined in the previous subsection, neighbouring pixels are on the same mesh if their Euclidean distance is less than min_d; the maximum geodesic distance of a pixel local to a keypoint is max_d.

Angular bin boundaries must be symmetric about the keypoint direction. The maximum angular boundary lies at $\frac{\pi}{2}$ either side of this direction. Thus, the logarithmic angular boundary,

b_n^a, of bin number n out of $N^a - 1$ is defined as:

$$b_n^a = sign\left(n - \frac{N^a}{2} + 1\right) \frac{\pi}{2}^{\left|\frac{2n+2}{N^a} - 1\right|} + \frac{\pi}{2}, \qquad n = 0 \ldots N^a - 1.$$

Note that only even values of N^a are considered.

With the histogram bin boundaries defined, a local point, $\mathbf{x} \in \mathcal{P}_k$, is assigned a distance, x^i, and angular, x^j, bin number as follows:

$$x^i = \min_n \left(g\left(\mathbf{x}, \mathbf{k}\right) \leq b_n^d\right), \qquad n = 0 \ldots N^d - 1,$$

$$x^j = \min_n \left(\theta_{xk} \leq b_n^a\right), \qquad n = 0 \ldots N^a - 1.$$

Each distance-angle bin value, b_n, is the total number of local points assigned to it. The proposed hand descriptor is represented as a vector of concatenated distance-angle bin values. Distance-angle bins are ordered by increasing distance, and in an anti-clockwise direction; changing distance can be thought of as an outer loop iteration and angular change as an inner loop iteration. To make the descriptor invariant to the total number of local pixels, each bin value, b_n, should be normalised to 1. This ensures that the descriptor is scale-invariant.

$$b_n' = \frac{b_n}{\sum_{k=0}^{N^d N^a - 1} b_k}, \qquad n = 0 \ldots N^d N^a - 1.$$

With a hand descriptor generated at every edge keypoint, a SVM is used to perform hand detection. In addition to the standard training method, a technique from [26] is adopted. An initial classifier is trained on a subset of all samples. This classifier is used to detect hands in an additional dataset where none are present. Any resulting false positives are considered "hard examples", and are added to the final training set. Performance is then increased when the classifier is retrained on the expanded training set. For further reading, background theory to the SVM is presented in Appendix A.

3.2.2 Body Detection

Initial Clustering

The first stage of the proposed body detection algorithm is to segment the scene into spatially separated clusters. Removal of the floor plane is a necessary pre-requisite for this task, as it connects all subjects in the input image. With the depth camera mounted on a mobile robot, offline calibration of the floor plane unit normal, $\hat{\mathbf{n}}$, and a point in the floor plane, $\mathbf{x}_f$, can be performed.

Points, $\mathbf{x} = [x, y, z]^T$, in the input point-cloud, $\mathcal{P}$, are removed if their Euclidean distance to the floor plane is within a small threshold, ϵ:

$$\mathcal{P}' = \{\mathbf{x} \mid \mathbf{x} \in \mathcal{P}, \| (\mathbf{x} - \mathbf{x}_f) \cdot \hat{\mathbf{n}} \| > \epsilon\}.$$

With floor points filtered from point cloud $\mathcal{P}'$, a connected components algorithm [62] is used to separate $\mathcal{P}'$ into spatially separated clusters. Figure 3.5 shows the resulting clusters in a typical scene.

For many human gestures, planar arms can lie up to 0.3 m from the body; foreshortened hands can be up to 0.5 m away. This will cause hands and arms to be clustered separately from the body, as shown in Figure 3.5, unless the minimum cluster separation distance is set to a very large value. However, this will cause adjacent bodies in crowded environments to be incorrectly merged, causing the people detection method to fail. This problem motivates the following Bayesian hand-body association algorithm, described in Section 3.2.3. With this method for associating separately clustered hands and bodies, a smaller minimum clustering distance can be chosen, enabling successful people detection in crowded environments.

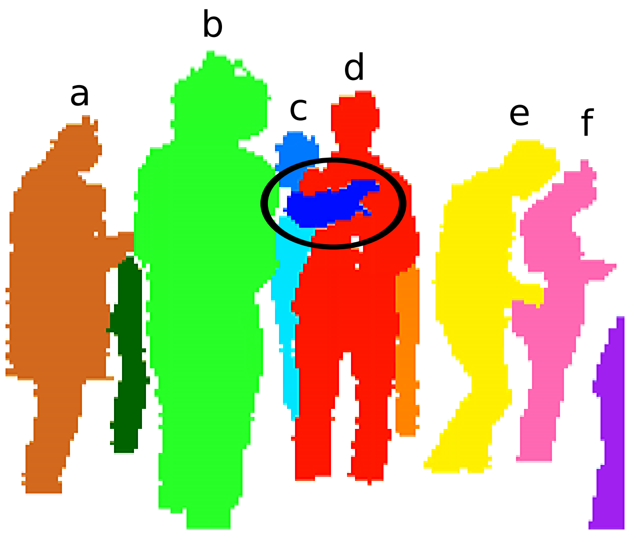


Figure 3.5: Annotated image showing typical results of the connected components algorithm on the depth point cloud. Separate clusters are coloured differently, with letters indicating the different people in the scene. The gesturing hand, highlighted by the black oval, has been clustered separately from gesturing person "d". In such crowded scenes, a method of associating hands to the correct person must be devised.

Body Detection

It can be seen from the depth image in Figure 3.5 that upper bodies all have a similar width and height, especially compared to the many other background objects that can exist in a scene. Principal component analysis of a cluster's points provides a natural way of extracting this shape information. A probabilistic filtering method can then be applied to the results, in order to separate bodies from background clusters.

Firstly, for every highest point in every cluster, local points within a set geodesic distance are extracted using Equation 3.1. For a particular cluster, these N points shall comprise the upper body set $\mathcal{P}_b$. The covariance matrix, $\mathbf{C}$, of these points, $\mathbf{x}_n = [x, y, z]^T$, is defined as:

$$\mathbf{C} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T,$$

where $\bar{\mathbf{x}}$ is the mean of the points in $\mathcal{P}_b$.

The first principal component of the points in $\mathcal{P}_b$, denoted by $\mathbf{v}_0 = [x, y, z]^T$, will give a vector running from head to toe, which is largely invariant to orientation and limb pose. Similarly, the second principal component, $\mathbf{v}_1$, will run horizontally along the upper body. $\mathbf{V} = [\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2]$, and its associated matrix of eigenvalues, $\lambda = \operatorname{diag}(\lambda_0, \lambda_1, \lambda_2)$, can be calculated from the covariance matrix using:

$$\mathbf{CV} = \mathbf{V}\lambda,$$

The eigenvalue associated with a particular principal component gives a measure of the variance of the points along its direction. Thus, the first eigenvalue, λ_0, gives a measure of upper body length. Similarly, the second eigenvalue, λ_1, gives a measure of width.

From a training set of body clusters, the mean values of λ_0 and λ_1 should be calculated, μ_0 and μ_1, along with the standard deviations, σ_0 and σ_1 respectively. f_0 shall be used to denote the body detection feature given by λ_0, and f_1 shall denote the feature given by λ_1. Due to the orthogonality property of principal components, f_0 and f_1 are independent variables:

$$\mathbf{v}_0 \cdot \mathbf{v}_1 = 0.$$

The probability that these features indicate the presence of a body, b, can then be naturally modelled by Gaussian distributions:

$$P(f_0 \mid b) \sim \mathcal{N}\left(f_0 \mid \mu_0, \sigma_0^2\right), \tag{3.2}$$

$$P(f_1 \mid b) \sim \mathcal{N}\left(f_1 \mid \mu_1, \sigma_1^2\right). \tag{3.3}$$

The Mahalanobis distance of a cluster's feature vector, $d(f_0, f_1)$, can be used as a measure of similarity to the average body in the training set. The square of this distance will be chi-square distributed, with two degrees of freedom. Setting the body detector to have a reasonable 5% false negative rate, with respect to the training set, the 0.95 quantile of this distribution occurs at $\chi_2^2 = 5.991$. Segmented clusters with an associated $d(f_0, f_1)^2 \leq 5.991$ thus represents a body in the scene; those with $d(f_0, f_1)^2 > 5.991$ can be filtered as background clusters.

$$d\left(f_0, f_1\right) = \sqrt{\sum_{n=0}^{1} \frac{\left(f_n - \mu_n\right)^2}{\sigma_n^2}}.$$

Finally, a noise-invariant reference point, $\mathbf{x}^r$, is defined for every detected body. It is calculated from the mean of the cluster's analysed point's, $\mathcal{P}_b$:

$$\mathbf{x}^r = \frac{1}{N} \sum_{\mathbf{x} \in \mathcal{P}_b} \mathbf{x}.$$

3.2.3 Hand-Body Association

Temporal tracking of detected hands and bodies is achieved using multiple independent Kalman filters [133]. The Kalman filter will be discussed in greater detail in Section 4.2.1. The task remains, however, of associating one with the other. Recursive Bayesian estimation is used to provide a solution to this problem.

If a hand is detected in a cluster that is also a known body, they can be associated with a high probability. However, separate clustering of hands and bodies occurs when the pixels connecting them are completely occluded. This can only occur with a select number of poses, such as a "stop" gesture with an outstretched hand, or when a forearm is planar with the body as shown in Figure 3.5. A natural likelihood for hand-body association in these situations can be formulated by analysing the displacement between hands and bodies.

Still ensuring generality, a separately clustered hand, h, will be at one of $\mathcal{G} = \{g_1, \ldots, g_N\}$ different displacements, relative to the body. As such, association likelihood can be most accurately represented using a Gaussian Mixture Model (GMM). This GMM is trained by having a user adopt different poses, and recording the hand-body displacement each time that the hand and body are separately clustered. The GMM is fitted to this data using the expectation-maximisation algorithm. The resulting mean displacement of component $g \in \mathcal{G}$ is μ_g, and $\mathbf{\Sigma}_g$ is the corresponding covariance matrix. The probability of hand h being associated with GMM component $g \in \mathcal{G}$, is dependent on the hand's displacement from the body, $\mathbf{d} = [x, y, z]^T$:

$$P(\mathbf{d}|g) = \mathcal{N}(\mathbf{d} \mid \mu_g, \mathbf{\Sigma}_g).$$

The likelihood of hand h being associated with a particular body, b_i, can then be given by the weighted sum of the hand probabilities for each GMM component $g \in \mathcal{G}$:

$$P(h \mid b_i) = \sum_{g \in \mathcal{G}} \pi_g P(\mathbf{d}_i|g), \tag{3.4}$$

where $\mathbf{d}_i$ is the displacement of hand h from body b_i, and π_g is the prior likelihood of GMM component g. Using Bayes theorem, the posterior probability for hand-body association is given by:

$$P(b_i \mid h) = \frac{P(h \mid b_i) P(b_i)}{\sum_{n=1}^{I} P(h \mid b_n) P(b_n)}, \tag{3.5}$$

where I is the number of detected bodies.

When modelling the prior probability, $P(b_i)$, of body b_i being associated with the hand in Equation 3.5, it would be beneficial to use temporal information. In this way, a body that had a high posterior probability of being associated with the hand in the last time period, can have a higher prior association probability in the current time period. As hands and bodies are tracked using Kalman filters, which have the Markov property, it is logical to also model the proceeding hand-body associations as a Markov process. Thus, the probability of b_i being associated with h at time t, given the associated probability at time $t-1$, is conditionally independent of all prior associations:

$$P\left(b_i^t \mid b_i^{t-1}, b_i^{t-2}, \ldots, b_i^0\right) = P\left(b_i^t \mid b_i^{t-1}\right).$$

Additionally, the likelihood of h being associated with b_i depends only on the current association, and is conditionally independent of all prior associations:

$$P\left(h^t \mid b_i^t, b_i^{t-1}, \ldots, b_i^0\right) = P\left(h^t \mid b_i^t\right).$$

Given all hand observations from time $0:t$, Equation 3.5 can be rewritten to incorporate the new temporal prior:

$$P\left(b_i^t \mid h^{0:t}\right) = \frac{P\left(h^t \mid b_i^t\right) P\left(b_i^t \mid h^{0:t-1}\right)}{\sum_{n=1}^{I^t} P\left(h^t \mid b_n^t\right) P\left(b_n^t \mid h^{0:t-1}\right)}, \tag{3.6}$$

where I^t is the number of detected bodies at time t. Given a particular hand observation, body $b_i = 1$ if it is associated with the hand, and $b_i = 0$ if they are not associated. Thus:

$$P\left(b_i^t \mid h^{0:t-1}\right) = \sum_{b_i^{t-1}=0}^{1} P\left(b_i^t \mid b_i^{t-1}\right) P\left(b_i^{t-1} \mid h^{0:t-1}\right) \tag{3.7}$$

Finally, in order for the posterior probability for each hand-body association at time $t-1$ to be set to the prior probability at time t, the following state prediction is defined:

$$P\left(b_i^t \mid b_i^{t-1}\right) = \begin{cases} 1 & \text{if } b_i^t = b_i^{t-1} \\ 0 & \text{otherwise} \end{cases}. \tag{3.8}$$

In this way, the optimal body to be associated with an observed hand, at a given time, is equal to the *maximum a posteriori* solution of Equation 3.6.

3.3 Results

The Microsoft Kinect RGB-D camera, attached to a Pioneer P3DX robot, was used for the following experiments. The depth image was downsampled to a resolution of 160×120 pixels. During hand detection, max_d, the maximum geodesic distance of pixels local to a keypoint, was set to 0.5 m. This maximises the descriptiveness of the forearm whilst reducing the variability of the bend at the elbow. Run-time performance of the hand detector averages at just over fifteen frames-per-second (FPS), with SVM classification taking over 70% of execution time.

Hand-body association performance is compared against the shape context [61] and the method from Plagemann et al. [1]. Completely different, non-overlapping, training and test datasets were used for results generation. The training dataset is described and evaluated exclusively in Section 3.3.1. In Sections 3.3.2 and 3.3.3, this training dataset is evaluated on new and different test sets. Additionally, the only difference between different runs of the test sets comes from the small amount of Kalman filter noise. Thus, due to their repeatability, the results need to be processed only once.

Results are generated using nine controlled test datasets (Section 3.3.2) and three crowded environment datasets (Section 3.3.3). Four gestures are used in each dataset for testing hand-body association: a wave to grab the robot's attention, a subtle push to have the robot leave, a subtle follow me motion, and a raised hand indicating that the robot should stop. Images of the gestures can be seen in Figure 3.9. All test datasets used have been made publicly available[1].

Ground truth hand and body positions in all datasets were manually labelled on a 640×480 resolution colour image, so that results could be automatically generated. Hand and body detection results are obtained by back-projecting the estimated 3D hand and body positions, and comparing them with ground truth. Ignoring a body part's z-component, due to 2D data labelling, hands are considered detected if they lie within 0.1 m of the ground truth. This allows for just enough error to accommodate for depth camera noise and inaccuracies from the ground truth annotation. Using the well-known tri-phase gesture model, only the stroke phase [134] (the unique component) of a gesture was analysed during results generation. Bodies are considered detected if they lie within 0.3 m of the ground truth. This distance threshold was chosen due to the fact that different body detection methods have slightly different detection points, and a reasonable assumption for a correct detection is if the detection point lies on the correct torso.

[1]http://www.imperial.ac.uk/hamlyn/eo/gesturedataset

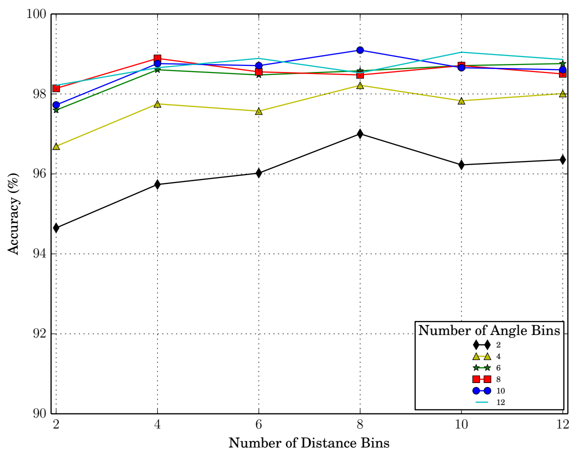


Figure 3.6: Graph showing the SVM classification accuracy of the proposed hand descriptor, for a range of angle and distance bins. Accuracy clearly increases with the number of angle bins used. Figure similar to the published paper on this work [4].

3.3.1 Algorithm Validation

Hand Descriptor Parameter Optimisation

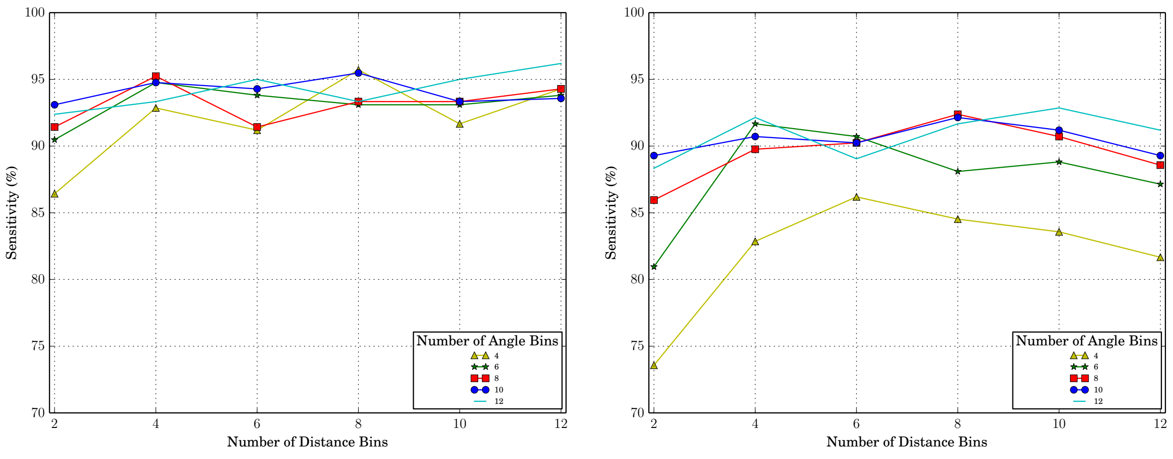
The training dataset for the hand detector consists of samples from relatively noiseless video sequences, some containing a person and others of random background. $1,400$ hands from a single user were manually labelled, along with $11,500$ randomly sampled background examples. A single split of samples into training and test sets was used for parameter evaluation, with a $70 : 30$ ratio. This resulted in a test set of 420 positive and 3450 negative samples. However, for the results presented in Sections 3.3.2 and 3.3.3, all of this data is used as the training set.

SVM classifiers were trained on the proposed descriptor, using a range of angle and distance bins. The test set classification accuracy is shown in Figure 3.6. As can be seen, the descriptor exhibits good performance, even for low numbers of distance and angle bins. Increasing numbers of angle bins results in a large accuracy increase. However, increasing numbers of distance bins does not produce such monotonic behaviour. This indicates the importance of the angular binning scheme for the descriptor's performance.

Results for the SVM trained on the descriptor with the optimal number of distance and angle bins, were further analysed. Each entry of the confusion matrix, shown in Table 3.1, shows the

Table 3.1: Confusion matrix of the optimal SVM classifier for the proposed descriptor. Rows represent the ground truth class. Columns represent the predicted class.

	Hand	**Background**
Hand	405	15
Background	16	3433



(a) Sensitivity of Proposed Descriptor (b) Sensitivity of Shape Context

Figure 3.7: Graph comparing sensitivity of the proposed descriptor and the shape context. The proposed descriptor outperforms the shape context for every number of angle and distance bins.

classifier's responses to the hand and background samples from the test set.

To more fully validate its performance, the sensitivity of the proposed descriptor was compared against that of the original shape context [61]. Using the depth image as input, the shape context histogram used the normalised number of edge points within set angles, and within set Euclidean distances. The maximum Euclidean distance of edge points to analyse was set to the optimal value of 0.3 m. This minimises the effect of background noise on the shape context. SVMs were trained on shape contexts in the same way as the proposed descriptor. Sensitivity is defined as:

$$\text{sensitivity} = \frac{TP}{TP + FN},$$

where TP denotes the number of true positives, and FN denotes the number of false negatives. The results, shown in Figure 3.7, show the improved performance of the proposed descriptor, for every combination of angle and distance bins.

Table 3.2: Table showing the improved performance of an SVM trained on the proposed descriptor, compared to one trained on the shape context.

	Accuracy	**Sensitivity**	**Specificity**
Proposed Descriptor	99.2%	96.4%	99.5%
Shape Context	98.5%	92.9%	99.2%

Optimal numbers of distance and angle bins were chosen for the shape context, and used to construct the results shown in Table 3.2. The test set accuracy, sensitivity and specificity is detailed for both descriptors, where specificity is defined as:

$$\text{specificity} = \frac{TN}{TN + FP}.$$

TN denotes the number of true negatives and FP denotes the number of false positives.

Hand-Body Association Validation

To evaluate the temporal behaviour of the hand-body association algorithm, an experiment was devised in which two people were positioned side by side. One of the subjects performed the pushing gesture for a period of time. This subject then moved their hand in front of the other person, so that their hand position would be more naturally associated with the wrong person. This process was repeated over a period of 25 seconds, with the tracked hand alternating in front of the two subjects. The effects of per-frame likelihood, given by Equation 3.4, and posterior probability, given by Equation 3.6, were recorded. The results are shown in Figure 3.8.

A higher likelihood value represents a higher hand-body association probability in the current time instant. In Figure 3.8, time periods where the hand is in front of the wrong subject are thus obvious. The posterior probability incorporates association probabilities from previous time instants. As can be seen, this gives a more stable association during transient periods of incorrectly lower likelihood. However, if the hand has a low likelihood of being associated with a body for longer than around 3 seconds, the posterior probability will decrease appropriately. This behaviour can be seen at the 20 second mark, and allows the framework to recover from

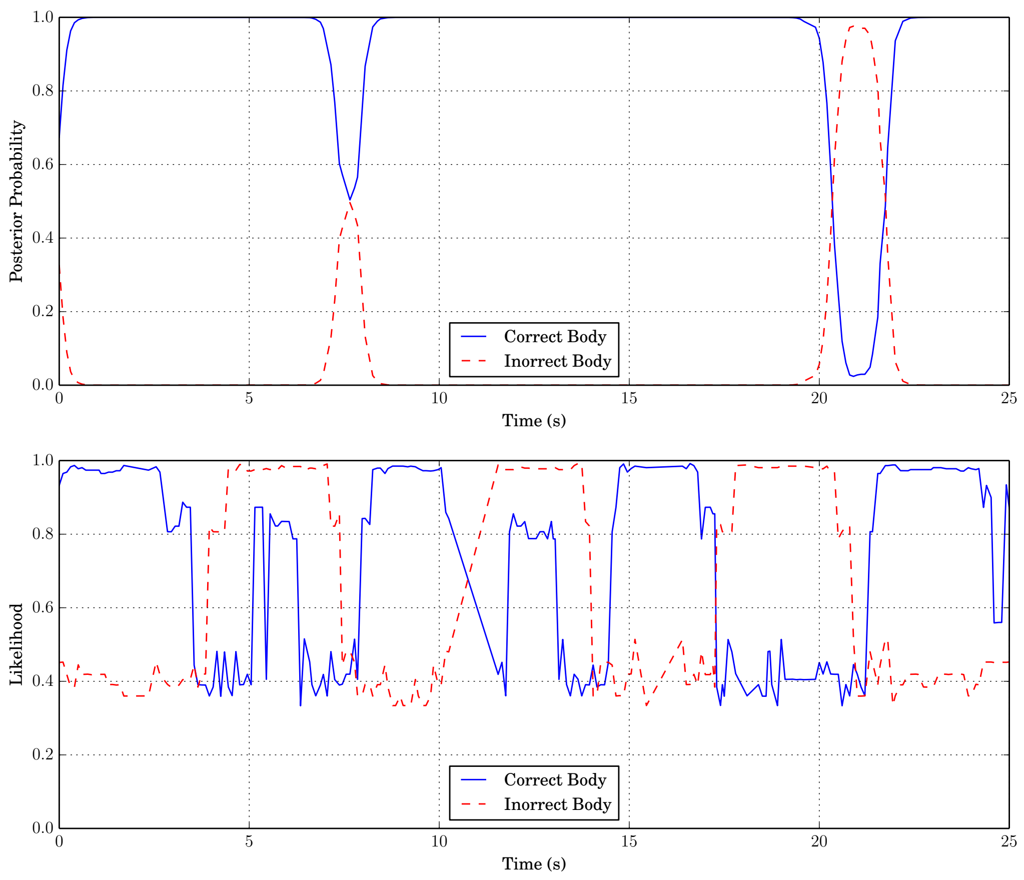


Figure 3.8: Graph showing the effect of differing per-frame likelihood on posterior probability. The posterior probability provides stable hand-body association results, during transient periods of incorrectly lower likelihood. Figure similar to the published paper on this work [4].

incorrect associations.

3.3.2 Results in Controlled Environments

To thoroughly evaluate hand detection performance, three criteria that affect hand detection difficulty were determined: angle to camera, distance to camera, and range of gesture motion. In order to rigorously evaluate the hand-body association algorithm under these parameters, nine controlled test scenarios were recorded. Each scenario is around a minute in length, and consists of a single subject gesturing towards the camera, at a constant distance but at a range of angles to the camera.

Three camera distance parameters were defined: 1.5 m, 2.0 m, 2.5 m. Similarly, three gesture

Table 3.3: Table showing the performance of the proposed body detection algorithm, in a single subject environment. Performance was compared with four alternative methods, at three different distances. **Acc.** is the percentage of accurately detected bodies. **FP** is the average number of false positives per-frame. **FPS** is the average number of frames-per-second that each method can process.

Method	**1.5 m**		**2.0 m**		**2.5 m**		**FPS**
	Acc. (%)	**FP**	**Acc. (%)**	**FP**	**Acc. (%)**	**FP**	
Proposed	98.6	0.33	95.6	0.18	88.9	0.18	30
Haar-Like Body	100	1.05	100	1.05	99.8	0.53	2
Haar-Like Face	96.1	0.03	94.4	0.02	93.7	0.03	4
HOG	27.1	0.04	4.3	0.03	5.0	0.28	6
Plagemann	26.2	0.10	49.7	0.05	53.9	0.02	10

motion ranges were defined: a large motion range using only waves, a medium motion range with half waves and half subtle gestures, and a small motion range using only subtle gestures. Each of the nine distance and motion combinations comprise the controlled scenarios.

The results of the proposed body detection method was evaluated in these single subject scenarios. For comparison, several competing methods were also evaluated on the exact same datasets. Table 3.3 shows the combined results of the nine datasets, grouped by distance. The OpenCV library implementation was used for both the Haar-like body and face detection methods [25] and the HOG people detection method [26]. The OpenCV Haar-like face detection classifier used was "haarcascade_frontalface_alt" and the Haar-like body detection classifier was "haarcascade_mcs_upperbody". The code for Plagemann's body part detector [1] was re-implemented, due to the author's original method being unavailable.

The input colour image was downsampled to a resolution of 320×240, and the input depth image was downsampled to 160×120. Clock speeds were calculated using a Dell Optiplex 880 computer, with an Intel®Core™i5-650 processor, which has 4 MB cache and a 3.2 GHz clock speed. The machine has 8 GB of Kingston DDR3 RAM, with a clock speed of 1333 MHz.

Whilst the proposed method has good performance in the single subject scenarios, the results of the Haar-like face detection algorithm are superior. However, this method runs at a maximum of 4 FPS on the testbed computer as opposed to the proposed method's 30 FPS. Indeed,

Table 3.4: Table showing hand detection performance of the proposed algorithm, compared to two alternatives, in a single subject environment. Scenarios are divided into three distance and three gesture motion range categories, each using a wide range of gesture angles. Each method is evaluated on all nine scenarios. Results are shown of both the percentage of accurately detected hands, and the average number of false positives. The best result for each scenario is shown in bold.

Method	**Large Motion**			**Medium Motion**			**Small Motion**		
	1.5m	**2.0m**	**2.5m**	**1.5m**	**2.0m**	**2.5m**	**1.5m**	**2.0m**	**2.5m**
	Accuracy (%)								
Proposed	88.8	**74.8**	**53.5**	**78.5**	**70.0**	**49.6**	**76.2**	**69.9**	**42.8**
Shape Context	**93.2**	55.6	32.3	68.7	67.7	28.9	50.0	38.6	20.6
Plagemann	49.0	47.0	34.6	15.5	30.3	20.2	17.2	23.9	14.5
	False Positives (Per-Frame)								
Proposed	**0.03**	**0.04**	**0.02**	**0.01**	**0.03**	**0.02**	**0.13**	**0.07**	**0.04**
Shape Context	0.12	0.26	0.21	0.15	0.40	0.22	0.25	0.28	0.21
Plagemann	1.17	1.26	0.99	1.37	1.41	0.87	1.40	1.38	0.98

the introduction of the proposed body detection algorithm was largely motivated by the high computational requirements of existing methods. In this way, more resources could be spent on the arguably harder problem of hand detection.

The poor performance of the HOG method can be explained by the provided OpenCV classifier's training set. This contained samples of people where the entire body, from feet to heads, were visible. Due to the placement of the camera on the mobile robot, even at 2.5 m away, the feet of the subject were cut off by the bottom of the image. Whilst this dataset is not at all a fair test of the HOG method's performance, its results are included as the employed OpenCV classifier is one of the most popular standard people detectors.

Hand detection performance was also evaluated in the nine controlled scenarios. Performance of the proposed hand detector was compared with both the shape context and the method by Plagemann et al. [1]. The entire dataset from Section 3.3.1 was used to train all three methods. The results are shown in Table 3.4.

As can be seen, the proposed method displays superior performance to the alternate methods, in almost every category. Most notable is the low number of false positives produced by the method. It can be seen that as distance to the camera increases, detection accuracy drops.

For the most part, this pattern is exhibited by all methods. This is due to the fact that depth camera noise increases with distance. At distances further than 2.5 m, this noise causes too much variation in the 3D hand points, making detection very difficult.

3.3.3 Results in Crowded Environments

In order to evaluate the complete hand-body association framework, three crowded scenarios were recorded. Each recording takes place in a different location with different lighting conditions. They shall be referred to based on the university location where they were recorded: "Computing", "Corridor", "Physics". Each recording is over a minute long; every few seconds, one or two simultaneous people from the crowd gesture towards the robot. In the same way as the controlled datasets in Section 3.3.2, people and gestures were manually annotated. Images of the proposed hand-body association method's results in these scenes are given in Figure 3.9. Tracked bodies are displayed in red, tracked hands are displayed in blue, and associations are denoted with a white line. As can be seen, these scenarios represent a challenging, real-world environment for HRI.

Hand-body association accuracy for the scenarios is shown in Table 3.5. Association accuracy is also listed for the shape context and the method by Plagemann et al.. This was generated by using the proposed hand-body association algorithm, but with body parts detected using the comparison methods. To highlight the differences between the hand detection results, body detection for the shape context was performed using the proposed body detection method.

With a minimum association accuracy of 76.7% in such a challenging environment, these results validate the success of the proposed algorithm. This is further reinforced by the 89.9% association accuracy in the "Physics" scenario, and the lower number of false positives produced by the proposed method. The hand detection method by Plagemann et al. does not perform as well as the other two methods. This might mainly be due to the fact that the algorithm was originally evaluated in much simpler, single person environments. Whilst the same training set was used for all three methods, the method by Plagemann et al. could potentially benefit most from a larger number of background samples.

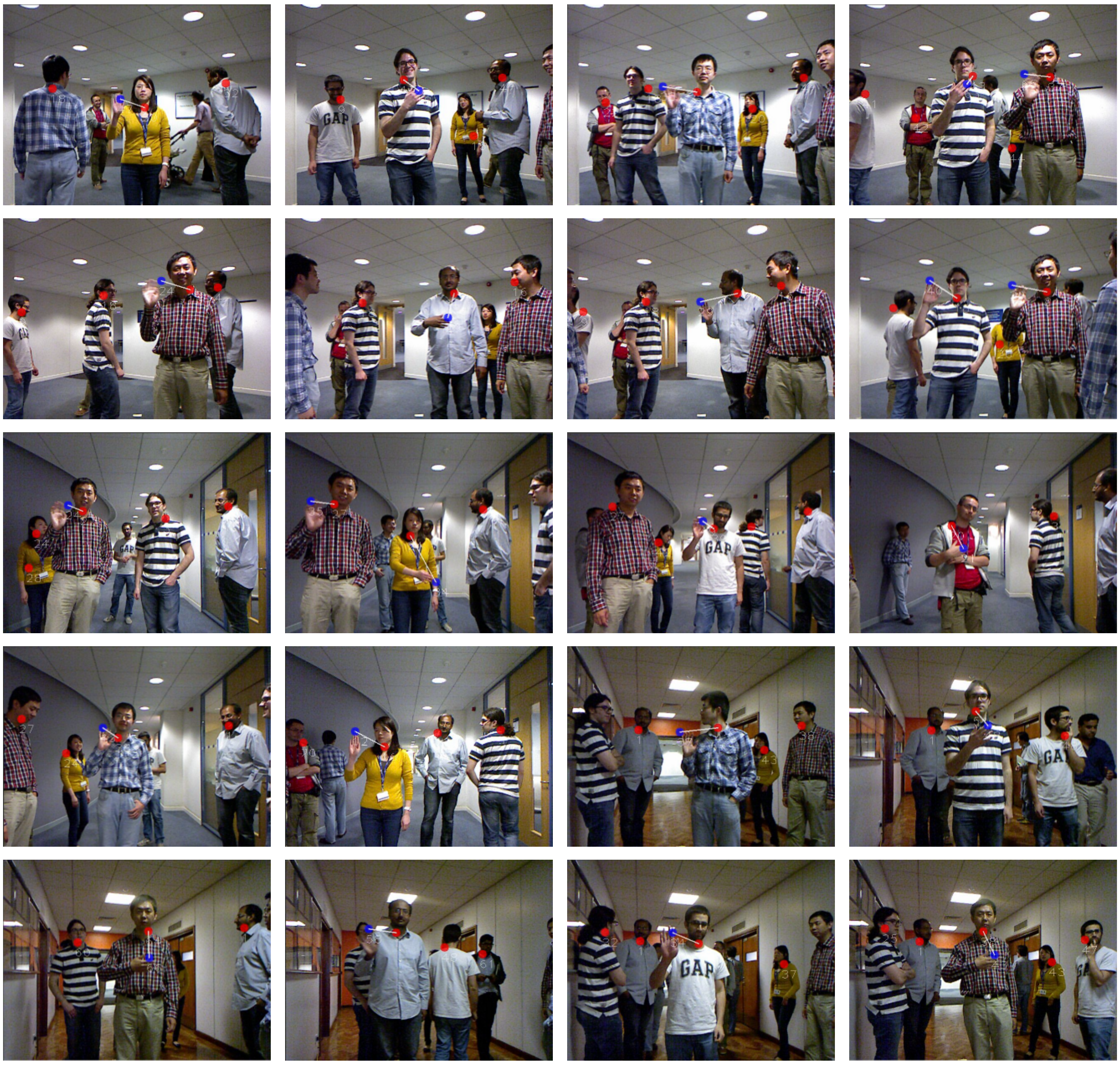


Figure 3.9: Example results showing the accuracy of the proposed hand-body association method in three different crowded scenes. Each scene has different lighting conditions, and features a range of people gesturing towards the robot. Tracked bodies are displayed in red. Tracked hands are displayed in blue. Associations are shown with a connected white line.

Table 3.5: Table showing hand-body association results for the three crowded scenarios.

	Scenario		
	Computing	**Corridor**	**Physics**
Average Gesture Length (s)	2.3	2.0	2.1
	Proposed Descriptor		
Hand Detection Accuracy (%)	82.3	80.2	83.3
Body Detection Accuracy (%)	85.1	85.1	91.9
Association Accuracy (%)	87.4	76.7	89.9
False Positive Associations per Frame	0.09	0.14	0.05
	Shape Context		
Hand Detection Accuracy (%)	57.2	56.3	42.2
Association Accuracy (%)	70.4	57.1	53.9
False Positive Associations Per Frame	1.06	0.68	0.48
	Plagemann		
Hand Detection Accuracy (%)	28.7	27.2	15.5
Body Detection Accuracy (%)	36.2	34.4	25.0
Association Accuracy (%)	29.8	24.6	10.2
False Positive Associations Per Frame	1.44	1.90	2.89

An additional reason for the reduced performance of the competing methods, is that the training set had only a single subject and did not have a wide a range of gestures as in the crowded environments. Thus, the proposed method was best at generalising to hand samples that it had not been trained on. This is due to the use of geodesic distances in the proposed hand descriptor, allowing it to perform well in crowded environments and recognise unseen hand poses.

Association accuracy can sometimes seen to be higher than the hand detection accuracy. This can be explained as a result of the subtleties of the gestures. If a Kalman filter is created at the start of a subtle gesture, it can lie within 0.1 m of the ground truth for much of its duration, even without further detections to update its position. Although decreasing the ground truth threshold below 0.1 m can alleviate this effect, this results in incorrectly reduced detection accuracy.

In spite of these results, a source of inaccuracy in the proposed method can be identified, being that the hand detector is susceptible to self-occlusions. When a foreshortened hand is presented that the detector was not trained on, detection frequently fails. Shorter gestures are usually less pronounced, and thus exhibit this behaviour more often. Decreasing max_d can alleviate this problem, at the expense of increased false positives.

Section 2.3.2 reviewed full body pose estimation methods. A popular pose estimation method is the NITE skeleton tracker by PrimeSense Inc.. Like many similar methods, the user was required to adopt a stationary "T-pose" in order to start tracking. A somewhat recent update to the method removed this constraint. However, during testing, the average time to start tracking was still 2.1 s, with the user guide stating that initial calibration takes around 3 s. With the longest average gesture length being 2.3 s long, this method could not be evaluated on the presented datasets.

3.4 Conclusions

The contribution of this chapter has been the introduction of a framework for hand and body association in crowded environments. The proposed method has three main novelties: a hand detection method, optimised for crowded environments; a computationally efficient body detection method; and a probabilistic algorithm for associating detected hands and bodies. Hand detection is performed with an optimised histogram descriptor, filled with segmented depth pixels belonging to the hand. Geodesic distances are used to filter background noise in crowded environments. This technique is also adopted in the body detection method, which runs in parallel. Subsequent hand-body association is performed using a Gaussian mixture model, trained on the displacement between hands and bodies, and recursive Bayesian estimation.

Detailed hand and body detection results were presented in nine controlled scenarios. Performance was compared against four state-of-the-art body detection methods, and two hand detection techniques: the shape context [61] and the method presented by Plagemann et al. [1]. Quantitative analysis of the complete hand-body association framework was performed in a number of crowded environments. The proposed method was shown to give increased detection accuracy against competing methods, with a low number of false positives. All datasets used in this chapter have been made publicly available.

Presented results highlighted the ability of the proposed hand detector to recognise unseen gestures and subjects in crowded environments. This is due to the use of geodesic distances in isolating information local to a keypoint. A comparison of the per-frame hand-body association likelihood and the posterior probability, shows that the hand-body association algorithm uses past association outcomes to reduce transient misassociations due to noise or misdetections. Due to the computational efficiency of the proposed body detector method, the whole framework runs in real-time.

The work presented in this chapter is used repeatedly throughout the rest of this thesis. For example, tracking has been performed in this chapter using Kalman filters. Chapter 4 will explore hand tracking strategies, and specifically address whether RGB information can be used to increase the accuracy of the depth methods presented here. This can be used to provide added context for deictic gestures, or for ignoring gestures that are part of human-human interaction. Body detection is an important component of almost every aspect of HRI. Chapter 5 will use the proposed method to assign gazes to people. This will allow a robot to not only understand the presence of people in the scene, but detect their focus of attention as well. Chapter 6 will use the proposed body detector in detecting groups of people. A robot that understands which people are interacting together can integrate more naturally into human environments. For example, a robot should not navigate between two conversing people.

Chapter 4

An Asynchronous RGB-D Sensor Fusion Hand Tracking Framework using Monte-Carlo Methods

4.1 Introduction

The wide range of applications that mobile robots will perform in the future, such as caregiving and companionship, necessitates many different sensing methods. Most notably in crowded environments, including hospitals and nursing homes, robust methods of body part localisation are vital for HRI problems such as group detection and gesture recognition. These, and many other HRI fields, have been an actively researched for many years, yet limited work has been successfully translated from lab-based research into real-world applications. Real-world environments pose many new challenges, such as crowds and changing dynamics of moving objects.

The work in this chapter was partially presented in S. McKeague, J. Liu, and G.-Z. Yang, "An asynchronous rgb-d sensor fusion framework using monte-carlo methods for hand tracking on a mobile robot in crowded environments," in *Social Robotics*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2013, vol. 8239, pp. 491–500

The contribution of this chapter is to provide a hand tracking method, capable of operating in crowded and dynamic environments, in order to facilitate HRI in real-world applications.

Many methods of localising body parts can be divided into two stages: detection and tracking. In Chapter 3, hand and body detection methods were proposed. However, a method of persistently tracking each detected hand over time must also be specified. In Equation 3.6 for example, a body was given a high prior probability of being associated with a hand, if they had been previously associated. The required tracking of hands and bodies in time was accomplished with the Kalman filter. However, many previous methods have been proposed, with specific advantages and disadvantages.

Existing hand tracking literature can be broadly divided into two categories [135]: appearance-based methods, reviewed in Section 2.3.1, and model-based methods, reviewed in Section 2.3.2. Appearance-based methods usually track the hand as a single point, whilst model-based methods give a more complete pose estimate. Appearance-based methods extract a set of image features from input information, and classify these as belonging to a hand or not. Model-based methods match an explicit prior model of the hand to observed input information. Appearance-based methods are generally faster and work at longer ranges than model-based methods, but only recognise a discrete number of hand poses.

For HRI tasks, which have real-time and working range requirements, appearance-based methods are a natural choice. However, problems such as misdetections, caused by occlusions, and reduced frame rates, caused by computational expense, are exacerbated in crowded and dynamic environments. Again, it is noted that no top ranking method of the 2012 ChaLearn Gesture Challenge [128] tracked individual body parts. To fulfil the need for robust hand tracking solutions in real-world environments, this chapter proposes augmenting a hand detector, which uses depth images, with a tracking algorithm that uses colour images. With an efficient RGB method of updating the tracking position, the computational expense of the depth-based hand detector is mitigated, and position and velocity accuracy are increased.

The proposed sensor-fusion hand tracker has several novelties. Firstly, a detection-less, Monte-Carlo RGB update method is used to efficiently update the hand position, when the underlying depth-based detector fails. This ensures accurate hand position updates at a 30 Hz rate, regardless of the speed of the depth detector. Secondly, unlike most RGB hand trackers, the method dynamically learns a skin colour distribution of each tracked subject. This ensures that the system works seamlessly with people of different races, and is robust to illumination changes and worn clothing. Finally, a novel asynchronous sensor update method is used to combine depth and RGB input sources together, for real-time applications.

Results of the sensor-fusion algorithm are compared against the Kalman filter, for three different underlying depth-based hand detectors, including the method proposed in Chapter 3. To access the method's practical value, experiments are conducted in a number of crowded and controlled environments. The sensor-fusion algorithm is shown to increase both hand-tracking accuracy and velocity reconstruction.

4.2 Method

A diagram of the proposed sensor fusion algorithm's system structure is shown in Figure 4.1. Rounded boxes represent functional components of the algorithm. Square boxes denote inputs and outputs. Boxes within the dashed line are components of the proposed tracker. Boxes outside represent the algorithm's two inputs: an RGB image registered to a corresponding depth image, and a hand detector that uses the depth image to return a single 3D position and 3D unit direction vector for each hand.

The proposed hand tracker can use any depth-based hand detector as input. Section 4.3 presents results using: the hand detector proposed in Chapter 3, the shape context [61], and the method by Plagemann et al. [1]. These methods operate at around 15 frames-per-second, but can achieve low false positive rates.

The proposed tracker is comprised of four main components. The hand state stores probability distributions of its position, direction and skin colour estimates. Random sampling is used to generate a number of point positions, near the expected hand. An RGB likelihood function weighs each sample's colour similarity, in the latest image, to the expected hand colour. A state

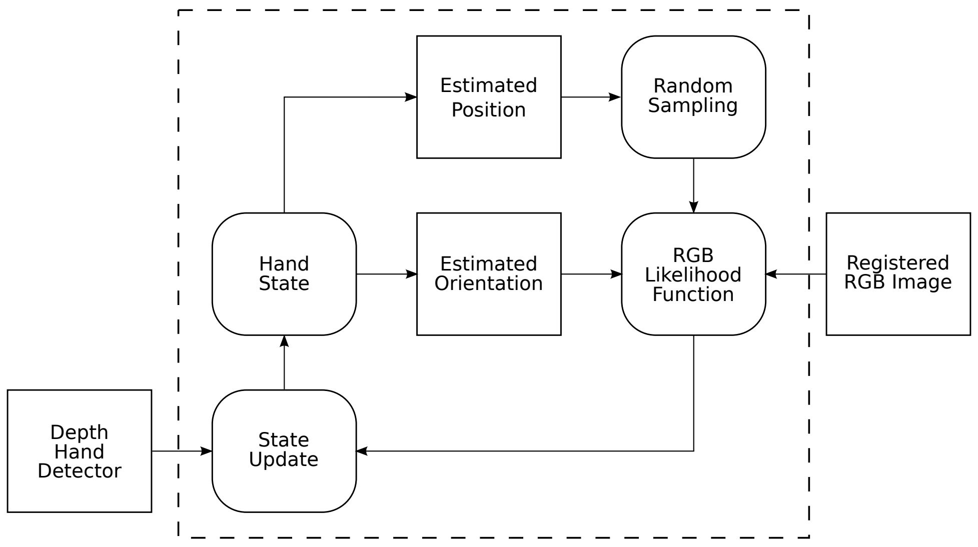


Figure 4.1: System structure diagram of the proposed sensor fusion algorithm. Rounded boxes represent functional components. Square boxes denote inputs and outputs. Boxes within the dashed line are components of the proposed tracker. Boxes outside represent the algorithm's two inputs.

update method asynchronously combines depth and RGB measurement updates to maintain a temporally consistent hand state.

Section 4.2.1 will describe the position and direction components of the hand state, and how to incorporate tracker updates from the depth-based hand detector. Section 4.2.2 details the detection-less Monte-Carlo method of updating the hand position RGB images. Section 4.2.3 describes the RGB likelihood function that weighs the random samples. Section 4.2.4 describes how to update the hand's colour and direction using RGB images. Finally, Section 4.2.5 describes the asynchronous state update method.

4.2.1 Hand State Prediction and Depth Update

The proposed tracking algorithm adapts the Bayes filter [136], also used in Chapter 3, to fuse measurement updates from two sources with different characteristics. The first is a hand detector using depth images, that provides a single value hand estimate with high precision but lower speed. The second is a hand likelihood function over the whole RGB image that is computed quickly, but with higher false positives.

A hand state, $\mathbf{s}$, is modelled with a 3D position, $\mathbf{x} = [x, y, z]^T$, and unit vector direction, $\overrightarrow{\mathbf{x}} = [ux, uy, uz]^T$: $\mathbf{s} = [x, y, z, ux, uy, uz]^T$. A colour component is independently stored, which is detailed in Section 4.2.4. The goal of the filter at time t is to maintain a probability distribution of the hand state, given all prior measurements from time 1 to t, $\mathbf{z}_{1:t}$, and knowledge of how the hand moves in the absence of any measurement:

$$P(\mathbf{s}_t \mid \mathbf{z}_{1:t}) = \frac{P(\mathbf{z}_t \mid \mathbf{s}_t)\, P(\mathbf{s}_t \mid \mathbf{z}_{1:t-1})}{P(\mathbf{z}_t \mid \mathbf{z}_{1:t-1})}. \tag{4.1}$$

Assuming that the changing hand state is a Markov process, Equation 4.1 can be written recursively:

$$P(\mathbf{s}_t \mid \mathbf{z}_{1:t}) \propto P(\mathbf{z}_t \mid \mathbf{s}_t) \int P(\mathbf{s}_t \mid \mathbf{s}_{t-1})\, P(\mathbf{s}_{t-1} \mid \mathbf{z}_{1:t-1})\, \mathrm{d}\mathbf{s}_{t-1}. \tag{4.2}$$

At time t the sensor fusion algorithm could receive either a depth measurement, $\mathbf{z}_t^{depth}$, or an RGB measurement, $\mathbf{z}_t^{RGB}$. The following section will formulate equations to estimate the hand state given depth measurements. Equations to process RGB measurements will be presented later, with position updates detailed in Section 4.2.2 and colour and direction updates detailed in Section 4.2.4. Due to their independence:

$$P(\mathbf{z}_{1:t}) = P\left(\mathbf{z}_{1:t}^{depth}\right) P\left(\mathbf{z}_{1:t}^{RGB}\right).$$

Solutions to Equation 4.2, are commonly split into two parts, called the "prediction step" (Equation 4.3) and "update step" (Equation 4.4). The "prediction step" defines how the hand state changes from a previous estimate, $\mathbf{s}_{t-1}$, to the current estimate, $\mathbf{s}_t$, in the absence of any measurement. The "update step" defines how the current hand state changes in response to a measurement, $\mathbf{z}_t$.

$$\overline{bel}(\mathbf{s}_t) = \int P(\mathbf{s}_t \mid \mathbf{s}_{t-1})\, P(\mathbf{s}_{t-1} \mid \mathbf{z}_{1:t-1})\, \mathrm{d}\mathbf{s}_{t-1}, \tag{4.3}$$

$$bel(\mathbf{s}_t) = \eta\, P(\mathbf{z}_t \mid \mathbf{s}_t)\, \overline{bel}(\mathbf{s}_t). \tag{4.4}$$

$\overline{bel}(\mathbf{s}_t)$ is also known as the prior state distribution; although it estimates the hand state at the current time, it doesn't include any measurement at the current time. $bel(\mathbf{s}_t)$, which does include measurement information for the current time, is also known as the posterior state distribution.

All depth-based hand detectors give a single point estimate of the hands position and direction: $\mathbf{z}^{depth} = [x, y, z, ux, uy, uz]^T$. The most natural measurement probability for the depth-based detection, given the current hand state, $P\left(\mathbf{z}^{depth} \mid \mathbf{s}\right)$, is thus a Gaussian probability density function (PDF). It is therefore natural to also model the hand state PDF, $P(\mathbf{s} \mid \mathbf{z})$, as a Gaussian. This is more computationally efficient than a non-parametric approach, such as a particle filter.

Under these conditions, the Kalman filter is used to provide a solution to the "prediction step", and "depth update step" (Equations 4.3 and 4.4). The Kalman filter represents both the prior and the posterior hand state PDFs as Gaussians: $\overline{bel}(\mathbf{s}_t) = \mathcal{N}\left(\overline{\mu_t}, \overline{\mathbf{\Sigma}_t}\right)$, $bel(\mathbf{s}_t) = \mathcal{N}(\mu_t, \mathbf{\Sigma}_t)$.

The "prediction step" of the Kalman filter specifies that between times $t-1$ and t, the state is linearly transformed with added Gaussian noise:

$$\mathbf{s}_t = \mathbf{A}\mathbf{s}_{t-1} + \mathbf{w}_{t-1} \tag{4.5}$$

$$P(\mathbf{w}) \sim \mathcal{N}(0, \mathbf{Q}). \tag{4.6}$$

Due to the unpredictable nature of hand motion during HRI gestures, the most appropriate motion model is Brownian motion. Thus $\mathbf{A} = \mathbf{I}_6$. The prior hand distribution, $\overline{bel}(\mathbf{s}_t)$, can then be calculated according to:

$$\overline{\mu_t} = \mu_{t-1} \tag{4.7}$$

$$\overline{\mathbf{\Sigma}_t} = \mathbf{\Sigma}_{t-1} + \mathbf{Q}_t \tag{4.8}$$

To incorporate a measurement from the depth-based hand detector at the current time, $\mathbf{z}_t^{depth}$, the "depth update step" of the Kalman filter also linearly transforms the state, with added

Gaussian noise:

$$\mathbf{z}_t = \mathbf{H}\mathbf{s}_t + \mathbf{v}_t \tag{4.9}$$

$$P(\mathbf{v}) \sim \mathcal{N}(0, \mathbf{R}). \tag{4.10}$$

As the depth-based hand detector gives a direct estimate of position and direction, $\mathbf{H} = \mathbf{I}_6$. The posterior hand distribution, $bel(\mathbf{s}_t)$, is then calculated according to:

$$\mathbf{K}_t = \overline{\boldsymbol{\Sigma}_t}\left(\overline{\boldsymbol{\Sigma}_t} + \mathbf{R}_t\right)^{-1} \tag{4.11}$$

$$\mu_t = \overline{\mu_t} + \mathbf{K}_t\left(\mathbf{z}_t - \overline{\mu_t}\right) \tag{4.12}$$

$$\boldsymbol{\Sigma}_t = \left(\mathbf{I}_6 - \mathbf{K}_t\right)\overline{\boldsymbol{\Sigma}_t}, \tag{4.13}$$

where $\mathbf{K}_t$ is known as the Kalman gain. Smaller values in $\mathbf{R}_t$, result in a higher gain, meaning the tracker places more weight on the hand detections, rather than the motion model. Due to the accuracy of depth-based hand detectors, the measurement covariance, $\mathbf{R}_t$, will be very small. The direction component of μ_t is re-normalised after every "depth update step".

With J independent hand trackers at time t, and K hands detected by the depth-based detector, the data association problem is tackled with the Hungarian algorithm [137]. This algorithm provides optimal hand-tracker assignments, unlike a naïve, nearest neighbour approach. A 3×6 matrix, $\mathbf{P} = [\mathbf{I}_3, \mathbf{0}_{3,3}]$, shall be used to extract the position component from a combined position and direction vector, where $\mathbf{0}_{3,3}$ is the 3×3 zero matrix. A detected hand shall be denoted as: $\mathbf{x}$. Thus, with a $J \times K$ cost matrix of the distance between each mean hand state, $\mathbf{P}\overline{\mu}$, and each detected hand, $\mathbf{P}\mathbf{x}$, the Hungarian algorithm finds the optimal assignment. With a particular tracker optimally assigned to a detected hand, $\mathbf{x}$, an additional distance threshold is applied to prevent wrong matches. The assignment is discarded if: $\|\mathbf{P}\left(\overline{\mu_t} - \mathbf{x}\right)\| > \epsilon^d$. The threshold, ϵ^d, should be set to the maximum distance that the hand could travel, given the maximum number of allowed misdetections. Without this threshold, a misdetected hand could result in the tracker being associated with a far away false positive.

For each unassigned hand, a new tracker is created. The differential entropy of $\overline{bel}\left(\mathbf{s}_t\right)$ is a measure of the uncertainty about the tracked hand state. A tracker is removed when the differential entropy of its distance components exceeds a threshold:

$$\frac{1}{2}\log\left|2\pi e\,\mathbf{P}\overline{\boldsymbol{\Sigma}_t}\mathbf{P}^T\right| > \epsilon^e, \tag{4.14}$$

where the bars represent the matrix determinant. The threshold ϵ^e should be chosen based on the number of hand misdetections to allow, before assuming that the hand is lost.

The best hand estimate at any point in time is given by $\overline{\mu_t}$. However, due to the use of the Brownian motion model in Equation 4.7, this will be no more accurate than the most recent "depth update step", given by Equation 4.12. RGB information from the current frame can give a better state estimate than Brownian noise. The proposed sensor-fusion algorithm thus has two requirements: use RGB information to compensate for hand misdetections in depth images, and better predict hand motion with a 30Hz RGB update process. These are accomplished by effectively modelling the RGB measurement probability: $P\left(\mathbf{z}^{RGB} \mid \mathbf{s}\right)$.

The "RGB update step" description will be separated into three steps: position, direction and colour. Hand position updates are detailed in Section 4.2.2. This refined position is used in Section 4.2.4, which details updates to the hand state's direction and colour components.

4.2.2 Detection-less Monte Carlo RGB Position Update

The task of robust hand detection in RGB images is much more difficult than using depth images. Figure 4.2 shows a common method of skin colour detection [5] applied to a typical HRI scene. To perform hand detection on these results, a somewhat arbitrary blob size would have to be chosen. Blobs above this size would be considered hands, resulting in many false positives. Additionally, misdetections could result from subjects with darker skin colours. Hand blob size affects the calculation of a single point to represent the hand; partial occlusions and varying sleeve lengths would cause much frame to frame instability in this calculation.

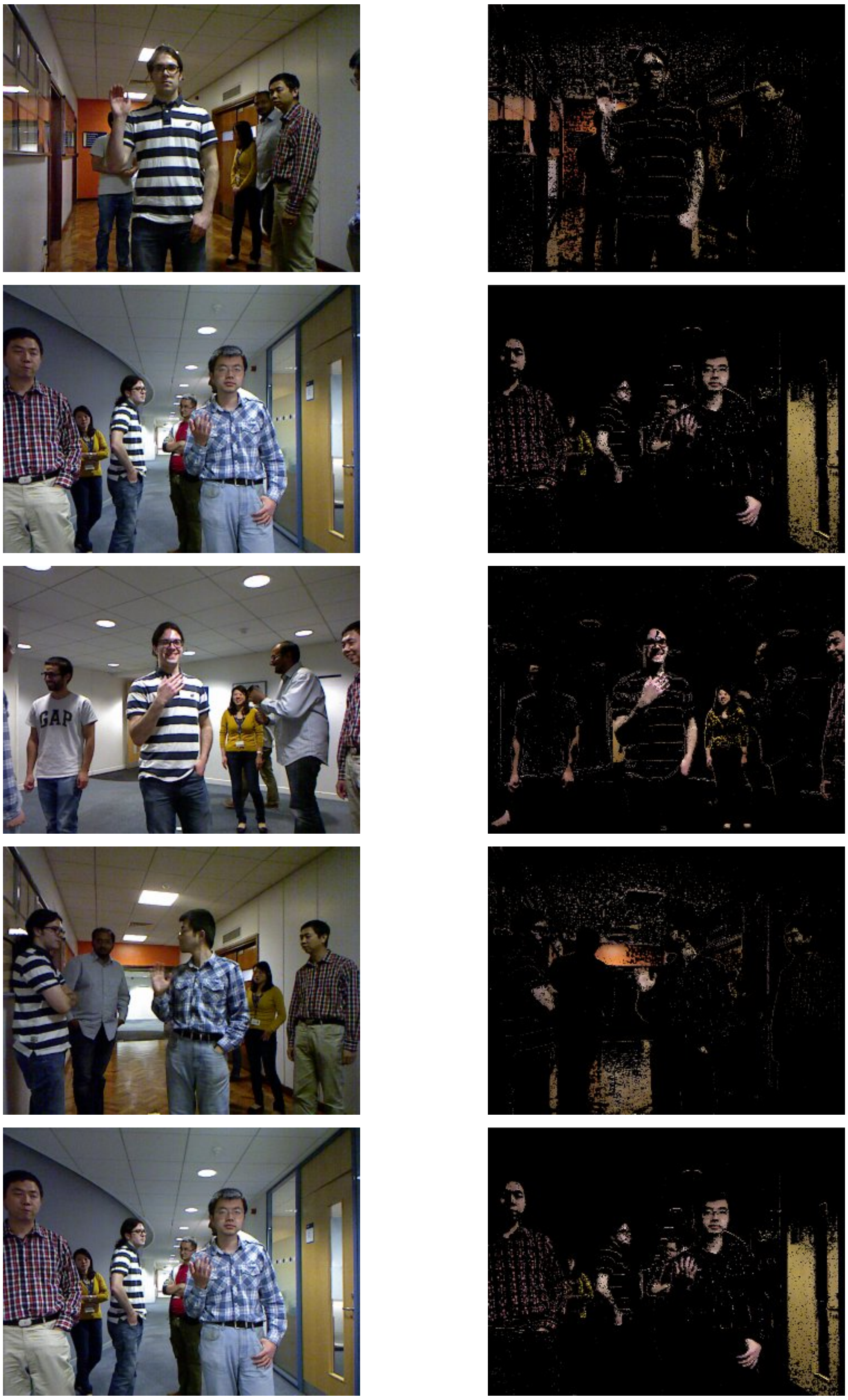


Figure 4.2: Images showing the results of an existing skin colour detection method [5] in HRI scenes. The varying hand blob sizes and large number of false positives, due to clothing style and illumination conditions, motivates the decision not to perform explicit hand detection in the RGB image.

Taking these disadvantages into consideration, explicit hand detection in the RGB image should be avoided. Thus, rather than the usual Gaussian PDF, an alternate formulation for $P\left(\mathbf{z}^{RGB} \mid \mathbf{s}\right)$ should be specified. The measurement covariance for the "depth update step", $\mathbf{R}_t$, is very small. After a depth update, the resulting $bel\left(\mathbf{s}\right)$ will be a narrow distribution near the detected hand. In the subsequent time period, there is a high likelihood that the hand will be nearby. By analysing adjacent skin coloured regions, an efficient estimate of the hand's subsequent movement can be calculated. Taking random samples from the latest prior PDF, $\overline{bel}(\mathbf{s}_t)$, provides a good way of evaluating this nearby information.

As noted in Figure 4.1, each input image is a combined depth image registered to an RGB image. Thus, every pixel will have six components: $[x, y, z, r, g, b]^T$. Each sample, m of M, will have two defined properties. The first is its state, $\mathbf{s}_t^{[m]} = [x, y, z, ux, uy, uz]^T$, drawn from $\overline{bel}\left(\mathbf{s}_t\right)$. If the sample does not lie within a distance threshold, ϵ, of its closest registered pixel in the input image, $\mathbf{c}_t^{[m]}$, then it is discarded. The threshold, ϵ, should be set to the distance that the hand could travel since the last "RGB update step". In the event that the hand is occluded or has moved too far from the tracker, this threshold ensures that registered pixels far away from the sample are not considered. The second sample property is its weight, $w_t^{[m]}$, based on the skin colour probability of its closest pixel. Again, matrix $\mathbf{P}$ is used to extract the a vector's position component:

$$\begin{aligned} &\text{sample } \mathbf{s}_t^{[m]} \sim \overline{bel}(\mathbf{s}_t), && \text{if } \|\mathbf{P}\left(\mathbf{s}_t^{[m]} - \mathbf{c}_t^{[m]}\right)\| < \epsilon, \\ &w_t^{[m]} = P\left(\mathbf{z}_t^{RGB} \mid \mathbf{c}_t^{[m]}\right), && 1 \leq m \leq M. \end{aligned}$$

Provided the sample weights are normalised, the complete sample set represents the position component of the non-parametric posterior PDF, $P\left(\mathbf{s}_t \mid \mathbf{z}_t^{RGB}, \mathbf{z}_{1:t-1}\right)$. The RGB likelihood function itself,$P\left(\mathbf{z}_t^{RGB} \mid \mathbf{c}_t^{[m]}\right)$, will be defined in Section 4.2.3. After the Kalman update step, from Section 4.2.1, the posterior covariance is never more than the prior covariance: $\boldsymbol{\Sigma} \leq \overline{\boldsymbol{\Sigma}}$. Because no explicit hand detection is performed in the "RGB update step", it can happen every frame with a reasonable RGB likelihood. If the Kalman equations were used in the "RGB update step", the resulting low covariance would limit samples to a small area. It would also keep the tracker's differential entropy low, making the removal of the tracker quite unpredictable.

Adjusting the position component of the covariance, $\mathbf{\Sigma}$, only during "depth update steps" would have the desirable effect of spreading RGB samples over a wider area, the more misdetections the depth-based hand detector makes. With enough depth misdetections, the tracker's entropy, given by Equation 4.14, would increase to the point where the tracker is deleted. RGB updates can then be used to refine the expected hand state, μ, for every input image without a "depth update step", whether due to misdetections or the lower frame rate of depth-based hand detectors.

The updated hand position in the "RGB update step" is given by the weighted sum of samples. All samples are discarded and redrawn in subsequent frames. The mean position of the tracker's posterior PDF is then updated as:

$$\begin{aligned} \hat{w}_t^{[m]} &= \left[\sum_{n=1}^{M} w_t^{[n]}\right]^{-1} w_t^{[m]}, \\ \mathbf{P}\mu_t &\leftarrow \mathbf{P}\sum_{m=1}^{M} \mathbf{s}_t^{[m]}\hat{w}_t^{[m]}, \end{aligned} \tag{4.15}$$

where $\leftarrow$ represents the assignment operator.

4.2.3 RGB Likelihood Function

The previous section described the sampling process used to update the hand position in the "RGB update step". However, the task remains of defining the likelihood function that generates sample weights: $P\left(\mathbf{z}^{RGB} \mid \mathbf{c}^{[m]}\right)$. The proposed method adaptively learns skin colour on a per-subject basis, providing invariance to varying illumination conditions.

There are two main facets governing skin colour detection methods in RGB images. A colour space model (CSM) that represents colour values, and a colour distribution model (CDM) that models these values [138]. Three main forms of CDM exist [11]: explicitly defined skin regions, non-parametric skin segmentation and parametric skin segmentation. Non-parametric histograms are the best CDM choice for the proposed sensor fusion hand tracker, as a result of their greater suitability for online learning. Additionally, the hue, saturation, value (HSV) CSM is selected, due to its ability to remove illumination effects [138]. Thus skin colour is modelled as a 2D histogram containing hue and saturation values.

The mean of each likelihood function, depth and RGB, should be a constant point on the hand. Weighing a sample by skin colour probability alone, would position the mean of the RGB likelihood function near the palm. Short sleeve lengths could cause this point to vary. The main depth detectors considered [1, 4] provide hand detections around the fingertips. As the fingertips provides a stable point about which to track the hand, the RGB likelihood mean should lie over them.

To centre the mean of the RGB likelihood function on the fingertips, two additional probability functions are introduced. The first is a distance transform, where the likelihood, P^d, of a registered pixel, $\mathbf{c}$, depends on its 3D distance to the closest edge, $edge\left(\mathbf{c}\right)$. The second likelihood function is an angular weighting, where likelihood, P^a, depends on the perpendicular distance to the hand direction vector. To calculate this, a vector from the registered pixel to the hand mean shall be denoted as $\mathbf{c}^d = \mathbf{P}\left(\mathbf{c} - \overline{\mu}\right)$. Additionally, a matrix, $\mathbf{D} = [\mathbf{0}_{3,3}, \mathbf{I}_3]$, shall be used to extract the direction component of the hand state:

$$
\begin{aligned}
P^d\left(\mathbf{c}\right) &= e^{-\alpha edge\left(\mathbf{c}\right)^2} \\
P^a\left(\mathbf{c}\right) &= e^{-\beta\left\|\mathbf{c}^d - \left(\mathbf{c}^d \cdot \mathbf{D}\overline{\mu}\right)\mathbf{D}\overline{\mu}\right\|}
\end{aligned}
$$

α and β are scaling parameters, controlling the spread of the probability around areas of maximum likelihood. α and β should be chosen such that the distributions P^d and P^a are wide enough to assign a repeatably high probability to samples near the hand edge and direction, given the nature of the Monte-Carlo process. Figure 4.3 graphically displays the likelihood functions P^d and P^a.

The combined likelihood function for pixel $\mathbf{c}$, $P\left(\mathbf{z}^{RGB} \mid \mathbf{c}\right)$, is simply the product of the three individual likelihood functions. $P^c\left(\mathbf{c}\right)$ shall represent the skin probability of pixel $\mathbf{c}$, obtained

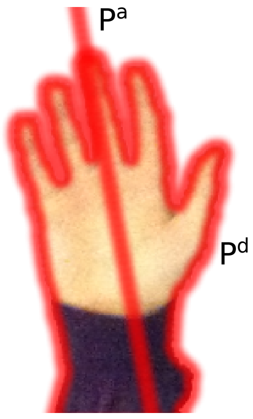


Figure 4.3: Illustration of the effect of the two likelihood functions, P^d and P^a. Areas of high probability are coloured red. P^d weighs points along the edge of the hand with a high probability; P^a weighs points along the hand direction with a high probability. When combined with skin colour probability, the intersection of all three probability functions lies over the fingertips, which is the tracked point on the hand.

from the current tracker's skin colour histogram. The mean of this combined likelihood function will lie over the fingertips, as desired:

$$P\left(\mathbf{z}^{RGB} \mid \mathbf{c}\right) = P^c\left(\mathbf{c}\right) P^d\left(\mathbf{c}\right) P^a\left(\mathbf{c}\right). \tag{4.16}$$

Using Equation 4.16, the position component of the "RGB update step" is completely defined. A method of storing a tracker's colour histogram, for evaluating P^c, remains to be specified. The next section will thus describe the colour and direction components of the "RGB update step".

4.2.4 RGB Colour and Direction Update

Each newly created hand tracker is assigned an initially empty skin colour histogram. After a "depth update step", the subsequent registered RGB image is used to update the tracker's skin colour histogram. At this time, denoted as t, the nearest registered RGB pixel to the best hand estimate, $\overline{\mu_t}$, is found, denoted as $\mathbf{c}_t$. If the distance between these points is below a threshold, $\|\mathbf{P}\left(\overline{\mu_t} - \mathbf{c}_t\right)\| < \epsilon$, then the tracker's histogram is updated with the hand's skin colour information.

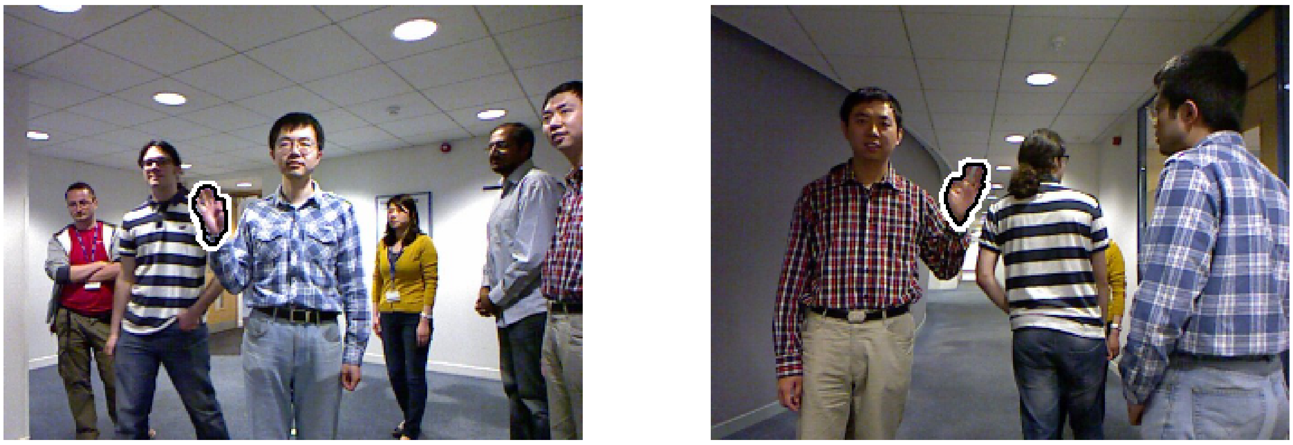


Figure 4.4: Images showing segmented hands from the geodesic distance analysis. The segmentation boundaries are given by the black lines, surrounded by white for visibility. The hue and saturation values of the segmented pixels will be used to update the appropriate tracker's skin colour histogram. This provides the proposed algorithm with illumination invariance and the ability to model various colours of skin.

To isolate skin coloured pixels, all points within a geodesic distance, max_d, to $\mathbf{c}_t$ are analysed. These segmented pixels shall be denoted by the set $\mathcal{H}$. As previously noted in Section 3.2.1, the geodesic distance between two pixels in a registered image is equal to the shortest cumulative distance between them, whilst traversing neighbouring pixels on the same surface [4].

If $\mathbf{c}_t$ lies on side of the hand, the area of pixels in $\mathcal{H}$ will be larger than if $\mathbf{c}_t$ lies on the fingertips. To ensure $\mathcal{H}$ has a constant area every time, max_d is initially set to a small value. The Euclidean distance between the furthest pixels in $\mathcal{H}$ is then compared to the length of a hand, taken as 17 cm. If the furthest pixels are less than 17 cm apart, the geodesic segmentation is repeated with a larger max_d. This process is repeated until the furthest pixels in $\mathcal{H}$ are separated by at least 17 cm. Figure 4.4 shows the results of this segmentation. The hue and saturation values the pixels in $\mathcal{H}$ are used to update the histogram of the appropriate tracker.

As well as the colour measurement, a hand direction measurement can be made from the pixels in $\mathcal{H}$. For this, a new set of 3D pixels, $\mathcal{H}^p$, shall be created, containing the position component of every 6D pixel in $\mathcal{H}$: $\mathcal{H}^p = \{\mathbf{Px} \mid \mathbf{x} \in \mathcal{H}\}$. The normalised, first principal component, $\hat{\mathbf{v}}$, of the points in $\mathcal{H}^p$, will be a vector running down the length of the hand. This vector is an estimate of the hand direction. The direction component of the "RGB update step" uses the Kalman filter equations, presented in Section 4.2.1.

Due to the fact that only the direction component of the hand state is being updated, and not the position component, a slight reformulation of Kalman filter "update step" Equations 4.12 and 4.13, is presented below. Again, matrix $\mathbf{D}$ is used to extract the direction component of the hand state, and $\leftarrow$ represents the assignment operator.

$$\mathbf{d} = \mathbf{D}\overline{\mu_t} + \mathbf{D}\mathbf{K}_t\,\mathbf{D}^T\left(\hat{\mathbf{v}}_t - \mathbf{D}\overline{\mu_t}\right)$$

$$\mathbf{D}\mu_t \leftarrow \frac{\mathbf{d}}{\|\mathbf{d}\|} \tag{4.17}$$

$$\mathbf{D}\mathbf{\Sigma}_t\mathbf{D}^T \leftarrow \mathbf{D}\left(\mathbf{I} - \mathbf{K}_t\right)\overline{\mathbf{\Sigma}_t}\mathbf{D}^T. \tag{4.18}$$

4.2.5 Asynchronous Measurement Update

With the "depth update step" and "RGB update step" defined, one last problem remains to be solved: due to the slower depth-based hand detector, the processing time for depth updates is longer than RGB updates. Assume that from an image at time t_0, a tracker is created from a depth detection. Figure 4.5 illustrates the problem graphically.

Images are generated at intervals of Δ_t, depth updates take Δ_d to process and RGB updates take Δ_{RGB}. Performing a standard depth update at $t_1 + \Delta_d$ will cause RGB update information from images at t_2 and t_3 to be lost, as $\Delta_{RGB} < \Delta_t$ and $\Delta_d > \Delta_t + \Delta_{RGB}$. A method must be presented to incorporate depth detections from past images, without losing subsequent RGB information.

To process the depth detection from Figure 4.5 in a temporally consistent manner, the tracker's state should first be reverted to what is was at t_1. To accomplish this, a history of RGB update information since the last depth update must be maintained. This history should store the image used for the update, its time, and the hand state at this time. In the current example, the most recent depth measurement was at t_0. The history would thus include hand states and images from times: t_1, t_2, t_3.

After reverting the state and performing the "depth update step", the tracker's state would be in an estimated position for time t_1, despite the current time being $t_3 + \Delta_{RGB} < t < t_4$.

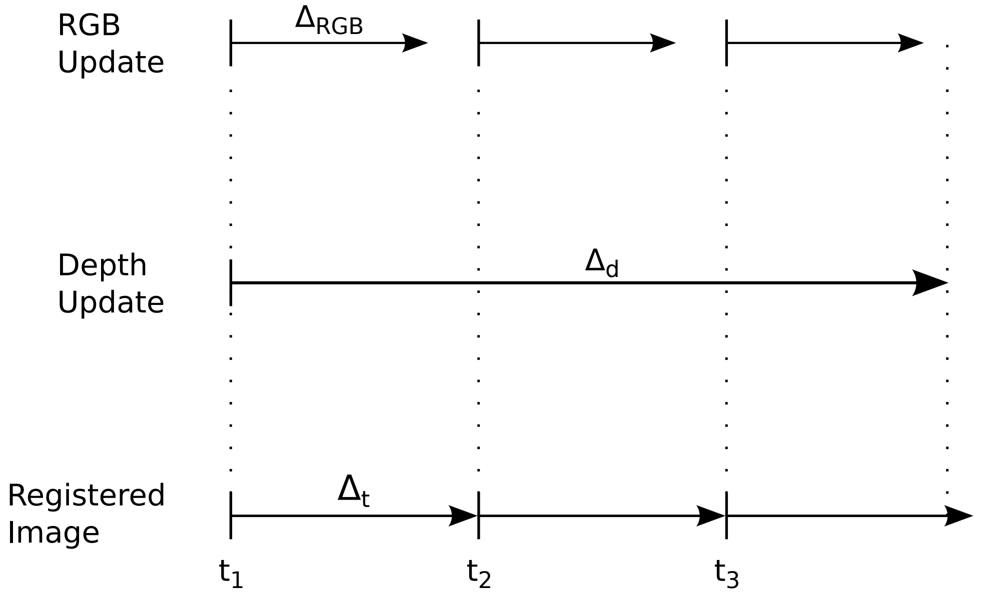


Figure 4.5: Illustration of the asynchronous measurement problem. Images are generated at intervals of Δ_t, depth updates take Δ_d to process and RGB updates take Δ_{RGB}. Performing a standard depth update at $t_1 + \Delta_d$ will cause RGB update information from images at t_2 and t_3 to be lost, as $\Delta_{RGB} < \Delta_t$ and $\Delta_d > \Delta_t + \Delta_{RGB}$.

To re-include RGB information from images at t_2 and t_3, the “RGB update step” should be iteratively re-performed using the relevant images in the history. This would put the tracker’s state in the most accurate possible estimate for the current time.

4.3 Results

All experiments were performed using a Microsoft Kinect depth camera, attached to a static Pioneer P3DX robot base. Images were recorded at a resolution of 640×480 pixels. From these images, a 160×120 resolution depth image was created, as input for the depth-based hand detectors. To ensure that the sensor-fusion tracker’s performance improvement is independent of the underlying depth hand detector, three different methods are compared: the method proposed in Chapter 3, the shape context [61] and the method by Plagemann et al. [1].

As stated in Section 4.2.1, ϵ^d is the distance threshold for assigning a depth-based hand detection to a tracked hand. Based on the maximum distance a hand could travel without being detected, before its corresponding tracker is removed, its value is set to $\epsilon^d = 0.35$ m. Fifty samples are used during the position component of the “RGB update step”, as this is the maximum number that still ensured the algorithm worked at 30 Hz. ϵ is the maximum distance between a sample and its closest registered RGB pixel. Based on the maximum distance the hand can move between two “RGB update steps”, its value is set to $\epsilon = 0.15$ m. The skin colour histogram contains 30 hue bins and 32 saturation bins. This provides a reasonable granularity of hand colour, without suffering from the curse of dimensionality. Finally, the probability scaling parameters, α and β, are set to 0.5 and 750 respectively, as this empirically provided repeatable likelihood values during sampling, for functions P^d and P^a respectively.

Hand tracking performance is compared to the Kalman filter, using a Brownian motion model, for all three of the depth-based hand detectors. Evaluation datasets are the same as used in Chapter 3: Section 4.3.2 presents hand tracking results in nine controlled scenarios and Section 4.3.3 evaluates the method’s performance in three challenging crowded environments. Again, hand tracking performance is evaluated using four gestures in each dataset: a wave, for attracting the robot’s attention; a subtle push, to indicate that interaction has finished; a subtle “follow me” motion; and a raised hand, for stopping the robot. All evaluated datasets have been made publicly available[1].

The same method as in Chapter 3 was used for automatic results generation. 2D ground truth hand positions in input images were manually labelled. For every frame, tracked 3D hand positions are back-projected into 2D, and compared with the ground truth. Ignoring the z (depth) component, a hand is considered correctly tracked if it lies within 0.1 m of the ground truth. This value allows for just enough error to accommodate for depth camera noise and inaccuracies from the ground truth annotation. Using the tri-phase gesture model [134], a gesture can be split into three stages: preparation, stroke and retraction. The preparation and retraction stages involve moving the hand away from, and towards, the rest position. As they contribute little information to the gesture recognition process [82], only stroke phase annotations are considered.

[1]http://www.imperial.ac.uk/hamlyn/eo/gesturedataset

4.3.1 Velocity Estimation Results

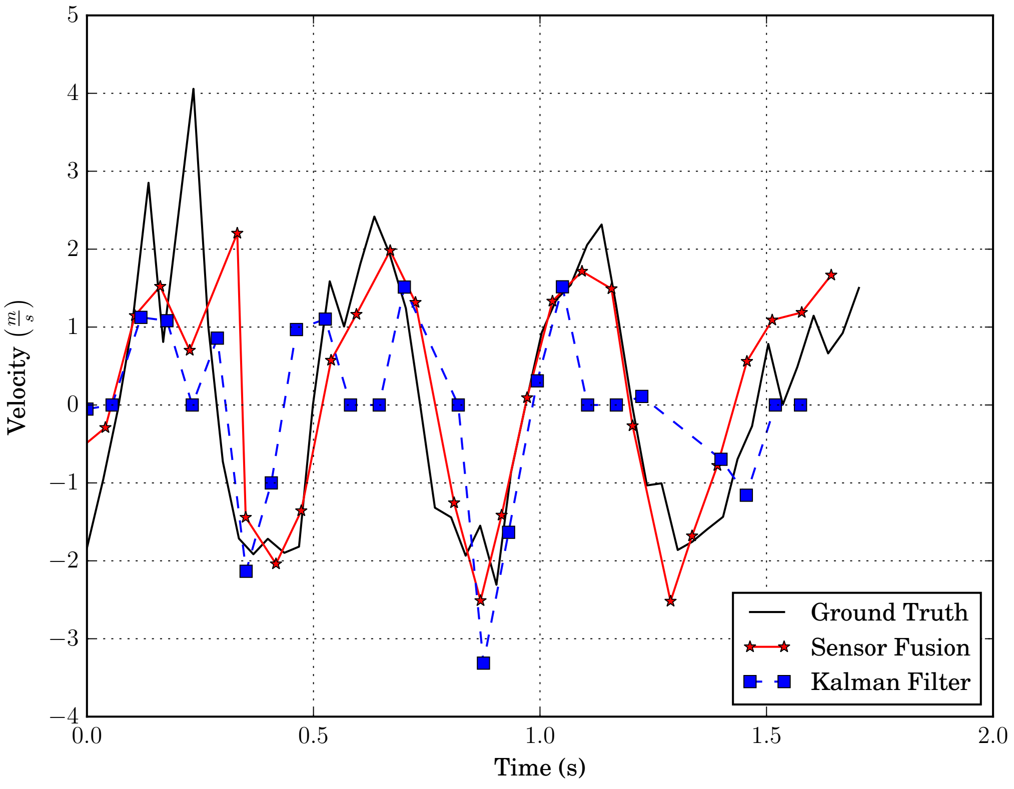
Hand velocity is a common measurement used to classify gestures. In order to evaluate the sensor-fusion hand tracker's improvements to velocity estimation, an experiment was devised where a single subject waved towards the camera. As the wave gesture was orthogonal to the camera, all significant motion occurred in the x-direction.

Using the hand detector proposed in Chapter 3, velocity information was extracted using both the proposed sensor fusion algorithm and a Kalman filter. A hand's velocity was calculated as the first derivative of the tracker's estimated position. Ground truth velocity was generated in the same way, with manual labelling of a hand's position at every frame. Two similar waves were recorded using this experimental setup, with the results displayed in Figure 4.6.

The sensor-fusion hand tracker can be seen to recreate the wave motion better than the Kalman filter. Peaks and troughs are more accurately represented, and less information is lost due to misdetections by the underlying hand detector.

The main difference between the two results, is that occasionally the Kalman filter velocity drops to zero. The Kalman filter outputs the most likely hand position, given by the "predict step", every 30 Hz. If a depth detection fails to happen between two "predict steps", then the output position of the second will be the same as the first, with added Gaussian noise. This can either happen because the depth-based hand detector failed to detect the hand, or because of its lower frame rate. In either case, the similar position will result in a near zero velocity.

In contrast, the "RGB update step" of the proposed sensor-fusion algorithm occurs at 30 Hz. Thus, for each outputted position value, RGB information is used to give a better hand estimate when the depth-based hand detector fails to do so. As can be seen, this results in more accurate velocity reconstruction.



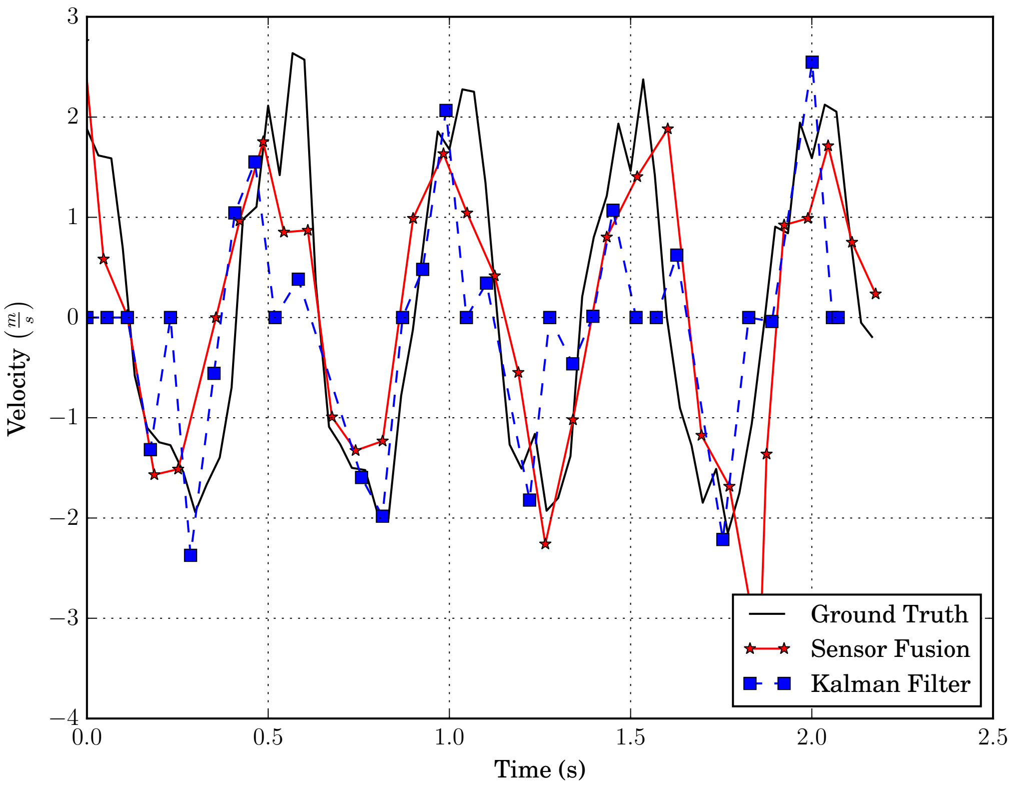


Figure 4.6: Two graphs, each comparing the x-component of a tracked hand's velocity, where a user waves towards an orthogonal camera. Each graph uses the same experimental setup, only with a different recording of the wave. The peaks and troughs of both waves are more accurately represented by the sensor fusion algorithm than the Kalman filter, due to the computationally efficient RGB update step.

4.3.2 Results in Controlled Environments

Hand tracking performance of the sensor-fusion algorithm was compared to the Kalman filter, in nine controlled environments. Each scenario is around a minute long and contains a single subject gesturing towards the camera. The gesturing subject stands at almost every point within the cameras viewing angle, but with a constant distance to the camera and constant gesture speed. The nine scenarios are divided equally between three camera-subject distances, and three ranges of gesture motion. The camera-subject distances evaluated are: 1.5m, 2.0m and 2.5m. The three gesture motion ranges are: a large motion range using only waves, a medium motion range using half waves and half subtle gestures, and a small motion range using only subtle gestures.

During each of the nine scenarios, the subject stands at four different angles to the camera, performing four gestures at each. As the target application of the hand tracker is HRI, it is only tested on subjects gesturing towards the robot. For applications other than HRI, hand tracking performance under large subject rotation can be improved by retraining the hand detector to include these situations.

Hand tracking results for each of these categories is displayed in Table 4.1. Quoted figures are the percentage of correctly tracked hands. **S** denotes the use of the sensor fusion algorithm, whilst **K** denotes the use of a Kalman filter.

Sensor-fusion hand tracking performance is better than the Kalman filter for nearly all scenarios. This increase is seen for all evaluated depth-based hand detectors. The greatest improvements are seen for gestures with a larger range of motion. If the range of gesture motion is small, even if the depth-based hand detector fails, the hand does not move much between frames. Thus, when using the Kalman filter, a previous hand detection may be a “correct” hand position for a number of frames. With a large range of motion, misdetections in the depth-based hand detector are more obvious. The sensor-fusion algorithm’s efficient “RGB update step” gives a good position estimate of the hand, increasing its tracking accuracy.

Table 4.1: Table showing the percentage of correctly tracked hands, for the sensor-fusion algorithm (**S**) and Kalman filter (**K**), in nine controlled scenarios. Scenarios are divided into three distance and three gesture motion range categories, each using a wide range of gesture angles. Results are shown for three different depth-based hand detectors, with the best results for each scenario shown in bold. The average percentage increase of correctly tracked hands, when using the sensor-fusion algorithm, is shown in the bottom row.

	Large Motion			**Medium Motion**			**Small Motion**		
	1.5m	**2.0m**	**2.5m**	**1.5m**	**2.0m**	**2.5m**	**1.5m**	**2.0m**	**2.5m**
				McKeague					
S	79.7	**70.2**	**62.2**	76.9	**68.8**	66.8	**76.5**	**65.9**	**56.0**
K	66.9	53.3	48.7	**77.0**	63.4	**67.4**	76.2	65.1	44.9
				Shape Context					
S	**79.8**	68.0	48.8	74.7	60.9	47.6	71.7	52.5	21.6
K	67.6	50.7	36.3	64.4	57.1	47.8	59.9	48.2	23.7
				Plagemann					
S	77.2	64.9	36.9	27.5	36.5	18.6	18.5	39.6	11.1
K	41.6	41.7	29.0	23.4	21.1	10.4	10.2	26.1	13.2
Average Increase	20.2	19.1	11.3	4.8	8.2	2.5	6.8	6.2	2.3

4.3.3 Results in Crowded Environments

Crowded and dynamic environments are extremely challenging for hand trackers. To assess the sensor-fusion algorithm's practical value, its performance was evaluated in three such scenarios. Each dataset is over a minute long, and contains one or two simultaneous people from a crowd gesturing towards the robot. Each dataset takes place in a different location with different lighting conditions, and has male and female gesturing subjects with varying skin colours.

Figure 4.7 displays several results from the crowded environments, using the sensor fusion tracker and the depth-based hand detector from Chapter 3. They highlight situations that would cause the most problems for traditional RGB hand trackers. 4.7a shows correct hand tracking from a subject with darker skin colour. 4.7b and 4.7c illustrate the algorithm's invariance to different clothing styles. A lot of specular reflection can be observed in 4.7d, further justifying the importance of the algorithm's learned skin colour histogram.

Quantitative hand tracking results for the crowded environments are displayed in Table 4.2. As before, quoted figures are the percentage of correctly tracked hands, for the sensor-fusion

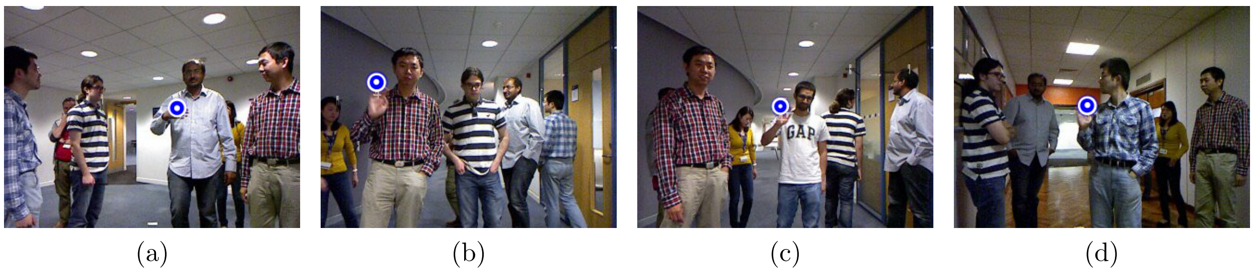


Figure 4.7: Example results showing the hand tracking performance of the sensor-fusion algorithm in a range of crowded environments. Tracked hands are displayed with a white and blue circle. Tracking is shown to work well despite different subject of different skin colour (a), clothing styles (b)(c), and illumination conditions (d). Figure similar to the published paper on this work [6].

algorithm (**S**) and the Kalman filter (**K**), using three depth-based hand detectors.

Again, the performance of the sensor-fusion algorithm surpasses the Kalman filter, for all but one of the results. Generally, the biggest increases are achieved when the underlying hand detector doesn't perform well. In these cases, the sensor fusion algorithm uses RGB information to compensate for the misdetections by the depth hand detector, thus maintaining a higher tracking accuracy than the Kalman filter.

Figure 4.8 shows sensor-fusion hand tracking results, when using the depth-based hand detector from Chapter 3. Tracked hands are contained inside a three circles of expanding size, coloured red, green and blue. The red circle's radius is the first standard deviation of the hand state's x position component. Green and blue are the second and third standard deviations respectively. Monte-Carlo samples from Section 4.2.2 will lie within these boundaries. The reliable results in such a crowded environment demonstrate the robustness of the proposed algorithm.

4.4 Conclusions

The contribution of this chapter has been an RGB-D sensor fusion algorithm for tracking hands in crowded environments. The algorithm solves two problems with current hand detection methods. The first is misdetections caused by occlusions, sensor noise, and the highly articulate nature of the hand. These problems are exacerbated in crowded and dynamic environments.

Table 4.2: Table showing the percentage of correctly tracked hands, for the sensor fusion algorithm (**S**) and the Kalman filter (**K**) in three crowded environments. The average percentage increase, when using the sensor-fusion algorithm, is shown in the bottom row. Each scenario has gesturing subjects with varying skin colour, different styles of clothing, and varying illumination conditions.

	Scenario 1	**Scenario 2**	**Scenario 3**
		McKeague	
S	88.4	78.8	89.2
K	87.4	76.7	90.2
		Shape Context	
S	80.4	63.0	65.3
K	70.5	57.1	53.9
		Plagemann	
S	58.0	44.6	18.5
K	34.3	34.0	14.2
Average Increase	11.5	6.2	4.9

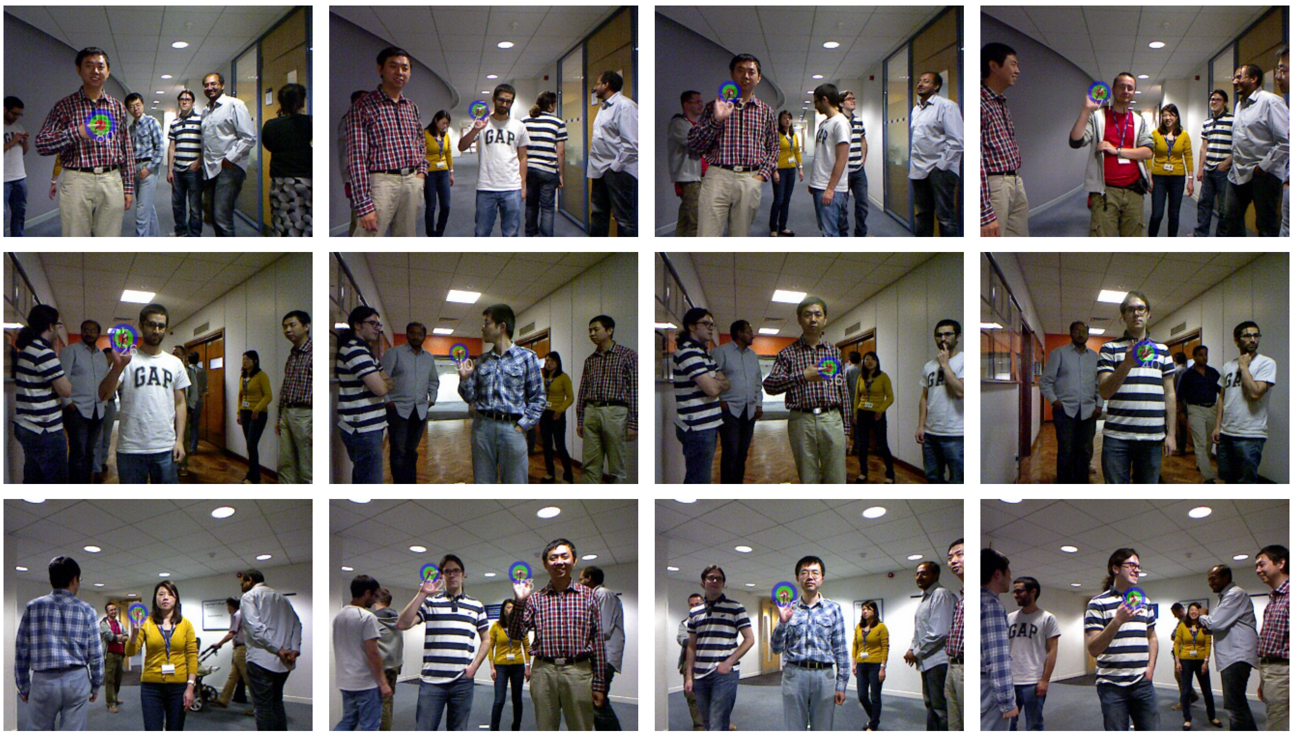


Figure 4.8: Hand tracking results of the sensor-fusion algorithm in three crowded environments. Tracked hands are centred around three circles of expanding size, coloured red, green and blue. Each circle corresponds to a standard deviation of the hand state's x component. Monte-Carlo samples, used to update the hand position in the "RGB update step", will lie within these boundaries.

The second problem the algorithm solves is the computational expense of depth hand detectors. The sensor fusion algorithm gives accurate hand position estimates at a 30 Hz rate.

Three main novelties of the proposed method were highlighted. A Monte-Carlo RGB update process was devised to compensate for the large number of false positives during traditional skin colour detection. A method of online skin colour learning was proposed, so that the algorithm works with a wide range of skin colours, clothing styles and illumination conditions. Finally an asynchronous measurement update step was developed, to integrate depth and RGB updates at different computational speeds into the tracking algorithm.

Tracking performance was evaluated in a large number of controlled scenarios, and several crowded and dynamic environments. All evaluated datasets have been made publicly available. The improved performance of the algorithm over standard Kalman filter tracking was demonstrated, for three different depth-based hand detectors. Using a gesturing subject, the algorithm's improved velocity estimation was also shown. This is a result of the efficient way in which RGB information is used to update hand position.

Hand tracking is a vital component of a HRI system. It is used in many fields, such as gesture recognition. Software was created for this thesis that combines the presented hand tracking method, with the hand body association method of Chapter 3 and the gaze estimation method of Chapter 5. In this way information about the people in the scene, their hand movement, and their gaze direction can be obtained. This framework could be extended to include information about gestures, recognition of objects being pointed at, or to facilitate interaction between robot arms and human hands.

Chapter 5

Gaze Estimation using a Long-Range Infrared Eye Tracker with Sensor Fusion

5.1 Introduction

For effective human-robot interaction (HRI), knowledge of both a person's position and gaze direction is important. A person's precise gaze direction gives a good indication of potential engagement or interest in the robot. This applies to many HRI tasks, and is particularly useful in crowded environments. Despite the general availability of a wide range of remote eye tracking devices, there is still a lack of adequately robust gaze tracking platforms for HRI. Existing vision-based approaches have insufficient granularity required for many HRI tasks, whilst existing solutions incorporating infrared (IR) lighting are too restrictive for use on a mobile robot. Thus, to facilitate real-world HRI, the contribution of this chapter is a

The work in this chapter was partially presented in S. McKeague, J. Liu, A. Vicente, and G.-Z. Yang, "Human gaze estimation using a sensor-fusion based long range infrared eye detector on a mobile robot," in *Submission*, Feb. 2015 sensor-fusion gaze estimation method, that gives better granularity than the traditional visual methods, without the constraints of traditional IR gaze estimation approaches.

Whilst people detection is a mature research problem [22], gaze estimation for moving robots has not been studied extensively. As such, much HRI research makes simple assumptions on the orientation of interacting subjects in a scene [57, 139, 140]. In certain applications, under controlled surroundings, gaze can be estimated using head-mounted cameras, static user positions or motion capture data. These methods, some of which are listed in Table 2.8, are unsuitable for natural HRI in real-life environments. Without knowledge of gaze direction, the presence of human-human interaction would cause many HRI methods to fail, particularly in crowded environments. For example, a subject gesturing towards another person in a scene would cause many gesture recognition systems to wrongly assume that the subject is trying to interact with the robot.

There are currently two main methods of gaze detection in literature: visual and infrared illuminated. Visual methods use colour cameras, and typically use a machine learning technique to classify head pose based on pixel information. IR solutions determine gaze by analysing the retinal and corneal reflections produced by arrays of IR LEDs. Visual methods are computationally expensive, because of the classification of numerous features, and they only allow for coarse head pose estimation. With many visual methods, a granularity of around forty five degrees is common [7, 113]. IR-based solutions provide very accurate gaze estimation, but typically have a short working range [110, 112, 114–116] and require person-specific calibration [112, 115, 116, 118]. A comprehensive list of the restrictions placed on various infrared solutions is shown in Table 2.8. As such, very few of these methods are appropriate for typical HRI applications.

In his 1966 book entitled, The Hidden Dimension, the famed anthropologist Edward T. Hall described proxemics as “The interrelated observations and theories of man’s use of space as a specialised elaboration of culture” [141]. Hall segmented the use of interaction space in proxemics into four distance categories. Arranged in order of increasing distance, they are: intimate, personal, social and public distances. Personal distances are those up to 1.2 m and social distances are between 1.2 m and 3.5 m. Most impersonal business takes place at social distances [141]. Similar to other modern depth cameras, the Microsoft Kinect has a maximal range of 0.8–4 m [1]. Ideally, gaze estimates would be obtainable at any distance. However, as the proposed method requires depth information, gaze estimates at public distances are prevented by hardware constraints.

Nevertheless, for many assistive robot tasks, especially in the domains of healthcare and business or in crowded environments, the human-robot interaction space will cover personal and social distances [142]. Accurate gaze estimates of subjects within this space is vital for detecting their attention [143]: whether directed at the robot, a nearby object, or another human. This is important for many tasks. For example, separating human-human interaction from subjects wishing to initiate HRI [117], locating objects that a user is pointing at [55, 144], detecting gestures using the head [119]. Within the human-robot interaction space, a human's gaze angle can be intuitively considered to lie within ±40°; greater values will place the robot outside the human's field-of-view.

Previous mobile robot algorithms for attention detection have relied on motion capture information [117] or been limited to single users [55, 119]. A recent study into how robots should approach and engage human subjects stated: "If gaze recognition is developed to work robustly in a real environment, [results for initiating conversation at social distances] could be greatly improved because the robot could engage in eye contact to decide whether to initiate interaction" [145]. The gaze estimation method proposed in this chapter attempts to overcome previous limitations, and provide robust gaze estimates for mobile robots in real-world environments. With HRI as the target application, the presented method concentrates on estimating subjects' horizontal gaze angle; the problem of estimating vertical gazing direction is not tackled in this Chapter.

Figure 5.1 shows example images, from real-world scenarios, where accurate gaze estimates are needed for effective HRI. The authors are not aware of any current gaze estimation method, capable of operating on a mobile robot, that offers such required granularity at social distances.

[1]`http://msdn.microsoft.com/en-gb/library/hh438998.aspx` - Accessed 13/09/2014

Figure 5.1: Images, from the robot's view, of real-world scenarios, in which a gaze estimate solution must function accurately to be of use for HRI.

To achieve this, the following chapter proposes a calibration-free, gaze estimation solution for mobile robots, by fusing data from an infrared and depth camera. Pupil detection is initially performed using the difference image from on and off-axis LED illumination. Using the resulting eye positions, a depth camera is used to produce a noisy estimate of gaze direction, by calculating the normal vector of a plane fitted to each subject's forehead. A more stable estimate of a subject's gaze direction is obtained from an equation that relates the pixel distance between detected eyes with the 3D position of the person to the camera. Using the depth-based gaze estimate to find the correct root of this equation, gives a robust, real-time, gaze estimate for subjects located between 1.2 m and 2.5 m from the camera.

The proposed gaze estimation method facilitates real-world HRI, by giving finer granularity of gaze direction than traditional vision-based approaches, with a greatly reduced computational cost. The technique is shown to produce effective gaze estimates at much longer ranges than traditional infrared illumination methods, without the constraints of many existing solutions. The proposed method requires no run-time calibration, and no constraints are placed on the position or number of people in the scene. These claims are substantiated experimentally under a wide range of scenarios, with a moving robot and multiple moving people. Performance is evaluated for a comprehensive number of distances and angles to the robot, with the resulting gaze estimation accuracy compared to the current state-of-the-art.

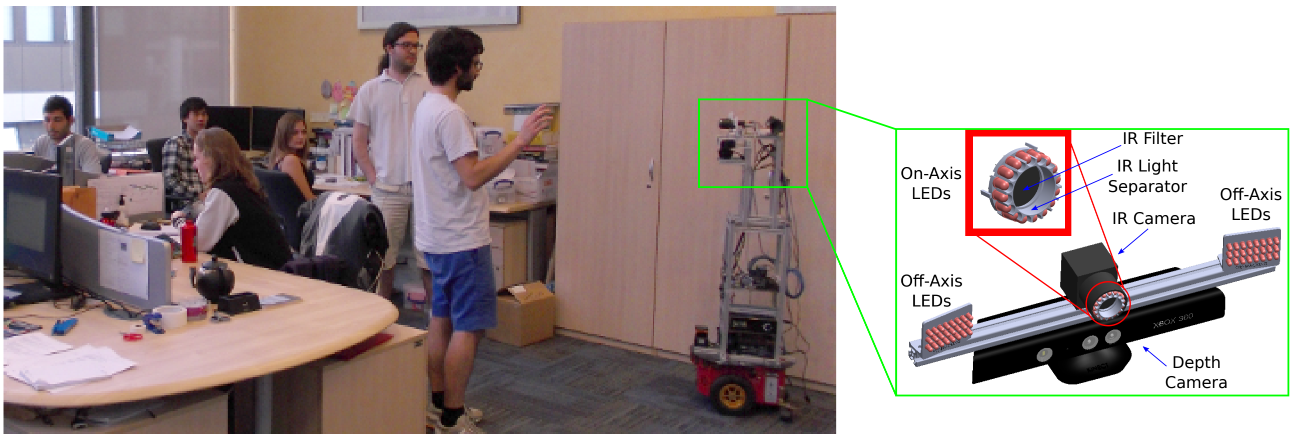


Figure 5.2: Annotated picture of the hardware setup mounted on the mobile robot that is used for all experiments. The robot can use the proposed gaze detection method to know whether the subjects wish to engage in HRI. The system uses on and off-axis IR LEDs to illuminate and darken the pupil respectively. Resulting information from both an IR and a depth camera are fused to give an accurate gaze estimate. An IR-passing filter is used to reduce the effect of visible light, and a plastic separator is used to reduce the effects of glare.

5.2 Method

5.2.1 System Setup

The main prerequisite for the proposed gaze detection system is a robot platform on which it is mounted. In this work, a custom frame, 1.35 m in height, is attached to a Pioneer 3-DX base, as shown in Figure 5.2. All sensors are mounted at the top of the frame, ensuring that subjects' heads are in the robot's field of view at social distances. This fulfils the requirement that, during HRI, eyes within ±40° must be detected.

The proposed system is based on the principal of retinal reflection of infrared light in order to detect eyes. As such an infrared camera, as shown in Figure 5.2, must be used. For this chapter, a Point Grey Flea®3 camera (model number: FL3-U3-13S2M-CS) was employed, with a Fujifilm lens (model number: DV3.4x3.8SA-SA1). In order to provide the illuminating light, a circular array of 16 on-axis infrared LEDs surrounds the camera lens, whilst two rectangular arrays of 24 LEDs each are placed to the side. For observable pupil reflection at large distances, a compromise between LED power and viewing angle must be sought. Empirically, it was observed that at least 60 mW/Sr is required for pupil reflection at HRI distances, whilst illumination half angles of 30° are required to cover all viewing angles. The LEDs employed were manufactured by Rodan (model number: HIRL5015). A circular infrared-passing filter, manufactured by Edmund Optics (model number: 43-949), is used to filter out visible light. Finally, a plastic separator around the lens prevents the LEDs from shining directly back into the camera. Specific hardware model numbers used have been quoted in this section for completeness, though the system should function with alternative hardware of similar specifications.

The circular on-axis LEDs are responsible for illuminating a subject's pupil, whilst the off-axis LEDs give a similar background illumination and leave the pupil dark. Image subtraction between alternate frames will be used to isolate pupils in the image. As such, the employed camera must have a high frame-rate in order to offset the impact that movement within the scene will have on the image subtraction. The Point Grey camera used operates at 60 Hz, giving a 1280×1024 resolution monochrome image in order to view pupils at distances of 3 m. To facilitate the method's sensor fusion requirements, this camera must be correctly registered with a depth camera.

Two pulse signals, with a frequency of the half the camera's frame rate, are used to alternate power to the on and off-axis LEDs. A 50% duty cycle and a phase shift of 180° ensures that in each frame only one set of LEDs is observable, and in the next frame only the opposite LED set is observable. A circuit diagram of the hardware required to accomplish this is shown in Appendix C.

5.2.2 Eye Detection

Figure 5.3 shows a schematic structure of the eye detection phase. Image subtraction of alternate frames is used to isolate pupils within the infrared image. This difference image should be thresholded at an appropriate value to remove the effects of background movement. With the use of the infrared-passing filter, it was found that the algorithm's performance does not rely on an overly accurate threshold value. Using 8-bit pixels, values over 30 successfully removed background noise, whilst only values over 190 started to affect pupil detection. Potential eye clusters are subsequently generated using a connected component algorithm [62].

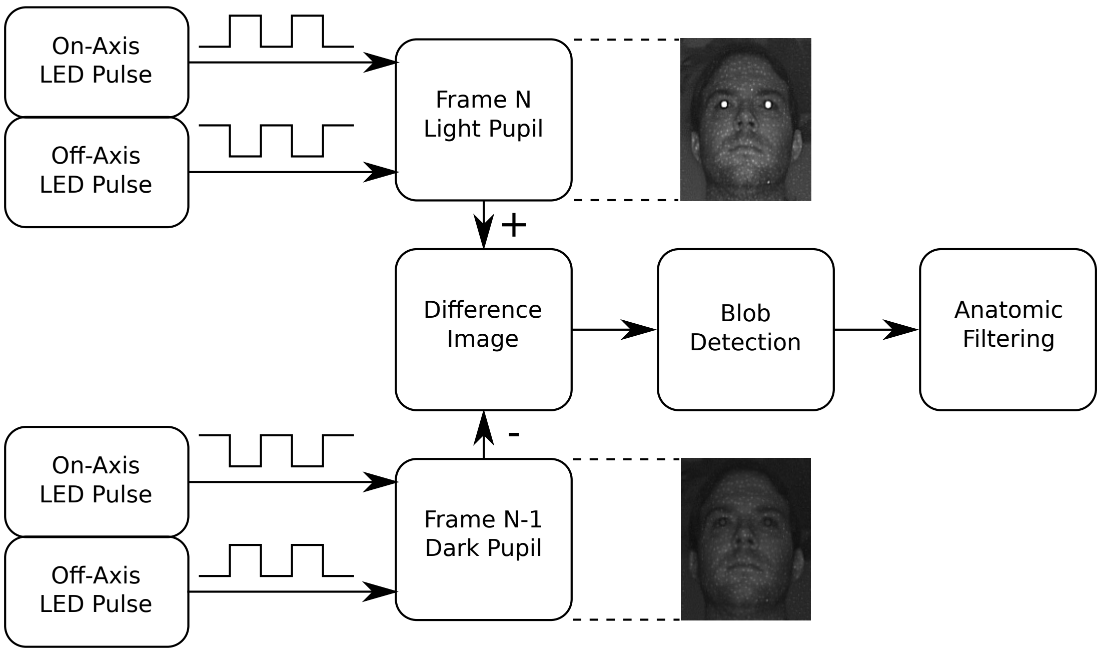
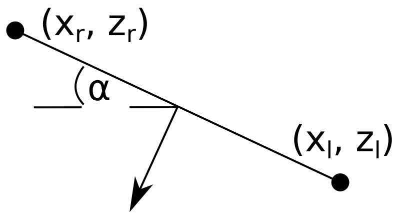


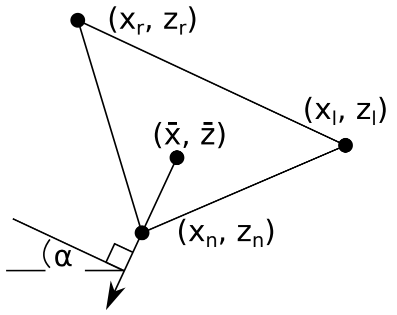
Figure 5.3: System structure diagram of the eye detection phase. The lighting of the on and off-axis LEDs is inverted in alternate frames. A subject's pupil can then be segmented using image subtraction on a consecutive pair of images. Connected component labelling should be performed on the resulting thresholded image to find candidate pixel blobs. Finally, anatomic filtering is used to remove false positives. The on and off-axis face images shown were taken at a distance of 1.5 m with the user looking directly at the robot.

The difference image can contain noise due to specular reflection from shiny surfaces. Filtering of clusters, based on anatomic likelihood, is thus performed. For N eye clusters, two strictly triangular N by N matrices are constructed. The first contains the pixel distance between the means of the clusters. The second contains the rotation angle from one cluster mean to the other. The range of pixel distances between human eyes can be defined based on the working range of the LEDs. The natural range of head angles during HRI can be similarly defined. The pairs of cluster means that satisfy both distance and angular constraints are defined as eye pairs. Because the infrared and depth cameras are registered together, the real-world 3D coordinates of these eye positions can additionally be found from the depth image.

Along with a subject's illuminated pupil, it was found that glasses produce several additional specular reflections. Empirically, it seems possible that a preprocessing stage could be used to filter these extraneous reflections. However, none of the subjects in the results Section 5.3 wore glasses, and the issue is considered to be outside the scope of this Chapter.



(a) Diagram of how the registered 3D right and left eye positions can be directly used to estimate a subject's gaze direction.



(b) Diagram of how the nose, (x_n, z_n), along with the mean of the nose and the registered 3D right and left eye positions, $(\bar{x}, \bar{z})$, can be used estimate a subject's gaze direction.

Figure 5.4: Two methods of estimating gaze direction, α, shown by the arrow, using information from a registered depth image. The gaze angle calculated from Figure (a) is termed the *vector* gaze angle, and that calculated from Figure (b) is termed the *nose* gaze angle. Right and left eyes are displayed as black circles and denoted by (x_r, z_r) and (x_l, z_l) respectively.

Given these eye positions, several techniques can be used to estimate a subject's gaze angle. Three depth-based methods will initially be presented in Section 5.2.3. These return a estimate of a subject's gaze angle, using the 3D eye pair positions and the depth image. A more accurate sensor fusion based method will then be presented in Section 5.2.4. This method estimates gaze angle using the 2D and 3D eye pair positions, information from the infrared camera, and the result of the depth-based gaze estimate.

5.2.3 Depth-Based Gaze Estimation

Assuming an upright person, 3D right and left eye positions from the depth image will be defined as $\mathbf{r}_d = [x_r, y_r, z_r]^T$ and $\mathbf{l}_d = [x_l, y_l, z_l]^T$ respectively. Due to the inherent noise in the depth image, these positions tend to be unreliable. Despite this, a gaze angle estimate can still be calculated using a pair of 3D eye positions, as shown in Figure 5.4a. Defining the vector between the eyes as $\vec{\mathbf{e}} = \frac{\mathbf{r}_d - \mathbf{l}_d}{\|\mathbf{r}_d - \mathbf{l}_d\|}$, this *vector* gaze angle estimate, α_v, can be calculated as:

$$\alpha_v = \tan^{-1}\left(\frac{\vec{\mathbf{e}}_z}{\vec{\mathbf{e}}_x}\right). \tag{5.1}$$

To mitigate the noise of the *vector* method, the nose of a subject could be used to give a more robust gaze angle, as shown in Figure 5.4b. This value will be denoted as the *nose* gaze angle. As a subject's nose will always lie in between their eyes and at no more than 6 cm below them. All points in the registered depth image satisfying these conditions can thus be isolated. The nose can be found from the point in this set with the furthest perpendicular distance from the eye vector.

For each point, $\mathbf{p}$, in this isolated set, a vector from the left eye is defined as $\overrightarrow{\mathbf{p}} = \mathbf{p} - \mathbf{l}_d$. The perpendicular distance between this and the eye vector is given by: $d = \|\overrightarrow{\mathbf{p}} - \left(\overrightarrow{\mathbf{p}} \cdot \overrightarrow{\mathbf{e}}\right)\overrightarrow{\mathbf{e}}\|$. The point, $\mathbf{p}$ that maximises d is defined as the subject's nose, $\mathbf{n}$.

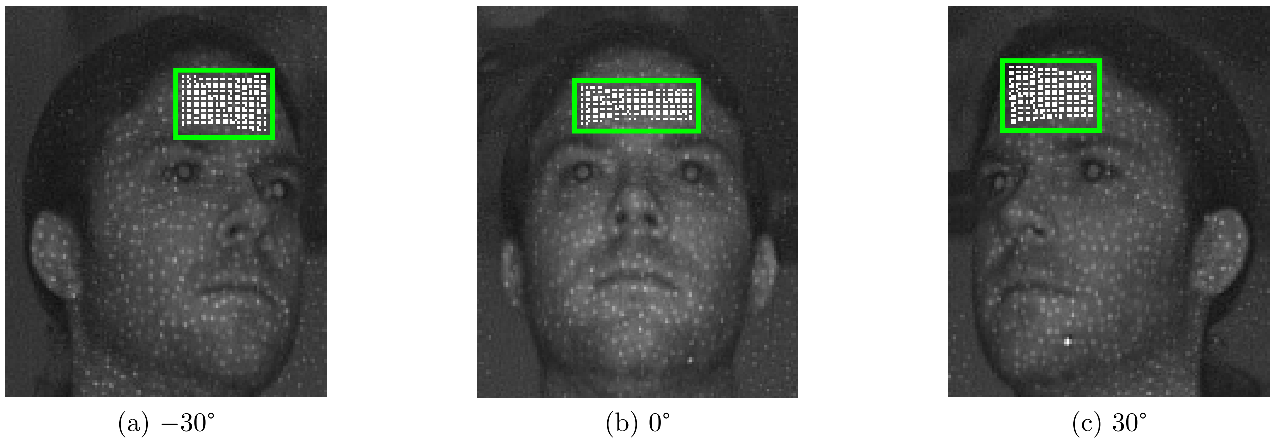
Knowing the position of the subject's eyes and nose, their mid-point can easily be found, $\mathbf{m} = \frac{\mathbf{r}_d + \mathbf{l}_d + \mathbf{n}}{3}$. Gaze direction can be calculated from the vector running through the nose from this mid-point, $\overrightarrow{\mathbf{n}} = \mathbf{n} - \mathbf{m}$. The *nose* gaze angle can be calculated as:

$$\alpha_n = \tan^{-1}\left(\frac{\overrightarrow{\mathbf{n}}_x}{\overrightarrow{\mathbf{n}}_z}\right). \tag{5.2}$$

The benefits of using more points to reduce depth image noise will be demonstrated through experimentation in Section 5.3. Thus, a final method of detecting gaze direction using the depth image was devised, using as many points as possible. The forehead of a subject can be well approximated as a plane that lies perpendicular to the gaze direction. Being the largest planar region of the face, analysis of the forehead can be used to robustly detect gaze direction.

The planar region of the forehead can be defined as the area between both eyes, starting 3 cm above them and extending to no more than 8 cm. All points in the registered depth image that satisfy these conditions can be easily extracted, as shown in Figure 5.5. Fitting a plane to these segmented points and finding its normal vector allows the computation of the subject's gaze direction. The effect of the depth camera's projected IR light can be seen in the image. Because the light is low intensity and mostly static from frame-to-frame, it is not visible when the difference image is calculated and thresholded, as described in Section 5.2.2.

Calculation of the forehead region is most accurate when the forehead is free from occlusion due to hair. Several subjects in the results Section 5.3 have hair that partially occludes the



(a) $-30°$ (b) $0°$ (c) $30°$

Figure 5.5: IR images showing the points comprising the planar region of a subject's forehead, when standing at 3 different angles to the camera. These points are coloured white and lie within the green box. As the white points are 2D pixels corresponding to the 3D forehead points, there is a perspective error for the more distant points at $\pm 30°$. This, along with sensor noise, accounts for the slightly differing bounding box size at 0°. The darker white points on the face are IR points from the depth camera. As they are low intensity and are mostly static, they have no effect on the thresholded difference image, described in Section 5.2.2.

forehead, and it was found that this did not greatly affect the results. The ability of the system to deal with fully occluded foreheads is solely due to the accuracy of the depth camera, and is not evaluated in this Chapter.

To extract the forehead plane, an $N \times 4$ matrix $\mathbf{A}$ shall be created, where N is the number of segmented forehead points. The left-most columns are filled with each point's x, y and z values. The right-most column should be filled with 1s. We shall denote the forehead plane to be found as a vector $\mathbf{x} = [x_0, x_1, x_2, x_3]^T$ that best satisfies $x_0x + x_1y + x_2z + x_3 = 0$, considering each forehead point. Calculating $\mathbf{x}$ involves solving the overdetermined linear system:

$$\begin{aligned} \underset{x}{\text{minimise}} \quad & \|\mathbf{Ax}\| = 0\,, \\ \text{subject to} \quad & \|\mathbf{x}\| = 1\,. \end{aligned} \tag{5.3}$$

This system can be solved using a linear least squares approach. Singular value decomposition can give a numerically stable solution the least squares problem. Using this approach, the real matrix $\mathbf{A}$ is factorised into:

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T.$$

The null space of $\mathbf{A}$ is spanned by the last $4 - r$ rows of the 4×4 matrix $\mathbf{V}^T$, where r denotes the rank of matrix $\mathbf{A}$. The value of $\mathbf{x}$ that best solves Equation 5.3 is given by the right-singular vector of $\mathbf{A}$, which has the smallest singular value.

The normal vector of the forehead plane is used to give an estimate of the gaze direction. This estimate will be termed the *forehead* gaze angle, and can be calculated as:

$$\alpha_f = \tan^{-1}\left(\frac{x_0}{x_2}\right). \tag{5.4}$$

5.2.4 Sensor Fusion Gaze Estimation

To offset the noise in the depth image, especially during camera or subject motion, the precision of the eye positions in the infrared image can be exploited to give the following *sensor fusion* gaze estimate. The mid-point of the eye vector, $\mathbf{m}_d = \frac{\mathbf{r}_d + \mathbf{l}_d}{2}$, will be used due to its stability, compared to $\mathbf{r}_d$ or $\mathbf{l}_d$ individually. The x-pixel component of the right eye relative to the midpoint of the infrared image shall be i_r, with the corresponding left eye being i_l. Knowing the focal length, f, of the infrared pinhole camera model, Figure 5.6 illustrates how gaze angle, α, can be derived using this *sensor fusion* method.

From Figure 5.6, the relationship between i_r and i_l with the 3D right and left eyes, $\mathbf{r}_d$ and $\mathbf{l}_d$, can be given as:

$$\begin{aligned} -i_r &= \frac{x_r}{z_r} f, \\ -i_l &= \frac{x_l}{z_l} f. \end{aligned} \tag{5.5}$$

With the previously defined stable estimate of $\mathbf{m}_d = [x_m, y_m, z_m]^T$, the following relationship

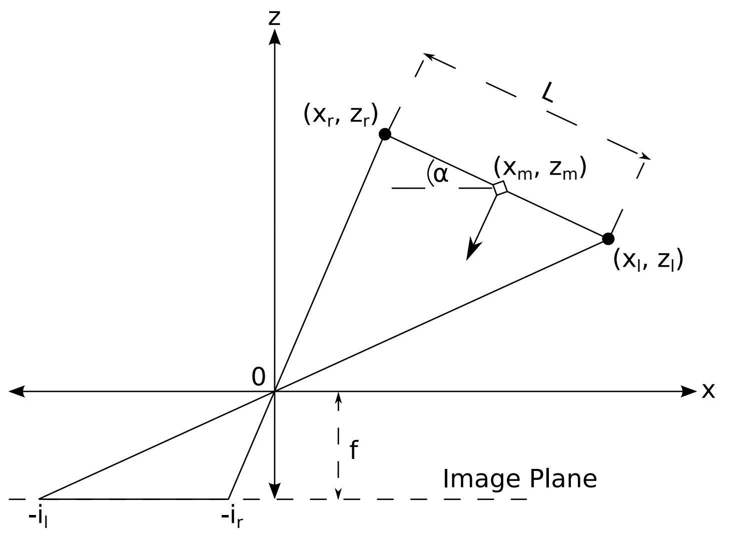


Figure 5.6: Illustration of how to derive gaze angle by fusing information from both the depth image and an IR pinhole camera. Right and left eyes, (x_r, z_r) and (x_l, z_l), are displayed as black circles with their midpoint, (x_m, z_m), shown as a white diamond. Interpupillary distance is denoted as L, camera focal length as f, and $-i_r$ and $-i_l$ denote the x-coordinates of the right and left eyes relative to the centre of the image plane. The sensor fusion algorithm aims to robustly calculate gaze angle, α, shown by the arrow between the eyes.

between the interpupillary distance, L, and the gaze angle, α, can be defined:

$$\begin{aligned} x_r &= x_m - \frac{L}{2}\cos(\alpha), & z_r &= z_m + \frac{L}{2}\sin(\alpha), \\ x_l &= x_m + \frac{L}{2}\cos(\alpha), & z_l &= z_m - \frac{L}{2}\sin(\alpha). \end{aligned} \tag{5.6}$$

Substituting Equations 5.6 into Equation 5.5 allows us to draw a relationship between pixel and world eye coordinates:

$$\begin{aligned} i_l - i_r &= \frac{x_m - \frac{L}{2}\cos(\alpha)}{z_m + \frac{L}{2}\sin(\alpha)} f - \frac{x_m + \frac{L}{2}\cos(\alpha)}{z_m - \frac{L}{2}\sin(\alpha)} f, \\ &= 4fL \frac{z_m \cos(\alpha) + x_m \sin(\alpha)}{L^2 \sin^2(\alpha) - 4z_m^2}. \end{aligned} \tag{5.7}$$

It can be noted that Equation 5.7 is not algebraically solvable for gaze angle α. One possible solution would be to simplify the equation, using the fact that under all normal HRI distances $z_m \gg L$. Defining $D_p = i_l - i_r$, Equation 5.7 can be rewritten to give an unsigned estimate of the gaze angle, which shall be denoted as α_u:

$$D_p = \frac{x_m - \frac{L}{2}\cos(\alpha_u) - x_m - \frac{L}{2}\cos(\alpha_u)}{z_m} f,$$
$$\alpha_u = \cos^{-1}\left(\frac{D_p z_m}{fL}\right). \tag{5.8}$$

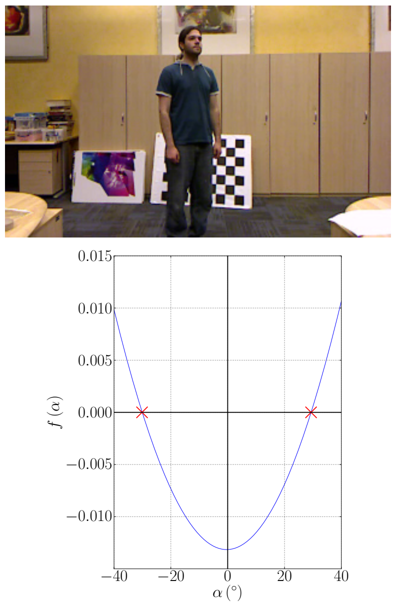
The smaller a subject's $|x_m|$, the more accurate α_u will be from Equation 5.8. During intermediate testing it was found that for $|x_m| > 0.5$, Equation 5.8 failed to give valid α_u values, due to D_p decreasing to the point where $D_p z_m \ll fL$. As this gives values close to 0, and due to α_u being an unsigned value, a different solution to Equation 5.7 must be found.

Rewriting Equation 5.7 as a function, $f(\alpha)$, enables the calculation of α as the roots of the f. Due to the fact that f is differentiable, many numerical analysis methods can be applied to find its roots. The Newton-Raphson method is an iterative root finding algorithm, and was chosen for this purpose due to its fast convergence.

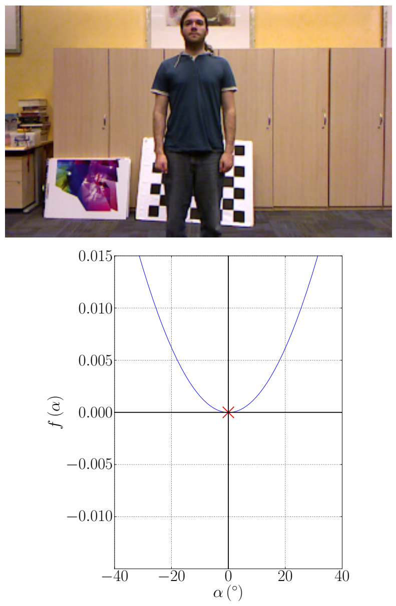
$$\begin{aligned} f(\alpha) &= \frac{z_m\cos(\alpha) + x_m\sin(\alpha)}{L^2\sin^2(\alpha) - 4z_m^2} - \frac{D_p}{4fL}, \\ \frac{\mathrm{d}(f(\alpha))}{\mathrm{d}\alpha} &= 4fL\frac{\sin(\alpha)^3 z_m L^2 - 2\sin(\alpha) z_m L^2}{(\sin(\alpha)L - 2z_m)^2(\sin(\alpha)L + 2z_m)^2} \\ &\quad + 4fL\frac{-\cos(\alpha)\sin(\alpha)^2 x_m L^2 + 4\sin(\alpha) z_m^3 - 4\cos(\alpha) x_m z_m^2}{(\sin(\alpha)L - 2z_m)^2(\sin(\alpha)L + 2z_m)^2}. \end{aligned} \tag{5.9}$$

To better understand $f(\alpha)$, Figure 5.7 shows how it is affected by differing subject position and gaze angle. If a subject is facing directly towards the camera, $f(\alpha)$ should have only one root, at 0°. For non-zero gaze angles, $f(\alpha)$ should have two roots. As demonstrated in Figure 5.7f, a subject with a positive gaze angle results in the positive-most root being equal to its value. Conversely, as shown in Figure 5.7d, a subject with a negative gaze angle results in the negative-most root being equal to its value. In both cases, the other root is unstable. The result of this *sensor fusion* method will be the value of the stable root, which will be termed α_s.

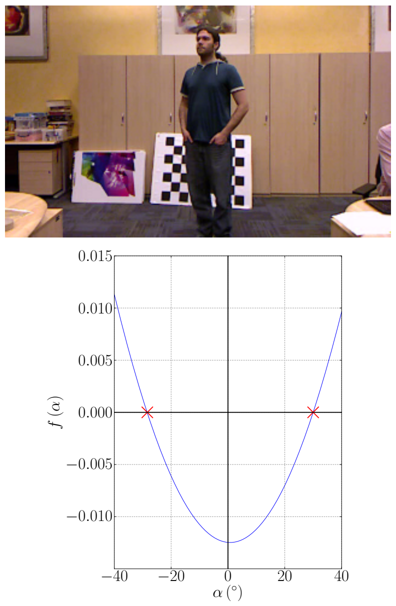
An important limitation of the Newton-Raphson method is that it only returns one root. If the



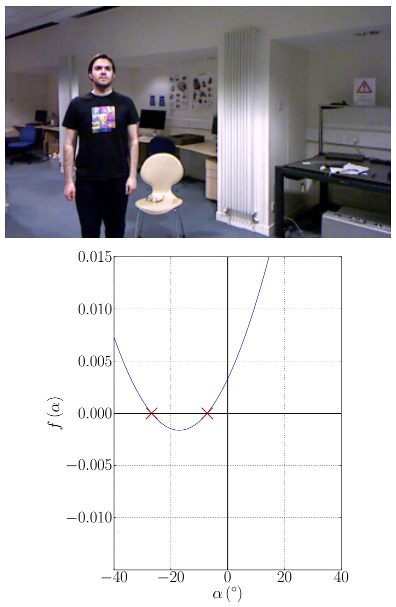
(a) −30° Centred



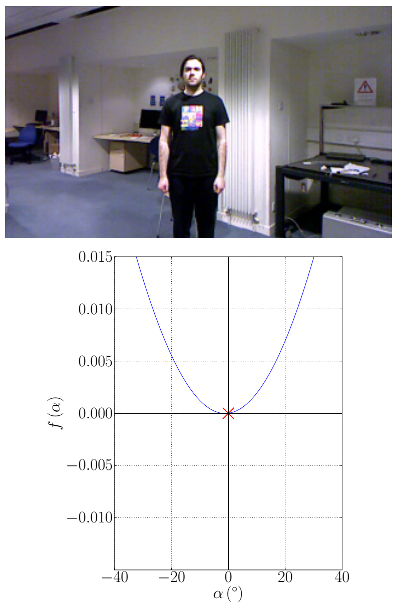
(b) 0° Centred



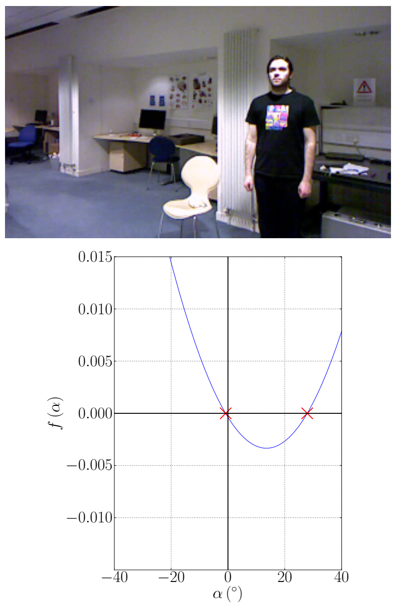
(c) 30° Centred



(d) −30° Camera Left



(e) 0° Centred



(f) 30° Camera Right

Figure 5.7: Six graphs of $f(\alpha)$, with roots marked in red, and the corresponding RGB image of the subject that generated the graph. (a) to (c) show graphs for three different gaze angles with a subject positioned in the centre of the camera. (d) to (f) show graphs for the same angles, but with a subject standing at various locations in the image. For a gaze angle of exactly 0°, $f(\alpha)$ should have only one root, at 0°. For gaze angles of ±30°, $f(\alpha)$ should have two roots. At 30°, the positive-most root of $f(\alpha)$ is a stable value of 30°. At −30°, the negative-most root is a stable value of −30°. The other root in both cases is unstable.

function is known in advance, the returned root can be predicted from the supplied starting point. However, as Figure 5.7 shows, changes to $f(\alpha)$ for a moving subject would mean that naïve application of the Newton-Raphson method would, somewhat randomly, return the unstable, incorrect root. A method must thus be devised to select which of the two roots is returned by the Newton-Raphson method.

In each of the graphs in Figure 5.7, it can be seen that, for non-zero gaze angles, both roots are separated by the function minimum. This function minimum is equal to the sole root of $\frac{\mathrm{d}(f(\alpha))}{\mathrm{d}\alpha}$ within ±40°. Calculating the second differential of $f(\alpha)$ allows us to use the Newton-Raphson method to find this root.

Knowing the functional minimum, which shall be defined as f_{min}, we can then alter the domain of the original Newton-Raphson method to limit returned roots. If the positive-most root is required, the input domain should be set to $[f_{min}, 45°]$. If the negative-most root is required, the input domain should be $[-45°, f_{min}]$. In each case, the initial guess should be set to the the mean of the domain bounds.

Now that the returned Newton-Raphson root can be selected in advance, the question remains: how is it determined whether to return the positive-most or the negative-most root? Conveniently, each of the gaze angles, α_v, α_n and α_f defined in Section 5.2.3 return a signed estimate for gaze angle, albeit a noisy one. α_f will be shown in Section 5.3 to be the most accurate of these methods. Thus, if $\alpha_f > 0$, then the positive-most root should be returned from the Newton-Raphson algorithm, otherwise the negative-most root should be returned. In this way, the result of α_f is used to select the stable root of $f(\alpha)$, α_s; the gaze estimate of the *sensor fusion* method.

To add increased robustness to the α_f values from Equation 5.4, we can low pass filter the results over a very small time window. Transient periods where α_f has an incorrect sign will thus not result in the wrong Newton-Raphson root being selected. Thus, for a moving window, W_T, at time T, α_f is adjusted as follows:

$$\alpha'_f = \frac{\sum_{t \in W_T} \alpha_{f,t}}{|W_T|} \tag{5.10}$$

The only remaining task for calculating α_s is to define the interpupillary distance, L, first introduced in Equation 5.6. Due to the biological importance of interpupillary distance, many studies have been done to establish its average value [146]. Using these studies, we take an average value of L to be 67 mm.

5.3 Results

The depth camera used in all experiments was a Microsoft Kinect, with resolution 640×480 pixels, running at 30 Hz, and mounted on the robot shown in Figure 5.2.

As stated in Section 5.2.2, when detecting blobs in the infrared difference image, most pupil values were around 190 and most noise values were below 30. Thus, values above a threshold of 150 were used to detect blobs, as a trade off between pupil detection accuracy and noise removal. During anatomical filtering, cluster pairs were filtered as pupils, based on their angular and distance separation. During HRI, it is considered that a subject's head will be tilted horizontally by no more than 30°; any cluster pair that forms a greater angle is thus discarded. Pupil detection can be reliably performed between 1–3 m. Within this working range, pupils are separated by distances of roughly 15–75 pixels; any cluster pair separated by distances outside this range is thus discarded. Finally, as a compromise between suppressing spurious depth-based gaze estimates, and the systems reaction time to a user looking in the opposite direction, a window size of 0.08 seconds was used during low-pass filtering.

The performance of the proposed *sensor fusion* algorithm was evaluated under several different scenarios, with multiple static and dynamic people, using both a static and moving robot. As previously discussed, the *sensor fusion* algorithm uses one of the three depth methods from Section 5.2.3. The most robust depth method will thus first be determined, in Section 5.3.1. Its performance will then be compared to the *sensor fusion* algorithm, in Section 5.3.2. Finally, these results will be presented alongside a state-of-the-art RGB gaze estimation method [7], in Section 5.3.3. Results were generated for a total of forty six recordings, equating to over ten minutes of ground truth annotated experiments.

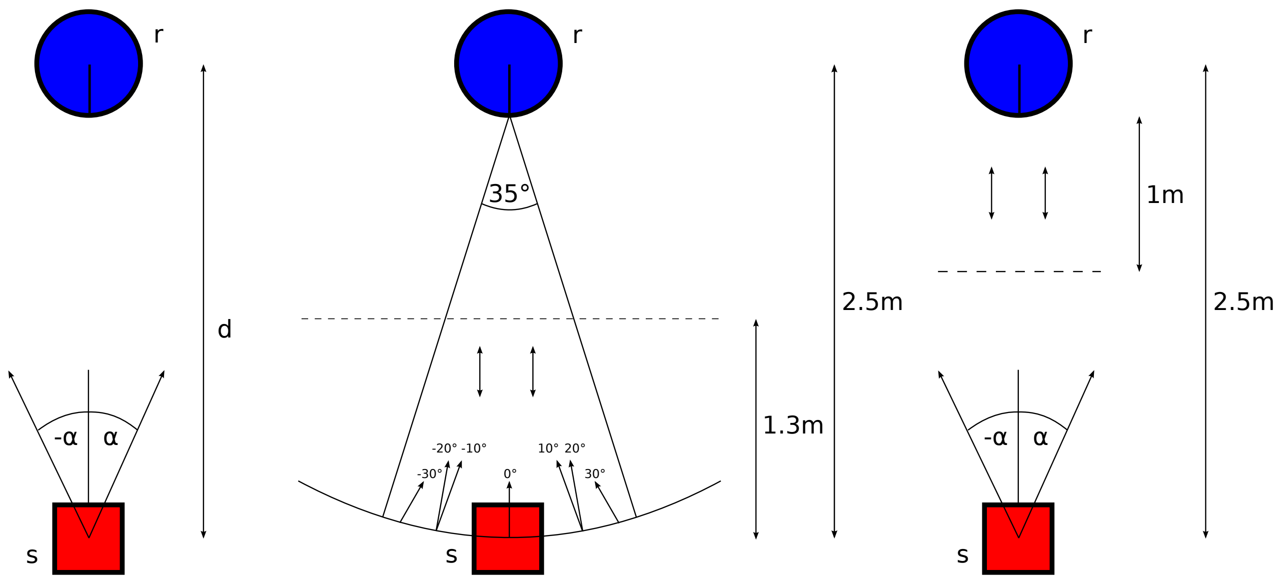
The following experiments can be classified into three different categories (Figure 5.8): those with a static subject, those with moving subjects and those with a moving robot. To evaluate how gaze detection accuracy is affected by the distance of the subject to the camera, a series of recordings were made of a static subject, at a known distance, looking in a particular direction (Figure 5.8a). In each recording, the user stood at one of three distances from the camera: 1.5 m, 2.0 m and 2.5 m. Additionally, nine angles were evaluated in the recordings, ranging from $-40°$ to $40°$, in $10°$ intervals. Figures 5.7a to 5.7c show screenshots of three of the 2.5 m static recordings.

For effective HRI, a gaze detection method will have to work under both people and robot movement. For each of these cases, a scenario was designed to evaluate the effect on gaze detection accuracy. The scenario to test the impact of moving people is shown in Figure 5.8b. For angles of $-30°$ to $30°$, in $10°$ intervals, the person moves forward and backwards, throughout the robot's $35°$ field of view, between 1.2 m and 2.5 m. Figures 5.7d to 5.7f are screenshots of these recordings. This scenario is performed with both one and two simultaneous people. Lastly, Figure 5.8c shows the scenario designed to investigate the effects robot motion. For the same range of angles used in the static scenario, the robot moves forward and backwards between 1.5 m and 2.5 m. The robot and people all moved at a slow walking pace.

5.3.1 Depth-Based Gaze Estimation Results

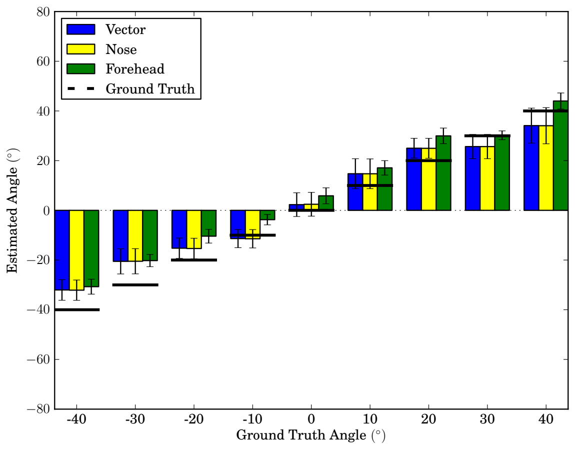
Figure 5.9 is a bar chart of the gaze angles generated by each of the three depth methods, *vector*, *nose* and *forehead*, for the static scenarios (Figure 5.8a). At 1.5 m the three methods are quite comparable. However, as distance increases the standard deviation of the *vector* and *nose* methods increases dramatically, whilst the standard deviation of the *forehead* method remains relatively constant. Additionally, the mean of the *forehead* method is closer to the ground truth at increasing distance.

This pattern is repeated in the results of both the moving person and moving robot scenarios (Figures 5.8b and 5.8c), which can be seen in Figure 5.10. The standard deviations of the *forehead* method in the moving robot experiments are much smaller than the other two methods,

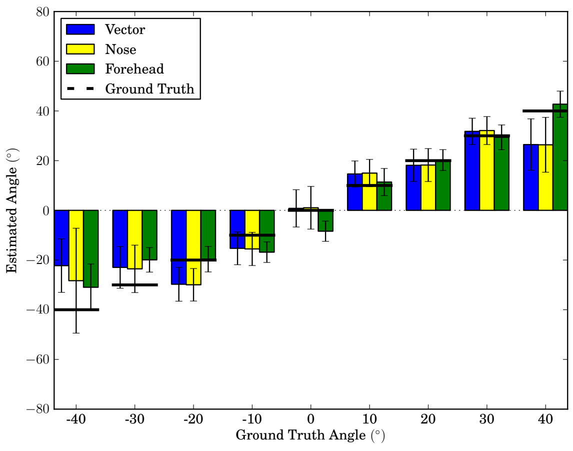


(a) Static Subject (b) Moving Subjects (c) Moving Robot

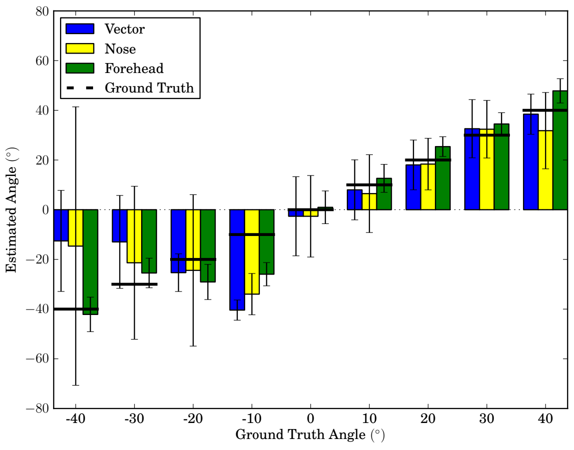
Figure 5.8: Diagrams of the 3 different scenarios used to evaluate gaze estimation accuracy. "r" denotes the robot position, and "s" denotes the subject position. As explained in the text, $\alpha = (-40°, -30°, \ldots, 30°, 40°)$, $d = (1.5\text{ m}, 2.0\text{ m}, 2.5\text{ m})$. (a) evaluates how gaze accuracy is affected by distance and angle to a static subject. (b) tests how accuracy depends on a subject moving throughout the camera's 35° field of view. This scenario is performed with both one and two simultaneous people. (c) evaluates how accuracy is affected by a robot moving towards and away from a subject, using the same gaze angles in (a).



(a) 1.5 m

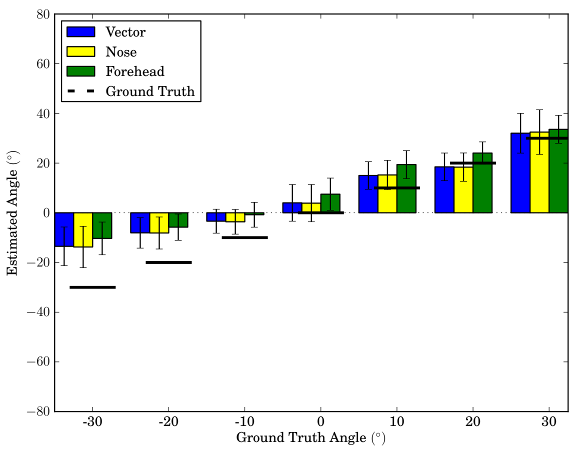


(b) 2.0 m

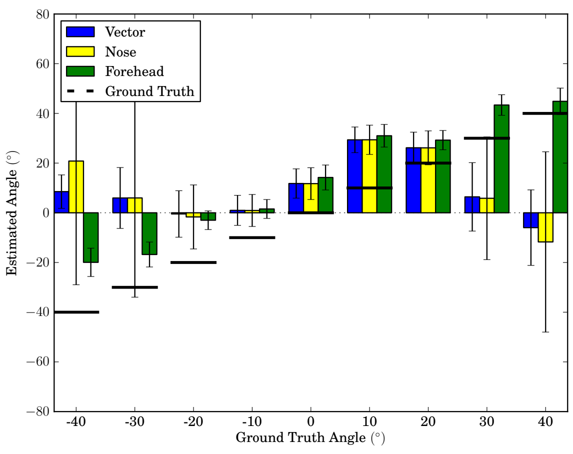


(c) 2.5 m

Figure 5.9: A bar chart of the gaze angle generated by each of the 3 depth methods considered, *vector*, *nose* and *forehead*, for the static scenario distances of 1.5 m, (a), 2.0 m, (b), and 2.5 m, (c). The bar value equates to the mean gaze angle for the particular method and distance-angle combination. The first standard deviation is displayed with an error bar. Especially with greater distances, the *forehead* method has a generally more accurate mean, and a much smaller standard deviation than the other two methods.



(a) Moving Person



(b) Moving Robot

Figure 5.10: A bar chart of the gaze angle generated by each of the 3 depth methods considered, for the single moving person scenario (a), and the moving robot scenario (b). Similar to the static scenario results (Figure 5.9), the mean of the *forehead* method is generally more accurate than the other two methods. The standard deviation of the *forehead* method, especially in the moving robot experiments, is also much smaller.

with more accurate mean values at more extreme angles. In the moving person experiment, the performance difference is less marked. However, all methods perform worse at negative angles than positive angles, with the gaze estimates being underestimated. This seems to be due to a "layering" effect in the Kinect's depth image, where at far distances, depth points tend to clump into layers, perpendicular to the camera plane. This effect is more notable under added movement noise and at negative angles, causing inaccuracies in the depth pixels, and thus the gaze estimates.

5.3.2 Sensor-Fusion Gaze Estimation Results

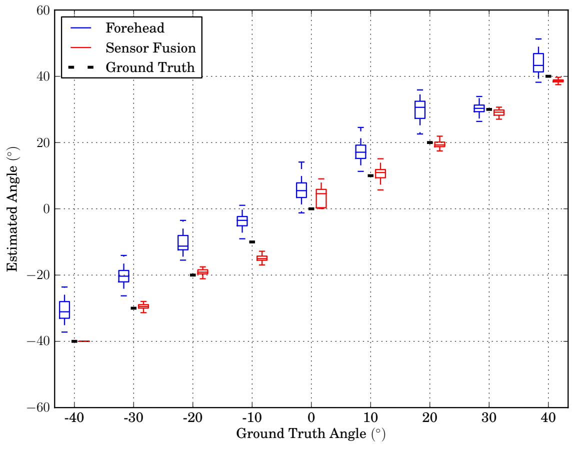
Due to its superior results, the sign of the *forehead* gaze angle was used to select the stable root of the *sensor fusion* method, as described in Section 5.2.4. Its performance was then evaluated using all the scenarios in Figure 5.8. To allow for a detailed results comparison, box plots of the two methods were generated. Results from the three static scenarios are shown in Figure 5.11, whilst the two moving scenarios are shown in Figure 5.12. For each ground truth value, the results from the *forehead* method are shown in blue to the left, whilst the *sensor fusion* results are shown in red to the right. The median gaze estimate at each distance-angle combination is displayed as a line on the box, which extends from the upper to the lower quartiles of the data. Whiskers are also plotted, which show the most extreme data points within 1.5 times the interquartile range.

The *sensor fusion* gaze estimates in the static scenarios (Figure 5.11), are clearly very accurate. The precise median and small interquartile range are present at all tested distances, with performance being visibly better than the *forehead* method.

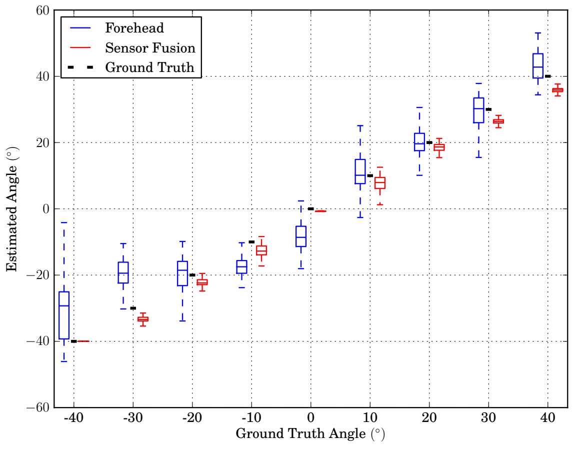
The results in Figure 5.12a show that although a moving subject naturally produces more noise in the depth image, the median of the *sensor fusion* method is never more than around 8° from the ground truth. For most tested angles, this is much more accurate than the *forehead* method, validating the method's robustness under HRI conditions.

Sensor fusion results from the moving robot scenario (Figure 5.12b), for every tested angle except $-10°$, are equally as accurate. The interquartile range is smaller than that of the moving person. This could be explained by the smoother motion of the robot compared to a person. Except at $-10°$, the median value is consistently close to the ground truth, although slight deviances are present at smaller angles. A screenshot for the 0° moving robot scenario, along with a plot of Equation 5.7, is shown in Figure 5.7e. At $\alpha = 0$, the graph has the smallest gradient. Noise in the depth image, from the moving robot and person, tends to translate the graph parallel to the y-axis. This will cause the biggest change in the roots of α for gaze angles near 0°. Despite this, the results are still conclusively better than the *forehead* method, which itself was the best of the three depth methods proposed in Section 5.2.3.

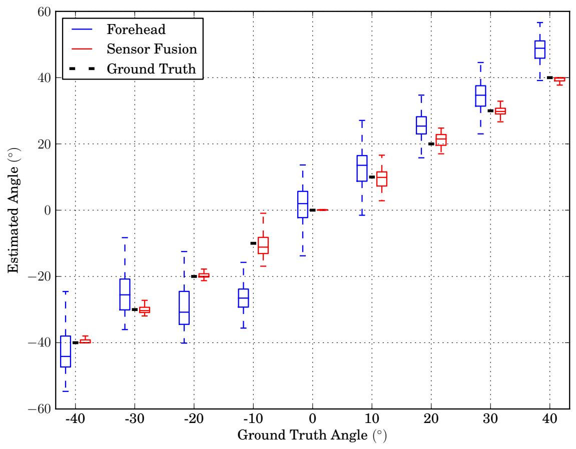
To further investigate the anomalous result for the moving robot experiment at $-10°$, Figure 5.13 is a plot of gaze angle against time for angles of $-10°$ (Figure 5.13a) and $-40°$ (Figure 5.13b). The large variance of the *sensor fusion* method at $-10°$ can be explained by the associated *forehead* gaze angle. When the sign of the *forehead* gaze angle is continuously wrong for longer than the low-pass filtering window size, the wrong root of the *sensor fusion* method will be chosen. This wrong root corresponds to the estimated angles around 15° in Figure 5.13a. Whilst a longer low-pass filtering window could be chosen to improve the results, this is the



(a) 1.5 m

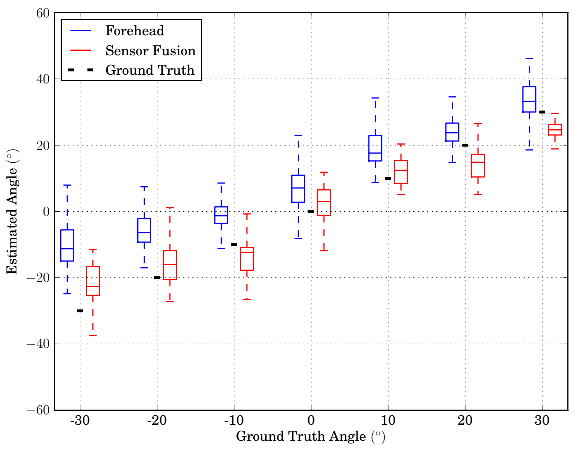


(b) 2.0 m

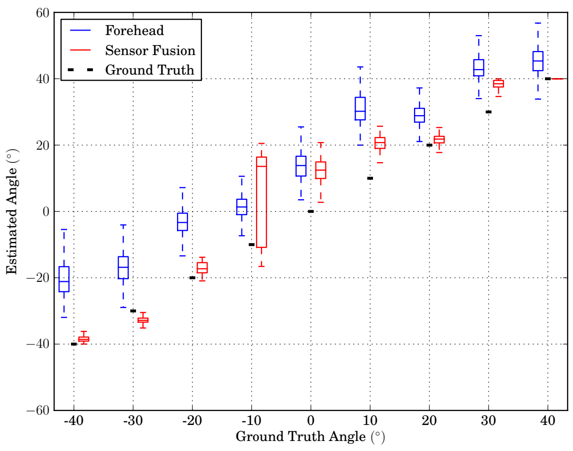


(c) 2.5 m

Figure 5.11: Box plots comparing gaze angles generated for the static scenarios using both the proposed *sensor fusion* algorithm and the *forehead* method. For each discrete ground truth angle, the *forehead* results are shown to the left and the *sensor fusion* results are shown to the right. Boxes extend from the upper and lower quartiles of the data, with the median shown as a line in the box. Whiskers extend to the most extreme data point within 1.5 times the interquartile range. Results for distances of 1.5 m, (a), 2.0 m, (b), and 2.5 m, (c), are shown. At virtually every distance-angle combination the *sensor fusion* algorithm has both a closer median and a smaller interquartile range than the *forehead* method.

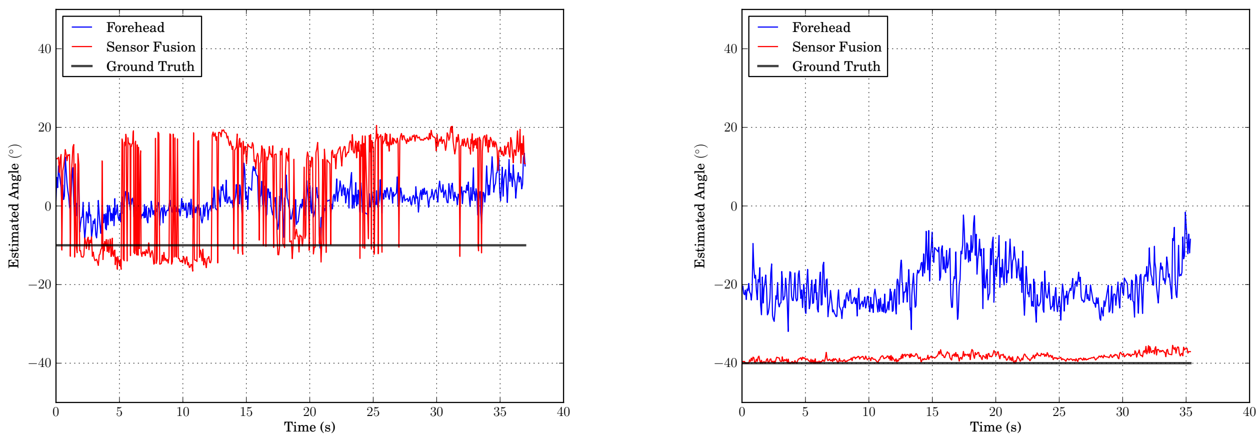


(a) Moving Person



(b) Moving Robot

Figure 5.12: Box plots comparing the *sensor fusion* gaze angle to that generated from the *forehead* method, for the single moving person scenario (a) and the moving robot scenario (b). Under depth camera noise from a moving person (a), both methods exhibit more noise than the results for the static scenario. However, the median of the *sensor fusion* method is closer to the ground truth for most gaze angles, being at most around 8° away. With sensor noise from a moving robot (b), the small interquartile range of the *sensor fusion* method should be noted, along with its consistently more accurate median value than the *forehead* method. The only exception to this is at $-10°$, which can be explained by the sign of the *forehead* method alternating between positive and negative values. This will cause the wrong root of the *sensor fusion* method to be chosen, resulting in a gaze angle of approximately 15° for the given experiment.



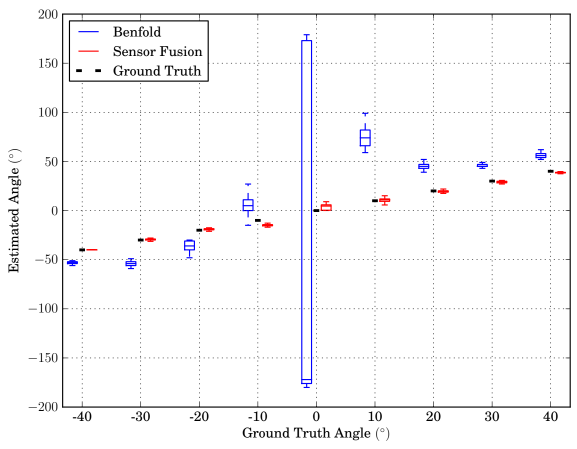
(a) −10° (b) −40°

Figure 5.13: A graph of estimated angle against time for the moving robot scenario at −10° (a) and −40° (b). The *sensor fusion* algorithm's anomalous results at −10° can be explained by the fluctuating sign of the *forehead* method. As its sign is used by the *sensor fusion* algorithm, if it is continuously wrong for longer than the low-pass filter window, then the *sensor fusion* result will also be wrong. This is the only experiment of all 46 to exhibit this problem, and it can be contrasted with the algorithm's accuracy and stability at −40°.

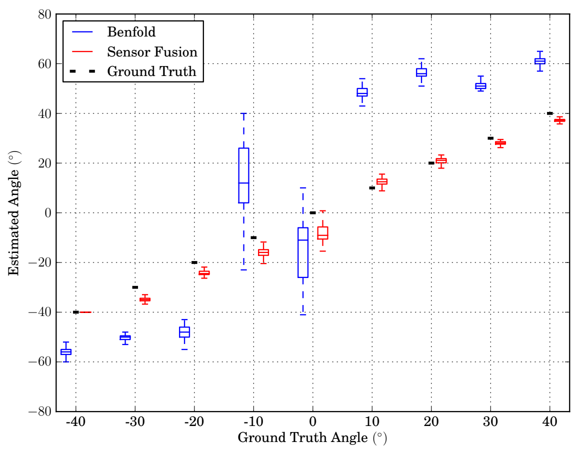
only experiment out of a total of forty six recorded to exhibit this problem. Additionally, these results can be contrasted with those at −40°, where the *sensor fusion* method gives a consistently accurate result using a moving robot. This suggests that the granularity of the system is as small as 1°–2°, even during movement.

5.3.3 Results Comparison with State-of-the-Art

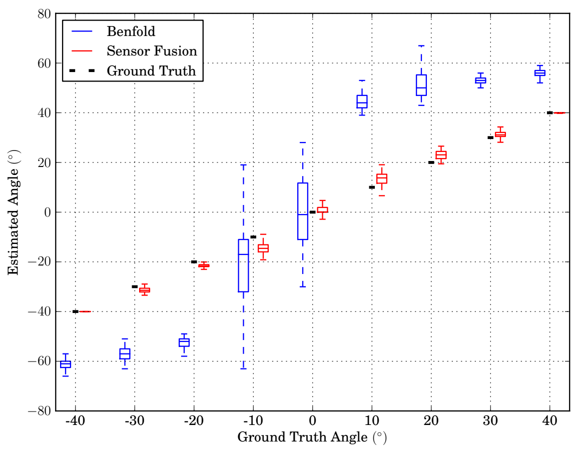
Due to the restrictions of most infrared gaze estimation methods, as discussed in Section 2.5, and the requirements of exactly replicating the hardware setup and experimental conditions, such as lighting, a meaningful results comparison would be prohibitively difficult. Thus, the *sensor fusion* results were compared against a state-of-the-art RGB gaze detection method [7]. As this method has a restriction on the camera viewing angle when used with moving subjects, only results from the static scenarios could be compared. As can be seen from the results in Figure 5.14, the *sensor fusion* method greatly outperforms the competing method. For every distance-angle combination, the median value is closer to the ground truth and interquartile range is smaller. The proposed algorithm is also more computationally efficient, producing



(a) 1.5 m



(b) 2.0 m



(c) 2.5 m

Figure 5.14: Box plots comparing gaze angles generated by the proposed *sensor fusion* method and a state-of-the-art RGB gaze detection method [7] for the static scenario. At every ground truth angle, the results from the comparison method are plotted in blue to the left, whilst the *sensor fusion* results are plotted in red to the right. For every distance-angle combination, the *sensor fusion* method has a more accurate median value and a smaller interquartile range.

gaze estimates at a full 30 Hz, whilst the competing method would occasionally drop to around 20 Hz due to the computational expense of the HOG method employed [26].

The results in Figure 5.14 were shown to the author of the method, who confirmed that they were representative of the algorithm's performance. He noted that the poor performance at 0° and 1.5 m was probably because the subject had little visible hairline. During classifier training, it was found that the classifier drew many of its features from the hairline of the subject, as it provided a good indication of the subject's gaze direction.

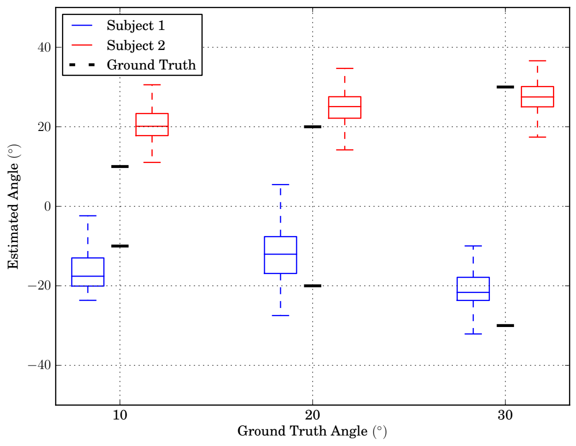


Figure 5.15: Box plot of the *sensor fusion* gaze angle for two simultaneously moving people. The corresponding results are displayed at either side of each ground truth angle. Due to the fast velocity of the subjects in these experiments, some inaccuracies arise as a result of the greater depth camera noise than in previous experiments. However, the results are still more accurate than the comparison RGB method in the more noiseless static scenario (Figure 5.14). Additionally, the distance of 2.5 m at which the more inaccurate values occur, exceeds the range of other IR gaze estimation methods.

5.3.4 Gaze Estimation in HRI

In order to further evaluate the *sensor fusion* algorithm under real-world conditions, the moving person scenario (Figure 5.8b) was repeated with two people moving at the same time. To accomplish this, the people detection method from Chapter 3 was used, with the Kalman filter tracker from Chapter 4. The Hungarian algorithm, described in Section 4.2.1, was used to assign detected gazes to tracked people.

Three experiments were performed. In each, one person's gaze was the opposite of the other. Thus, instead of seven different experiments with angles $(-30°, -20°, \ldots, 20°, 30°)$, three experiments were performed with angles $(\pm 30°, \pm 20°, \pm 10°)$. The results are shown in Figure 5.15.

Due to depth camera noise arising from the fast moving people, the gaze estimates at 10° are slightly over estimated. However, the results are still more accurate than the static results of the competing RGB method (Figure 5.14). It should also be noted that the distance of 2.5 m at which most of these inaccurate values seem to occur is much greater than the working range of most other infrared gaze estimation methods.

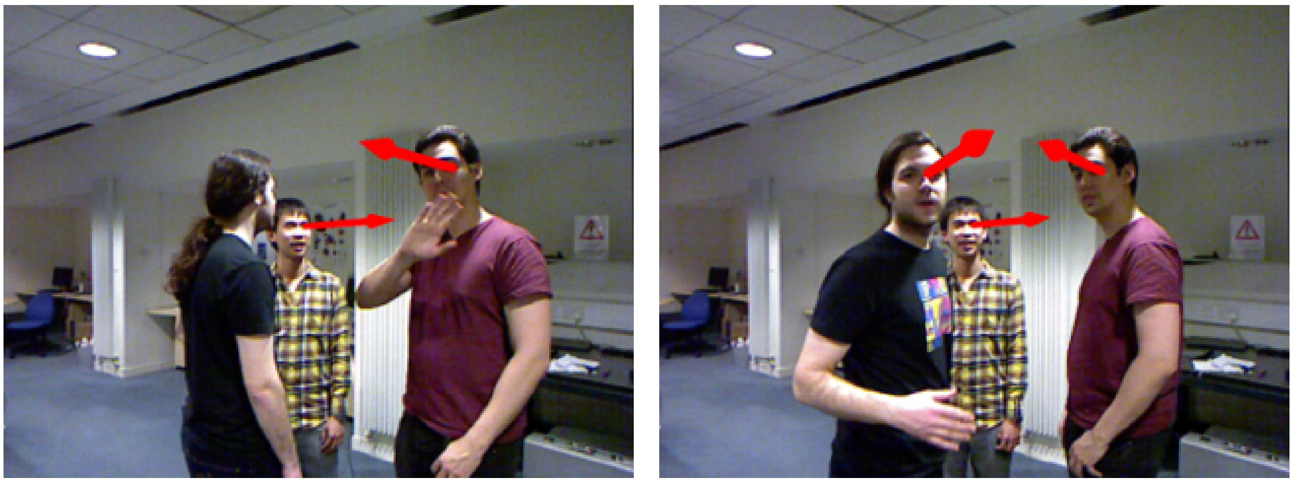


Figure 5.16: Images, from the robot's view, of the *sensor fusion* algorithm's performance in typical HRI scenes. Gaze estimates are displayed with a red arrow, that has been projected into an RGB image for visualisation purposes. By estimating the gaze of everyone in the scene, the subject gesturing towards the robot can be correctly identified as wanting to initiate HRI.

Images of the *sensor fusion* algorithm's performance in typical HRI scenes are shown in Figure 5.16. Two background subjects are not interacting with the robot, but by analysing all three peoples' gaze, the gesturing subject can be correctly identified as wanting to initiate HRI.

Crowds and dynamics cause problems for many existing gaze detection methods. However, the proposed method is designed to robustly work in real-world environments. To assess this, the robot was placed outside a lecture hall, and recorded people leaving a lecture. Gaze estimates of the people walking past are shown in Figure 5.17. These scenes were previously illustrated in Figure 5.1, as the types of crowded scenarios where accurate gaze estimates are needed for effective HRI.

5.4 Conclusions

The contribution of this chapter has been a sensor-fusion based gaze estimation method, using both infrared pupil detection and depth camera data, for use on a mobile robot. The method is designed to facilitate HRI tasks, such as human-attention detection and group detection.

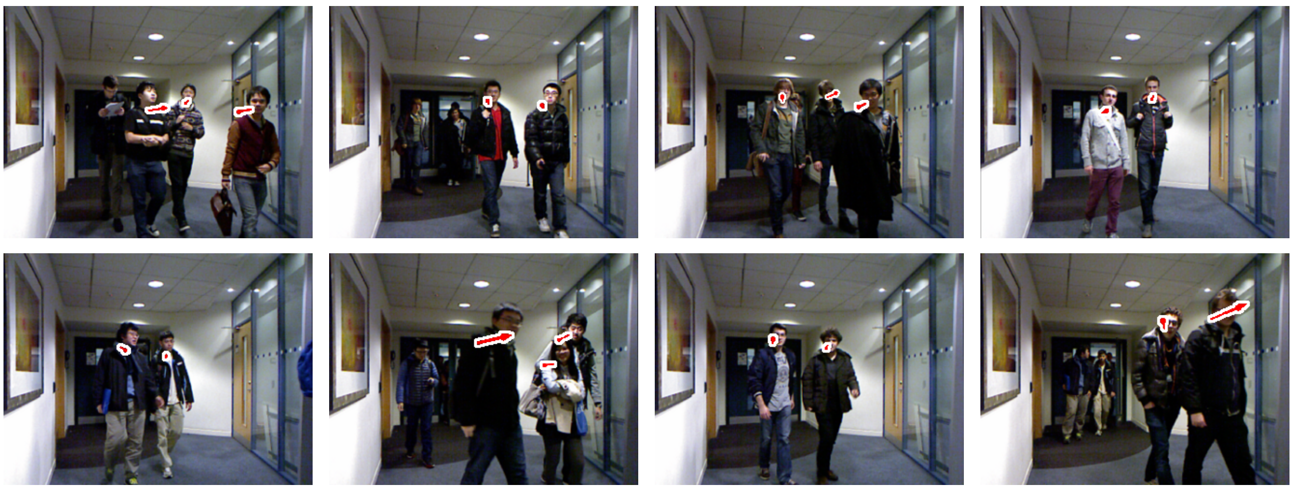


Figure 5.17: Images from the robot's view, showing the estimated *sensor fusion* gaze angles in real-world scenarios, including those originally shown in Figure 5.1. Gaze estimates are displayed with a red arrow on a white background, that has been projected into an RGB image for visualisation purposes. For people looking into the camera, this arrow will naturally appear quite short.

Most existing methods of infrared gaze estimation have too short a range to be applied to this problem area, as well as user-specific calibration. Many traditional vision-based methods lack the gaze accuracy required for HRI tasks. With the benefits of the proposed sensor fusion method, these drawbacks are compensated. The calibration-free method works for natural HRI distances of 1.2 m to 2.5 m, is not constrained by the number of moving or static people in a scene, and gives accurate gaze estimates at a 30 Hz rate.

The algorithm used two arrays of infrared LEDs, one near the camera axis that illuminates a subject's pupil, and one off-axis array that provides background illumination. Each array is activated in alternate frames, with the image difference providing clearly segmented pupils. Based on the detected eyes of each subject, two gaze estimates are generated: one based on forehead analysis from depth images, and a more accurate sensor-fusion estimate that combines the forehead estimate with eye positions in the infrared image. Each sensor-fusion gaze estimate involves numerically solving an equation using the Newton-Raphson method. However, the equation has two roots and only one is the correct gaze estimate. To select the root corresponding to the subject's gaze, the estimate from the forehead analysis is used.

The accuracy of the proposed sensor-fusion method was evaluated under multiple conditions:

using a static subject, using multiple moving people and with a moving robot. Forty six experiments were performed, equating to over 10 minutes of ground truth annotated data. The performance of the sensor fusion method was tested against a state-of-the-art vision-based method and the results of the forehead analysis method. In the vast majority of cases, the sensor-fusion algorithm greatly outperformed the comparison methods, with a higher running speed than the vision-based method.

Chapter 6 uses the proposed gaze estimation method to detect groups. An analysis is presented of how the inclusion of gaze information affects group detection accuracy. Without a suitable method of evaluating gaze direction, many HRI methods would fail when exposed to background human-human interaction. In future work, the proposed gaze estimation method should be combined with the hand detection and tracking modules presented in Chapters 3 and 4. With an added gesture recognition module, this work would allow a robot to ignore gestures directed at humans, and respond to gestures aimed at the robot.

Chapter 6

Multi-Modal Gaze Contingent Group Detection

6.1 Introduction

To seamlessly integrate robots into human-environments, methods must be developed by which robots can understand human social context: a robot should avoid interrupting an interacting group, it should not navigate through the middle of two conversing people, but it should consider interacting with an ungrouped person. This problem of group detection impacts many additional aspects of a robot's behaviour. For example, a robot that detects a gesturing subject should not respond if they are gesturing to someone else. The contribution of this chapter is to provide a method of group detection for mobile robots, which uses detected subjects' gaze directions to robustly detect static and dynamic interactions.

People detection and tracking is a widely studied problem in computer vision and robotics. Much less research tackles the problem of group identification, with many methods using only spatial and motion information [147, 148]. However, real-world environments are frequently cluttered, crowded, and have ungrouped people passing close to each other. In these situations, knowledge of only a subject's position and velocity is not enough to robustly determine their group participation. A more informative indicator must be used.

Chapter 5 proposed a real-time gaze estimation method, which used infrared eye detection and depth image analysis. When combined with the people detection method from Chapter 3, accurate gaze estimates on a mobile robot were obtained for multiple moving people. Such gaze estimates are an important indicator of group participation, especially in narrow or crowded environments.

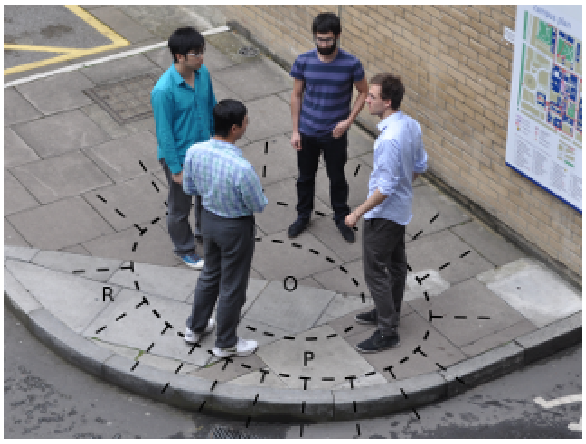
Many previous group detection methods make no distinction between the different structures of static and dynamic groups [8,9,149,150]. However, from existing sociological literature, specific models for both can be defined. With a detection method that explicitly handles these two distinct formations, the resulting group classification accuracy can be increased.

The proposed group detection method thus uses the detected gaze of subjects in a scene, combined with their position and velocity information. From this information, a novel set of group detection features is defined, based on sociological concepts of static and dynamic group structures. The proposed method firstly identifies interacting pairs of people in a scene, using a subsequent recursive algorithm to complete the group detection process. Two classification approaches are compared: a model-based method, which calculates interaction probability using multiple likelihood functions, and a logistic regression model [151]. Group detection accuracy is evaluated under a wide range of group configurations, using moving and stationary subjects. The performance of the employed features is also detailed, with the impact of gaze information on group classification explicitly quantified. Finally, performance is compared to the state-of-the-art, using four external datasets.

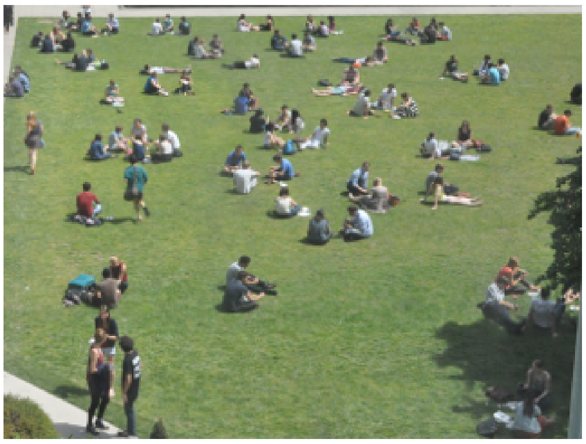
6.2 Method

6.2.1 Sociological Group Concepts

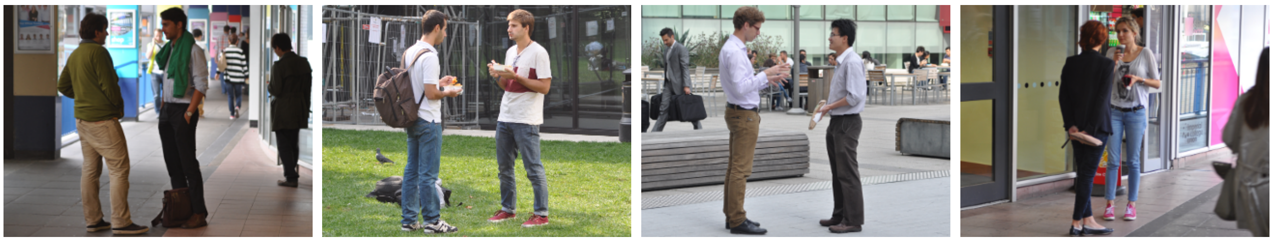
Sociology is the study of social behaviour. In the field of sociology, an important concept in the formal definition of groups is the "facing", F-formation [152]. The F-formation is a circular arrangement of the spatial-orientations of group members, such that they form a shared



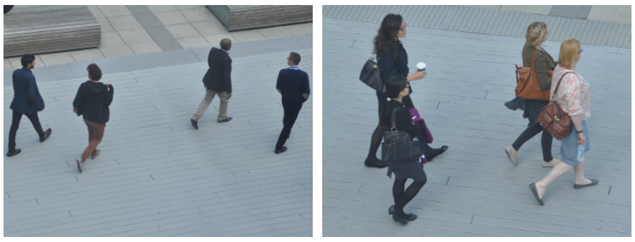
(a) Picture of a conversing group formed in an F-formation, with the components of their shared transactional spaced annotated. Conversation is directed into the inner "O"-space. To achieve this, participants position themselves in the "P"-space. Areas outside the group lie in the "R"-space.



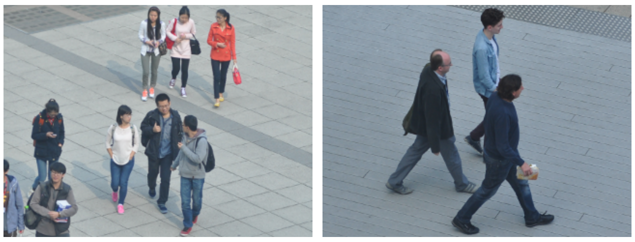
(b) Wide shot picture showing multiple groups naturally formed by F-formations



(c) Images of static pairs



(d) Images of dynamic pairs



(e) Images of dynamic groups

Figure 6.1: (a) and (b) show images of circular F-formations that static groups of people naturally adopt when interacting, and especially when conversing. Images of static pairs in (c), show a different arrangement called the vis-a-vis F-formation. As shown in images from (d), dynamic pairs of people also form a different structure, termed the side-by-side arrangement. As dynamic group size increases, (e), the side-by-side arrangement sometimes separates, but generally the participants' gaze still maintains a common focus.

transactional space. This formation is common amongst static groups of talking people, and is illustrated in Figures 6.1a and 6.1b. As shown, the space around the formation can be divided into a number of sections. The “O”-space is the inner part of the group’s space, and is where interactions are directed. Areas outside the group’s transactional space are defined as the “R”-space. Participants of the group position themselves between these two areas, in an area known as the “P”-space. The dimensions of these spaces naturally depend on the number of participants in the group. Kendon doesn’t explicitly state what these letters stand for.

Kendon notes that in conversations between two static people, a different arrangement is common, where participants directly face each other. Shown in Figure 6.1c, this arrangement is termed the vis-a-vis F-formation. In a similar way, dynamic groups have their own unique formations. Figure 6.1d shows candid images of dynamic pairs forming what Kendon terms a side-by-side arrangement. Figure 6.1e shows slightly larger dynamic groups. As group size increases, the side-by-side arrangement sometimes separates, but generally the participants’ gaze still maintains a common focus.

A related concept in the definition of groups is that of proxemics, introduced by the anthropologist Edward T. Hall. Hall described proxemics as: “The interrelated observations and theories of man’s use of space as a specialised elaboration of culture” [141]. Defined as the study of humans, anthropology and sociology have complimentary theories for defining groups. Expounding on this topic, Hall explores how the space required for interactions between people is influenced by their environment. Through observations and interviews with healthy adults from the United States, Hall defined four different distance categories that govern the different types of interpersonal interactions that occur within them.

Actions such as wrestling or comforting take place at intimate distances of 0–50 cm. Interactions amongst close acquaintances take place in personal distances of 50–120 cm. Impersonal business takes place within social distances of 1.2–3.5 m, though most involvement happens up to 2.1 m. Finally, little interactions other than public speaking occur at public distances of 3.65 m and above.

It is common within social environments for people to arrange themselves in groups with an

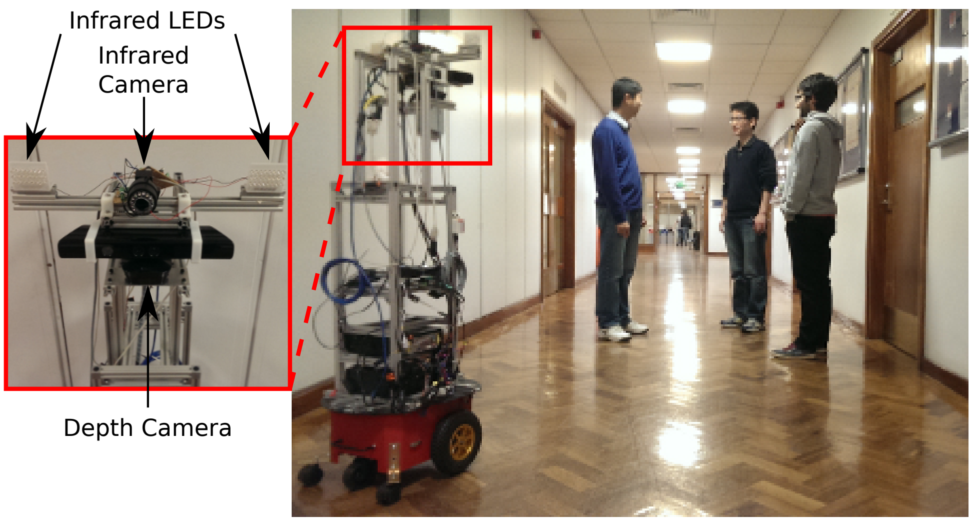


Figure 6.2: Image of robot equipped with an infrared gaze detector and a depth camera for people detection. The proposed group detector can reduce disruption by enabling the robot to either move around, not through, the group or wait until the conversation has finished before interacting with a participant.

F-formation, as shown in Figure 6.2. For a mobile robot to accurately detect interacting groups, a method of detecting people and their associated gaze must be defined. To provide robustness in crowded environments, the people detector from Chapter 3 was employed. Building further on previous work, gaze estimation was performed using the method from Chapter 5. This method uses on and off-axis infrared LEDs to segment pupils within a scene, with additional depth-image analyses used in calculating subjects' gaze angles. Section 5.3 displays the accurate results of this gaze detector, at distances of 1.2 m to 2.5 m.

A flowchart of the proposed group detection algorithm is shown in Figure 6.3. This method is motivated by conceptualising a group as a collection of interacting persons. Thus, pairs of interacting people within a scene are initially identified. Any detected pairs that share a common participant are merged together. People that are have not been assigned to a group are then evaluated, to see if they are interacting with an existing group. If so, then they are merged into the appropriate group. This last step is repeated until no further unassigned person is added to the groups in the scene.

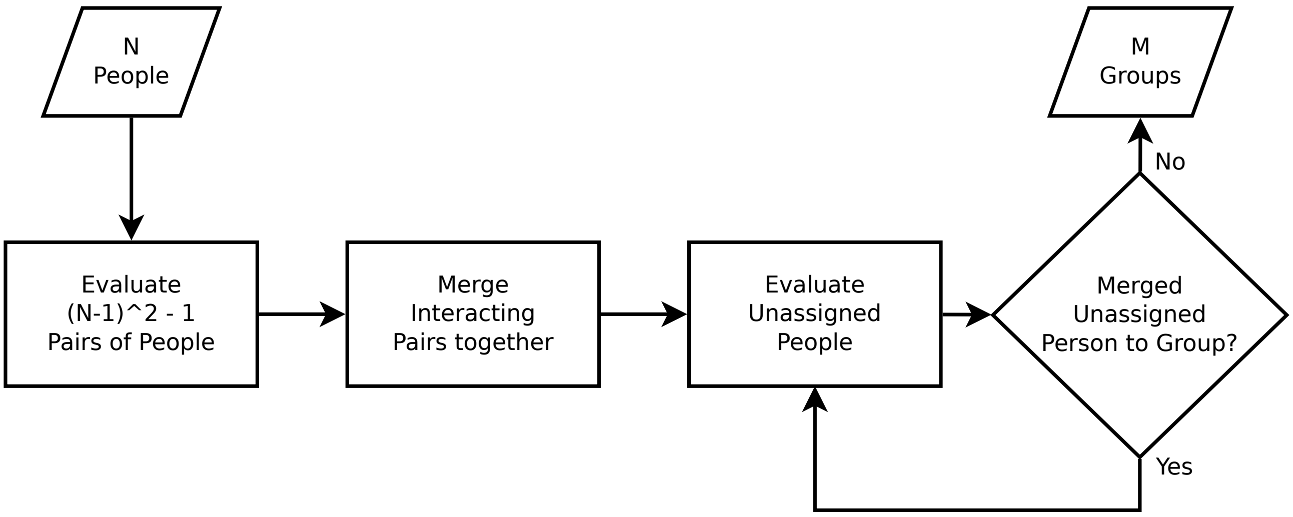


Figure 6.3: A flowchart of the proposed group detection algorithm.

6.2.2 Pairwise Group Detection

The first step in the proposed group detection algorithm is to detect whether two people are interacting. To achieve this, a *pairwise* interaction probability is generated, based on three modalities: the distance between the two people, their gaze direction and their speed. The *pairwise* probability will be generated for all permutations of possible pairs in the scene. Two ways of generating the *pairwise* interaction probability will be presented and evaluated in Section 6.3: a model-based method and classification using logistic regression.

In the model-based method, three separate likelihood functions are defined; one for each of the distance, gaze and velocity modalities. Each function takes, as input, features from the two people being evaluated. Function outputs are a normalised probability; high values indicate a high probability that the two people are interacting. Each likelihood function is then combined to give the final *pairwise* interaction probability. A logistic regression model is also used to generate a *pairwise* interaction probability, using the same features from the model-based method as input.

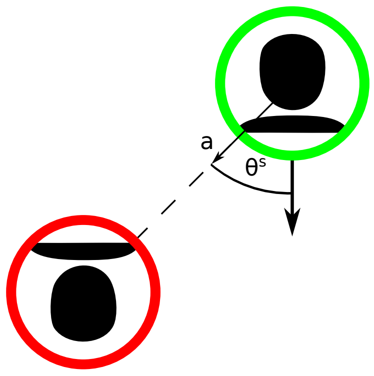
Pairwise Feature Extraction

In Section 6.2.1 it was noted that as the space between two people increases, the nature of possible interactions changes. At public distances, above 3.65 m, meaningful interaction with another person is unlikely. Thus, the distance between two people in a scene can be used to indicate interaction likelihood. In defining the distance feature for generating the *pairwise* interaction probability, a person's height is ignored, and their position is defined as $\mathbf{x} = [x, z]^T$. The feature chosen is the Euclidean distance between two people, and is denoted as $\|\Delta_{\mathbf{x}}\|$.

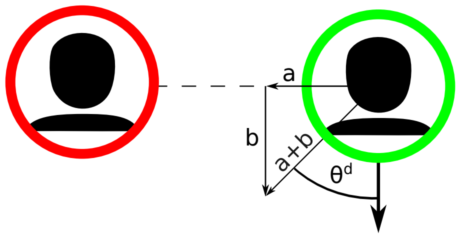
A person's gaze direction is an obvious cue of their focus of attention. Gaze features are thus a natural indicator of *pairwise* interaction probability, and can be extracted using the gaze estimation method from Chapter 5. There are situations when the gaze detector fails to detect a person's gaze: if they look away from the camera, if they blink, or if they look down to the ground. To explicitly model this situation a feature, g, shall be introduced. If the gaze of pair participant $i \in \{1, 2\}$ is detected, then $g_i = 1$, otherwise $g_i = 0$. If $g_i = 0$, as replacement for their detected gaze, person i's assumed gaze is directly away from the camera.

As illustrated in Figure 6.1c, when two static people interact it is natural for their gaze be focused on each other, as a result of the common vis-a-vis formation. It can thus be assumed that when one person looks directly at another, they have a high *pairwise* interaction probability. To model this, a static gaze feature, θ_i^s, shall be defined that encodes the angular difference between a person's gaze and the other person in the pair being evaluated. Figure 6.4a illustrates how θ_i^s is calculated. In all equations, the value $\cos(\theta_i^s)$ is used as the static gaze feature, instead of θ_i^s, because it is normalised between 1 and -1.

As shown in Figure 6.1d, dynamic pairs of people have a side-by-side arrangement rather than a vis-a-vis formation. This changes their natural gazing direction, due to the need to both look where they are going and maintain interaction. Gaze directions for dynamic pair participants are frequently focused between the direction of motion and the other participant. Another example of this can be seen in Figure 6.5. It can thus be assumed that when a moving person has a gaze direction that is the average of their motion direction and the direction of the



(a) Illustration of a static pair. It is natural for interacting participants to look directly at each other, shown by the vector "a". θ^s is the angular difference between a person's gaze and the other participant.



(b) Illustration of a dynamic pair. It is assumed that a pair participant's gaze is focused between the direction of the other person, given by the vector "a", and their motion direction, given by the vector "b". θ^d is the angular difference between a person's gaze and the mean of vectors "a" and "b".

Figure 6.4: Illustration of the gaze features, θ^s and θ^d, for generating the *pairwise* interaction probability. In each illustration, gaze features are calculated for the person given by the green circle, with the other pair participant represented by the red circle. The unmarked arrow extending from the green person denotes their gaze direction.

other person in the pair being evaluated, they have a high *pairwise* interaction probability. A dynamic gaze feature, θ_i^d, is introduced to model this, encoding the angular difference between a person's detected gaze and this assumed gazing direction for dynamic pair participants. Figure 6.4b illustrates graphically how θ_i^d is calculated. Again, the value $\cos\left(\theta_i^d\right)$ shall rather be used as the dynamic gaze feature, due to its normalised range.

To maintain a constant interaction space, the velocities of interacting pair participants must be similar. Again this can be seen from Figure 6.1. Thus, two velocity features are used in generating the *pairwise* interaction probability: μ_v, the magnitude of the mean velocity of the two people; and Δ_v, the magnitude of the velocity difference of the two people.

With all features used to generate the *pairwise* interaction probability now defined, the complete list can be stored as a vector: $\mathbf{f}_{p-p} = [||\Delta_{\mathbf{x}}||, \cos\left(\theta_1^s\right), g_1, \cos\left(\theta_2^s\right), g_2, \mu_v, \Delta_v, \cos\left(\theta_1^d\right), \cos\left(\theta_2^d\right)]^T$. The task remains to classify interacting pairs of people, given a pairwise feature vector extracted from each permutation of possible pairs in a scene.

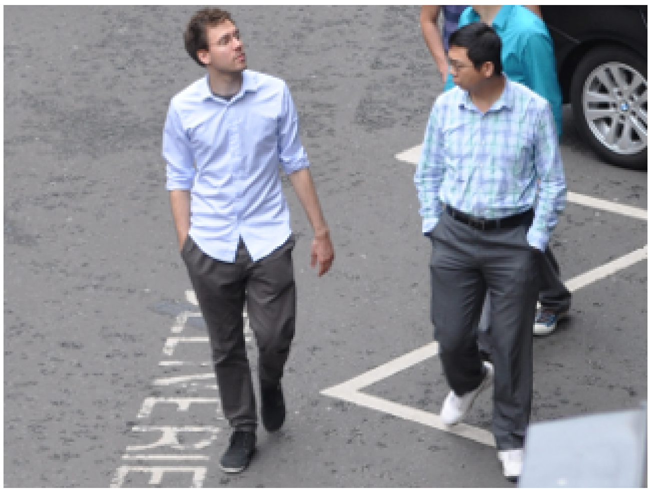


Figure 6.5: Image of a pair moving towards the camera. Each participant's gaze is approximately half way between the direction of motion, and the other person. This is due to the need to look where they are going, whilst maintaining interaction.

Pairwise Group Classification

With the distance, gaze and velocity features defined, pairwise group classification can be performed using either the model-based method or through logistic regression. However, the three model-based likelihood functions for the three modalities must still be defined. The function inputs are all elements of $\mathbf{f}_{p-p}$.

In the model-based pair classification method, the distance likelihood function shall be denoted as $s_d\left(\|\Delta_{\mathbf{x}}\|\right)$. Intuitively, people located within personal distances (0–1.2 m) [141] should have a high distance likelihood. As two people move further apart, this likelihood should exponentially decrease. This behaviour can be modelled by the following piecewise Gaussian function:

$$s_d\left(\|\Delta_{\mathbf{x}}\|\right) = \begin{cases} 1 & \text{if } \|\Delta_{\mathbf{x}}\| < u_d, \\ \exp\left(-\left(\frac{\|\Delta_{\mathbf{x}}\|-u}{\sigma_d}\right)^2\right) & \text{otherwise.} \end{cases} \tag{6.1}$$

u_d is a positive variable that denotes the distance within which two people will have a maximum distance likelihood. For distances outside this value, the positive σ_d value controls how quickly the likelihood decreases. Low values of σ_d result in probabilities that decrease faster with distance. Figure 6.6 illustrates the effect of distance between two people on their distance

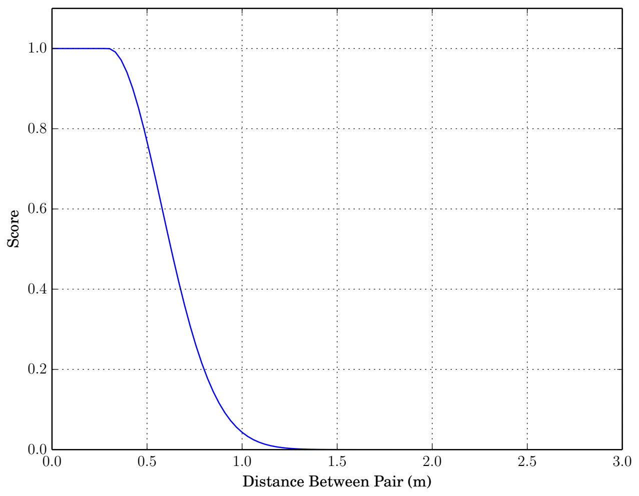


Figure 6.6: A graph of how the model-based, *pairwise* distance likelihood varies with distance between two people. Likelihood is maximal at personal distances, and decreases exponentially as the candidates move further apart. This graph was generated using the optimal model-based parameters, as defined in Section 6.3.2. Thus, $u_d = 0.296$ and $\sigma_d = 0.397$.

likelihood probability.

The gaze likelihood function for the model-based pair classification method shall be denoted as $s_\theta(\theta_i, g_i)$. θ_i is the angular difference between pair participant i's detected and expected gaze directions, as shown in Figure 6.4. For static pair participant i, this difference has already been defined as θ_i^s. The corresponding gaze likelihood is thus $s_\theta(\theta_i^s, g_i)$. Similarly, for dynamic pair participant i, the difference between detected and expected gazes has been defined as θ_i^d. This corresponding gaze likelihood is $s_\theta\left(\theta_i^d, g_i\right)$.

The guiding principle when defining $s_\theta(\theta_i, g_i)$ is that two people looking in their expected gaze directions should have a maximum *pairwise* gaze likelihood. As their focus of attention moves away from the expected values, likelihood should exponentially decrease. A scaled von Mises distribution, being the circular equivalent of the Gaussian function, can be used to capture this behaviour:

$$s_\theta(\theta_i, g_i) = \begin{cases} \exp(\tau\cos(\theta_i) - \tau) & \text{if } g_i = 1, \\ c & \text{otherwise,} \end{cases} \tag{6.2}$$

$$s_\theta(\theta_1, g_1, \theta_2, g_2) = s_\theta(\theta_1, g_1) \cdot s_\theta(\theta_2, g_2). \tag{6.3}$$

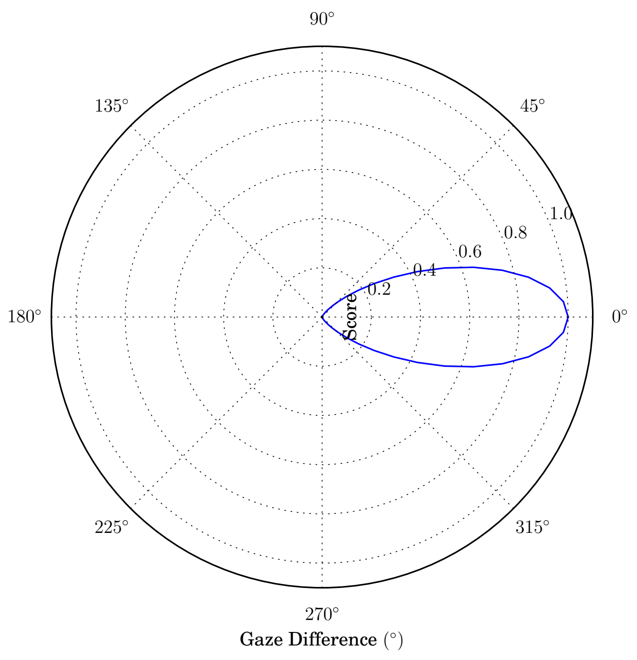


Figure 6.7: A polar plot of how the model-based, *pairwise* gaze likelihood, varies with a pair candidate's gaze angle. Gaze difference, θ, is the angular difference between the person's detected and expected gaze directions, as defined in Figure 6.4. A difference of 0° thus gives a maximal likelihood. As the person's gaze deviates from its expected direction, likelihood decreases appropriately. This graph was generated using the optimal model-based parameters, as defined in Section 6.3.2. Thus, $\tau = 8.7$.

Note that a non-detected gaze gives a constant likelihood of c. τ is a positive variable that controls how quickly a person's gaze likelihood decreases, as their gaze deviates from its expected direction. A higher τ results in faster decreasing probabilities. The effect of a person's gaze direction on gaze likelihood is shown graphically in Figure 6.7.

Finally, the velocity likelihood function for the model-based pair classification method shall be denoted as $s_v(\mu_v, \Delta_v)$. This function should assign a high interaction likelihood to subjects with similar velocities, especially as their average velocity increases. This behaviour can again be captured with a piecewise Gaussian function:

$$s_v(\mu_v, \Delta_v) = \begin{cases} \exp\left(-\left(\frac{\Delta_v}{\sigma_v}\right)^2\right) & \text{if } \mu_v > u_v, \\ \frac{\mu_v}{u_v}\exp\left(-\left(\frac{\Delta_v}{\sigma_v}\right)^2\right) & \text{otherwise.} \end{cases} \tag{6.4}$$

If a pair's μ_v is above the positive variable u_v, then their velocity likelihood is given by an

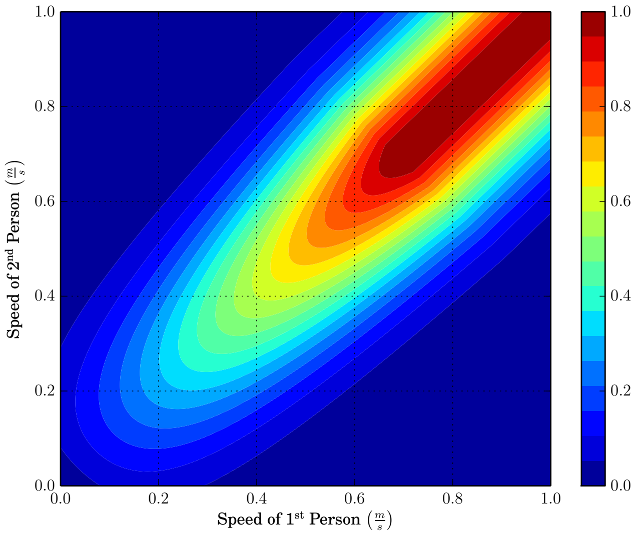


Figure 6.8: A plot of how the model-based, *pairwise* velocity likelihood for two candidates varies with their speed. Plotted values assume that they are both facing in the same direction. Velocity likelihood is maximal when their speeds are similar, and above a threshold. The more the candidates' speed differs, the more the interaction likelihood decreases. This graph was generated using the optimal model-based parameters, as defined in Section 6.3.2. Thus, $\sigma_v = 0.267$ and $u_v = 0.549$.

exponential function that has a maximal value when both participants have the same velocity. If μ_v drops below u_v then likelihood is reduced, as slower velocities provide less information about groupings in the scene. σ_v is a positive variable that controls how quickly interaction likelihood decreases as the velocity of the two people differs. Low values of σ_v result in more quickly decreasing values. The effect of the pair's speed on velocity likelihood is shown in Figure 6.8.

The three model-based likelihood functions defined for distance, gaze and velocity modalities must be combined to generate the *pairwise* interaction probability. In accomplishing this, the differences between static and dynamic groups, shown in Figures 6.1c and 6.1d, must be incorporated. Thus, the following equation for the model-based *pairwise* probability places more weight on the static gaze features for slow moving pairs, with the dynamic gaze features taking precedence for faster pairs.

$$\begin{aligned} s^m_{p-p}\left(\mathbf{f}_{p-p}\right) = &\left(1 - s_v\left(\mu_v, \Delta_v\right)\right) s_\theta\left(\theta^s_1, g_1, \theta^s_2, g_2\right) s_d\left(\|\Delta_\mathbf{x}\|\right) \\ &+ s_v\left(\mu_v, \Delta_v\right) s_\theta\left(\theta^d_1, g_1, \theta^d_2, g_2\right) s_d\left(\|\Delta_\mathbf{x}\|\right). \end{aligned} \tag{6.5}$$

Whilst s_θ is parameterised in terms of θ_i for clarity, only $\cos\left(\theta_i\right)$ is directly used in s_θ and $\mathbf{f}_{p-p}$. Two people whose *pairwise* interaction probability is above a threshold, $s^m_{p-p} > \lambda^m_{p-p}$, are defined to be interacting as a group. The values of model-based parameters $\mathbf{p}_{p-p} = \left(u_d, \sigma_d, \tau, c, u_v, \sigma_v, \lambda^m_{p-p}\right)$ will be defined in Section 6.3.2.

As stated previously, a logistic regression model can also be used to generate a *pairwise* interaction probability, using the same input features, $\mathbf{f}_{p-p}$, as the model-based method. In explaining the logistic method, a dummy vector, $\phi = [1, \mathbf{f}^T_{p-p}]^T$, of size 10×1, shall be created for every 9×1 pairwise feature, $\mathbf{f}_{p-p}$. The resulting *pairwise* probability from the logistic method shall be denoted as s^l_{p-p}. Standard logistic regression uses the regression coefficients $\mathbf{w}$ to model the equation:

$$s^l_{p-p} = \sigma\left(\mathbf{w}^T\phi\right), \tag{6.6}$$

$$\begin{aligned} \ln\left(\frac{s^l_{p-p}}{1 - s^l_{p-p}}\right) = &\, w_0 + w_1\|\Delta_\mathbf{x}\| + w_2\cos\left(\theta^s_1\right) + w_3 g_1 + w_4\cos\left(\theta^s_2\right) + w_5 g_2 \\ &+ w_6\mu_v + w_7\Delta_v + w_8\cos\left(\theta^d_1\right) + w_9\cos\left(\theta^d_2\right), \end{aligned} \tag{6.7}$$

where σ is the logistic sigmoid function. However, this function will give a different s^l_{p-p} value if the order of pair participants, $i \in \{1, 2\}$, are swapped when extracting $\mathbf{f}_{p-p}$. Specifically, the terms involving $\cos\left(\theta^s_1\right)$ and $\cos\left(\theta^s_2\right)$, g_1 and g_2, $\cos\left(\theta^d_1\right)$ and $\cos\left(\theta^d_2\right)$, must be made order-independent. This can achieved by enforcing their corresponding regression coefficients to be equal:

$$\begin{aligned} \ln\left(\frac{s^l_{p-p}}{1 - s^l_{p-p}}\right) = &\, w_0 + w_1\|\Delta_\mathbf{x}\| + w_2\left(\cos\left(\theta^s_1\right) + \cos\left(\theta^s_2\right)\right) + w_3\left(g_1 + g_2\right) \\ &+ w_4\mu_v + w_5\Delta_v + w_6\left(\cos\left(\theta^d_1\right) + \cos\left(\theta^d_2\right)\right). \end{aligned} \tag{6.8}$$

With the number of regression coefficients reduced from 10 to 7, this technique has the added benefit of requiring less data to train the classifier. In the same way as the model-based method, two people are said defined to be interacting as a group if their resulting *pairwise* probability is above a threshold: $s^l_{p-p} > \lambda^l_{p-p}$. By ordering the output probabilities for all negative training data samples, λ^l_{p-p} is set equal to the probability sample that gives the desired false positive rate of the classifier.

Both the model-based method and the logistic classifier can be used to list all pairs of interacting people in a scene. However, this list of pairs can contain overlapping people, as might be the case with a group of three or more people. Thus, all pairs with a common participant should thus be merged. This will result in a list of groups with unique participants, and a list of people that have been labelled as not interacting with another person.

6.2.3 Person-to-Group Merging

In groups of three or more people, a participant does not have to directly interact with a specific person. For example, one person might direct their attention between the two other group participants. To resolve this, the merged pairs of interacting people generated in Section 6.2.2 will be used as an initial set of groups, as shown in Figure 6.3. For the remaining unassigned people, a method of detecting whether they are interacting with an existing group must be devised.

To this end, a *person-to-group* interaction probability is introduced. The *person-to-group* probability will be calculated for each permutation of unassigned people to existing groups. To generate it, the same two methods from the pairwise group detection step are used: a model-based method and a logistic classifier.

Classifying a person as interacting with a group is a different problem than classifying two people as interacting. It requires different models, training and test data. Thus, the *person-to-group* probability is not calculated using the pairwise features $\mathbf{f}_{p-p}$. A different set of features is extracted, though still from the same three modalities: distance, gaze and velocity.

Person-to-Group Feature Extraction

In Figure 6.1b it can be seen that different groups with the same number of people can be spread out over different areas. Specifically, in the F-formation all people are roughly equidistant from the mean. This distance is not constant however: a person interacting with a small, tight group will be closer to its mean than a person interacting with a large, spread out group. A sensible distance feature for calculating *person-to-group* interaction probability should thus take into account the spread of the people within the existing group. This is accomplished using Mahalanobis distances, rather than the Euclidean distances used in $\mathbf{f}_{p-p}$.

To calculate the Mahalanobis distance between an unassigned person and an existing group, the covariance matrix, $\mathbf{C}$, of the positions, $\mathbf{x} = [x, z]^T$, of all group members is calculated. For group a group of size N, with mean position, $\bar{\mathbf{x}} = [\bar{x}, \bar{z}]^T$, the covariance matrix is defined as:

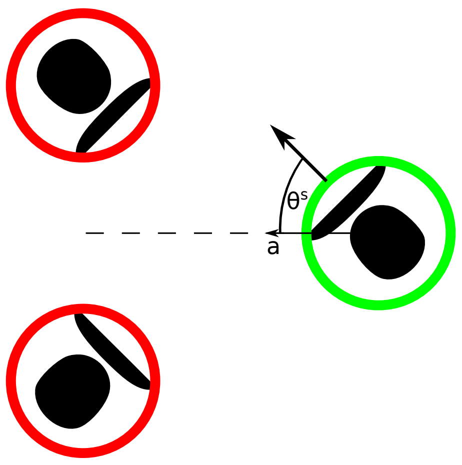
$$\mathbf{C} = \frac{1}{N} \sum_{k=0}^{N} (\mathbf{x}_k - \bar{\mathbf{x}})(\mathbf{x}_k - \bar{\mathbf{x}})^T.$$

The Mahalanobis distance, D_M, of an unassigned person with position $\mathbf{x}$, to this group is given by:

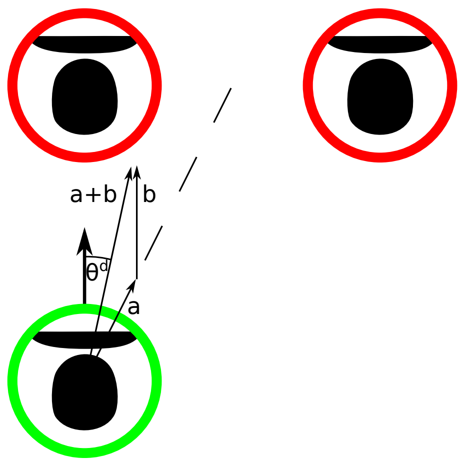
$$D_M(\mathbf{x}) = \sqrt{(\mathbf{x} - \bar{\mathbf{x}})^T \mathbf{C}^{-1} (\mathbf{x} - \bar{\mathbf{x}})},$$

The gaze features used to generate the *person-to-group* probability are the same used for the *pairwise* probability, with two modifications. Firstly, a group cannot be said to have a gaze of its own, and thus only one vale of θ^s and θ^d is used. Secondly, the expected gaze direction of the unassigned person must be redefined, to take all members of the existing group into account.

For static groups, as shown in Figure 6.1a, a participant's gaze direction is naturally focused towards "O"-space in the F-formation; this can be approximated as the mean position of the other group members. Figure 6.1e suggests that as dynamic groups increase in size, the side-by-side arrangement of the participants sometimes separates, but generally the participants' gaze still maintains a common focus. Similar to the case of dynamic pairs, group participants need to both look where they are going and maintain interaction. These expected gazing



(a) Illustration of a static group. It is natural for an interacting person to look towards the mean of the remaining group participants, shown by the vector “a”. θ^s is the angular difference between an unassigned person’s detected gaze and this expected gaze direction.



(b) Illustration of a dynamic group. It is assumed that a group participant’s gaze is focused between the mean position of the remaining group participants, given by vector “a”, and the group’s direction of motion, given by vector “b”. θ^d is the angular difference between an unassigned person’s gaze and the mean of vectors “a” and “b”.

Figure 6.9: Illustration of the gaze features, θ^s and θ^d, for generating *person-to-group* interaction probability. In both illustrations, gaze features are calculated for the unassigned person given by the green circle. Members of the existing group are denoted by red circles. The unmarked arrow extending from the unassigned green person denotes their gaze direction.

directions are shown for an unassigned static person, in Figure 6.9a, and an unassigned dynamic person, in Figure 6.9b. Equivalent to their *pairwise* probability definitions, θ^s and θ^d denote the angular difference between the unassigned person’s detected and expected gaze directions, when interacting in an existing static and dynamic group respectively.

The velocity features of the *person-to-group* interaction probability are equivalent to those of the *pairwise* probability. μ_v is the magnitude of the average velocities of the unassigned person, and mean velocity of the group participants. Δ_v is the magnitude of the velocity difference between the unassigned person, and the mean velocity of the group participants.

The complete list of features used to calculate the *person-to-group* probability are thus: $\mathbf{f}_{p-g} = \left\{M_D\left(\mathbf{x}\right), \cos\left(\theta^s\right), g, \mu_v, \Delta v, \cos\left(\theta^d\right)\right\}$. The remaining task is to use these features, generated from each permutation of unassigned people to existing groups, and classify whether they are interacting together or not.

Person-to-Group Classification

Classifying an unassigned person as interacting with an existing group or not, can again be performed with either the model-based method or logistic regression. The three model-based likelihood functions used to generate the *person-to-group* interaction probability are the same as for the *pairwise* probability, with one small modification: as there is only one static and one dynamic gaze feature, the gaze likelihood function is given by Equation 6.2, rather than Equation 6.3. The *person-to-group* gaze likelihood is thus given by $s_\theta\left(\theta^s, g\right)$ when evaluating an unassigned static person and group, and $s_\theta\left(\theta^d, g\right)$ when evaluating an unassigned dynamic person and group.

The equation to calculate the model-based *person-to-group* probability is very similar to that for the *pairwise* probability, given by Equation 6.5:

$$s^m_{p-g}\left(\mathbf{f}_{p-g}\right) = \left(1 - s_v\left(\mu_v, \Delta_v\right)\right) s_\theta\left(\theta^s, g\right) s_d\left(M_D\left(\mathbf{x}\right)\right) + s_v\left(\mu_v, \Delta_v\right) s_\theta\left(\theta^d, g\right) s_d\left(M_D\left(\mathbf{x}\right)\right).$$

An unassigned person who has a *person-to-group* probability above a threshold, $s^m_{p-g} > \lambda^m_{p-g}$, will be merged into the comparison group. The values of model-based parameters $\mathbf{p}_{p-g} = \left(u_d, \sigma_d, \tau, c, u_v, \sigma_v, \lambda^m_{p-g}\right)$ are different to those used to generate the *pairwise* probability. They will be defined in Section 6.3.2.

To generate the *person-to-group* probability using logistic regression, the same features, $\mathbf{f}_{p-g}$, as the model-based method are used as input. Unlike the logistic classifier used to generate the *pairwise* probability, standard logistic regression can be performed on $\mathbf{f}_{p-g}$, as the features are order-independent. The classifier output, denoted as s^l_{p-g}, is the probability that an unassigned person is interacting with an existing group. If this value is greater than a threshold, $s^l_{p-g} > \lambda^l_{p-g}$, then the person and group are merged. The value of λ^l_{p-g} can be generated in the same way as $\lambda^l p - p$ for the pairwise logistic classifier.

The result of both the model-based method and logistic classifier is to generate an amended list of groups in the scene, whose definition may have changed as a result of unassigned people being merged into them, and a list of the remaining unassigned people.

6.2.4 Recursive Merging Algorithm

Section 6.2.2 defined a method by which all pairwise groups in the scene could be detected. Section 6.2.3 defined a method by which people unassigned to a group could be merged into an existing group. The complete group detection algorithm, as shown in Figure 6.3, involves generating initial pairwise groups, then recursively merging unassigned people into existing groups, until no more merging takes place. Algorithm 2 formally defines this process.

Algorithm 2 Overall group detection algorithm, detectGroups, which uses a recursive process to merge unassigned people into existing groups. getPairwiseGroups is the function described by Section 6.2.2 and mergePeopleIntoGroups is the function described by Section 6.2.3.

```
Require: $P$                                        ▷ A list of people not assigned to a group
Require: $G$                                                   ▷ A list of existing groups
  function recursiveMerge($P$, $G$)
      $P_u$:                                ▷ An updated list of people not assigned to a group
      $G_m$:                                             ▷ An updated list of merged groups
      $P_u$, $G_m$ ← mergePeopleIntoGroups($P$, $G$)
      if $G_m = G$ then
          return $P$, $G$
      else
          return recursiveMerge($P_u$, $G_m$)
      end if
  end function

Require: $P$:                                               ▷ A list of people in the scene
  function detectGroups($P$)
      $P_u$:                                        ▷ A list of people not assigned to a pair
      $G_m$:                                          ▷ A list of merged pairwise groups
      $P_u$, $G_m$ ← getPairwiseGroups($P$)
      return recursiveMerge($P_u$, $G_m$)
  end function
```

6.3 Results

To evaluate group detection performance, datasets of natural groups were recorded using the robot pictured in Figure 6.2. Over twenty minutes of data was collected, of grouped and

Table 6.1: Categories of groups recorded for use as positive samples when evaluating group detection methods.

Group Size	Movement	Frames	Length (s)
2	Static	982	33
2	Dynamic	1806	60
3	Static	2904	97
3	Dynamic	433	14
4	Static	2301	77
	Total	8426	281

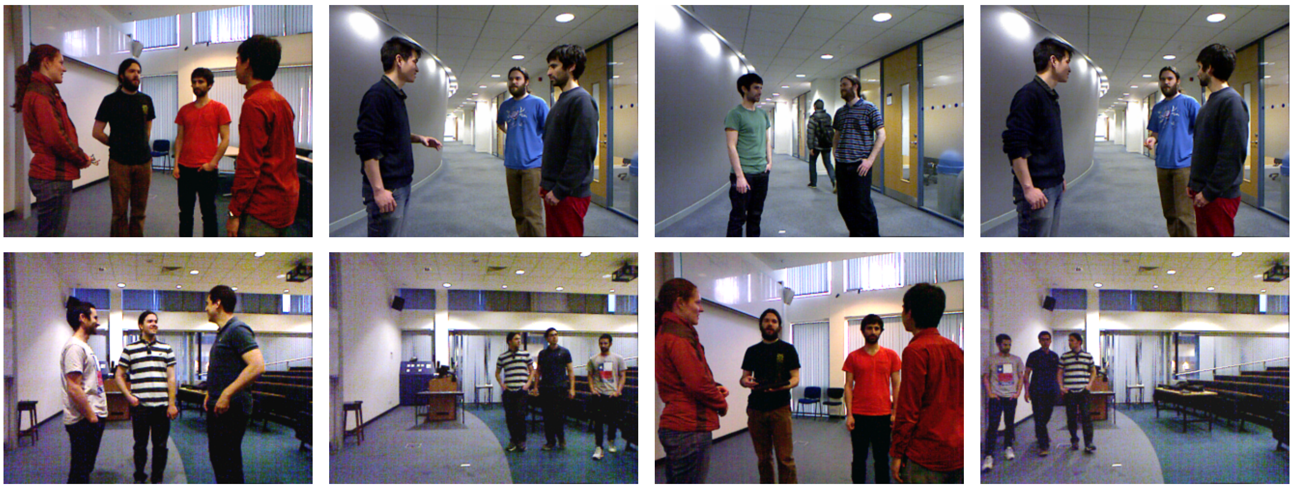


Figure 6.10: Images of static and dynamic groups used as positive samples for the group detection methods.

ungrouped persons. Static and dynamic groups with two, three and four participants were recorded, for use as positive samples. A breakdown of group categories with corresponding numbers of frames is detailed in Table 6.1. Dataset recordings were made at a 30 Hz rate. Images from the datasets shown in Figure 6.10. As negative samples, ungrouped people were recorded, in various scenarios. These are documented in Table 6.2, with some corresponding images shown in Figure 6.11.

As can be seen from the images, there is no quantifiable boundary between people who are grouped and ungrouped. Groups in the datasets were formed naturally by the participants. The task of the group detection methods is to replicate this subjective human judgement of what constitutes a group, and what does not.

Sections 6.3.1 and 6.3.2 will use these datasets to evaluate the performance of the logistic

Table 6.2: Categories of ungrouped persons recorded for use as negative samples when evaluating group detection methods. The categories were selected to test both the pairwise group detection and person to group merging stages.

People	**Dataset Description**	**Frames**	**Length (s)**
2	2 Static Ungrouped	15677	523
2	2 Dynamic Ungrouped	776	26
2	1 Static Ungrouped, 1 Dynamic Ungrouped	2975	99
3	2 Static Grouped, 1 Static Ungrouped	989	33
3	2 Static Grouped, 1 Dynamic Ungrouped	873	29
3	2 Dynamic Grouped, 1 Static Ungrouped	1468	49
	Total	22758	759

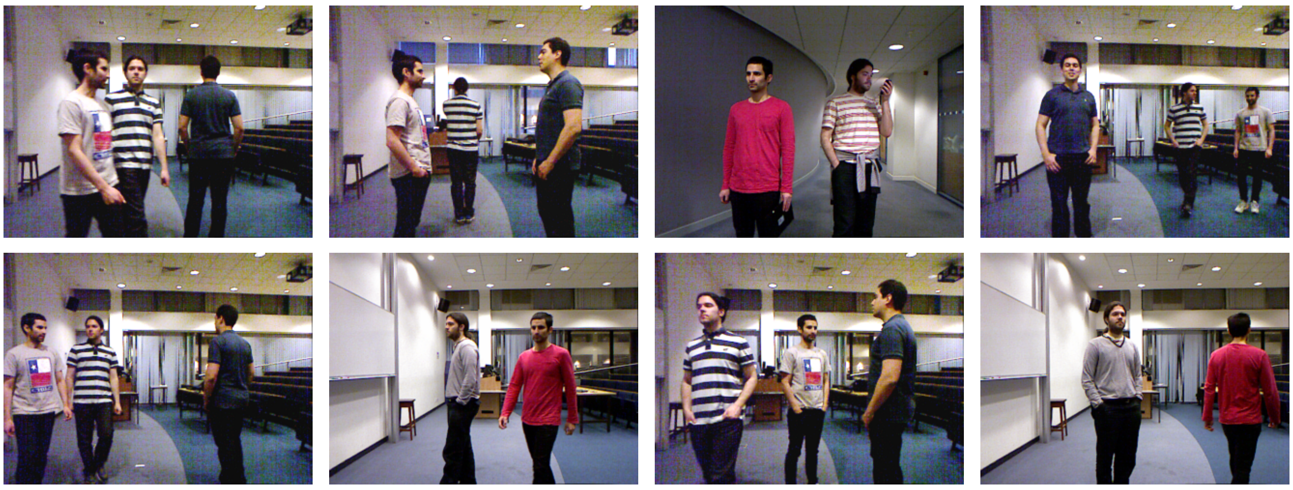


Figure 6.11: Images of static and dynamic ungrouped persons used as negative samples for the group detection methods. Note that in order to evaluate the person to group merging stage, some images show scenes where a group is present, as well as an ungrouped person.

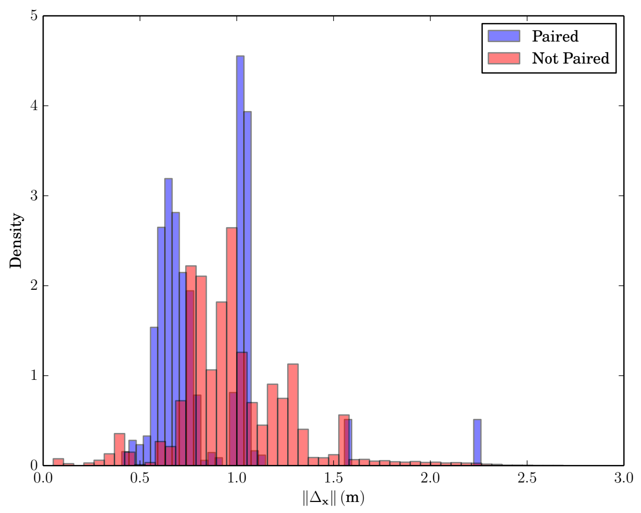


Figure 6.12: Normalised histograms for distance feature values, $\|\Delta_\mathbf{x}\|$, between two people, both for when they are interacting as a pair, shown in blue, and when they are not, shown in red. As can be seen, the distance feature alone does not provide enough information to separate the two classes.

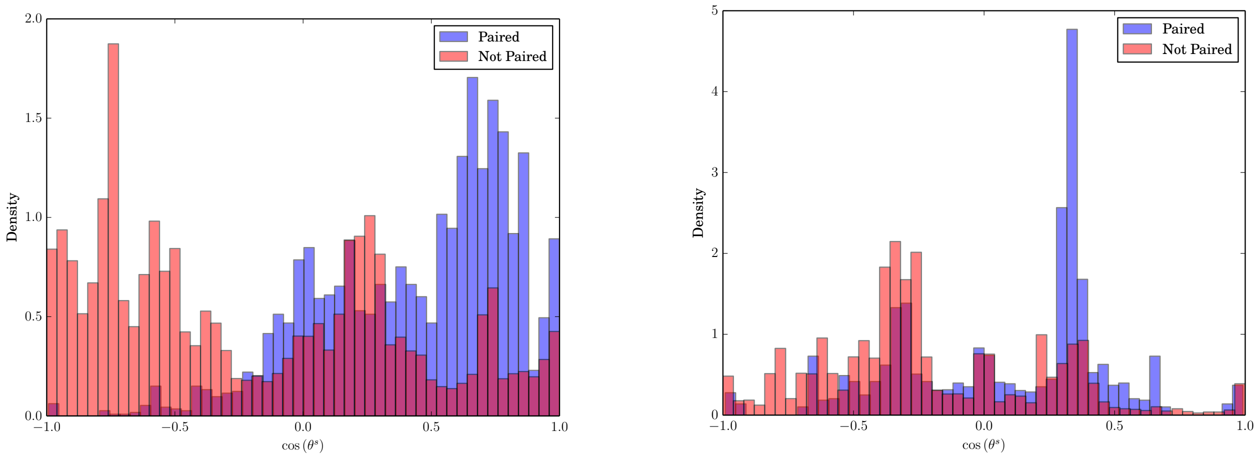
classifier and model-based method on these datasets. Section 6.3.3 will then evaluate the logistic classifier on publicly available datasets and compare the results with the state-of-the-art.

6.3.1 Feature Evaluation

Before detailing the group detection results, an analysis will be performed of the group classification capacity of the main pairwise features, $\mathbf{f}_{p-p}$. This analysis will use all the data from two person groups, described in Tables 6.1 and 6.2, incorporating both the positive samples of paired people, and the negative samples of ungrouped people. All presented histograms are normalised so that the sum of their elements adds up to 1.

Figure 6.12 shows a histogram of the Euclidean distance between two people, $\|\Delta_\mathbf{x}\|$, for both positive pairs of people and negative unpaired people. As can be seen, the distance feature does not well separate the two classes. However, in general, pairs of people have a closer separating distance than their unpaired counterparts.

Figure 6.13 shows equivalent histograms for the static gaze feature, $\cos(\theta^s)$. Figure 6.13a shows values of $\cos(\theta^s)$ when the gaze detector successfully detected a person's gaze. Figure 6.13b



(a) Detected Gazes (b) Non-Detected Gazes

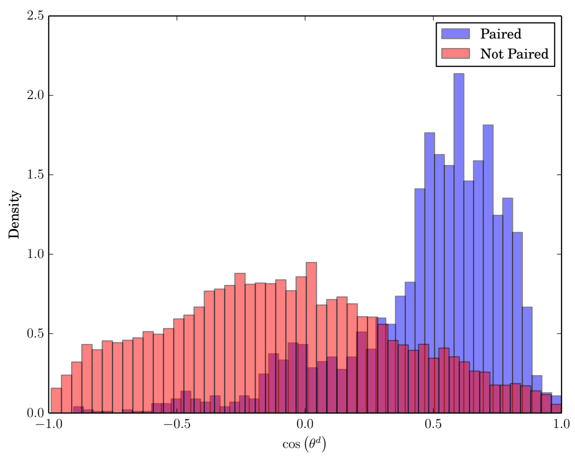
Figure 6.13: Normalised histograms of the static gaze feature, $\cos(\theta^s)$, for when the gaze detector captures a person's gaze, (a), and when it fails to detect a person's gaze, (b). When the gaze detector fails, a person is assumed to be looking directly away from the camera. Good class separation is achieved when a person's gaze is detected.

shows values of $\cos(\theta^s)$ when the gaze detector fails to detect a person's gaze. As mentioned previously, when the gaze detector fails, a person's gaze is assumed to be directly away from the camera. This is because the gaze detector fails when both eyes are not visible, and in this situation it can be assumed that the person is looking away from the camera. The assumed gazing direction, directly away from the camera, is the average of all gazing directions where both eyes are not visible.

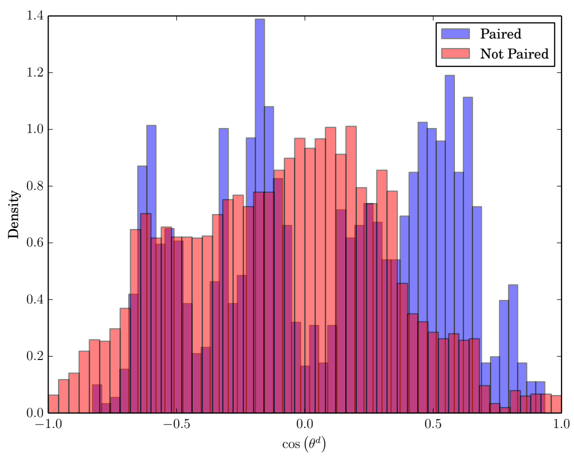
With gazes successfully detected by the gaze detector, Figure 6.13a shows that $\cos(\theta^s)$ values can separate paired and unpaired people well. As expected, pairs of people have $\cos(\theta^s)$ values approaching 1, meaning their gaze is close to the other person. Conversely, unpaired people have $\cos(\theta^s)$ values closer to -1, indicating that they are looking away from the other person.

When the gaze detector fails to detect a person's gaze, Figure 6.13b shows that $\cos(\theta^s)$ values appear to give no informative information. The values for people who are in pairs, and those who are not, are very similar. This motives the introduction of the feature, $g_i \in \mathbf{f}_{p-p}$, so that the failure of the gaze detector can be explicitly specified.

Figure 6.14 shows histograms of the dynamic gaze feature $\cos\left(\theta^d\right)$. Figure 6.14a shows values of $\cos\left(\theta^d\right)$ when the gaze detector successfully detected a person's gaze. Figure 6.14b shows



(a) Detected Gazes



(b) Non-Detected Gazes

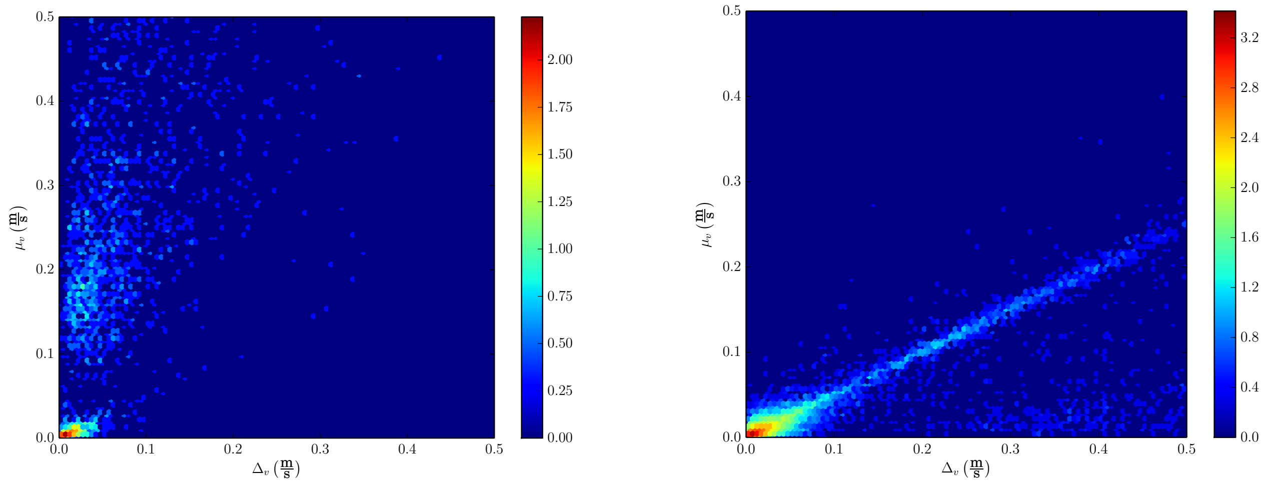
Figure 6.14: Normalised histograms of the dynamic gaze feature, $\cos\left(\theta^d\right)$, for when the gaze detector successfully detects a person's gaze, (a), and when it fails, (b). Like the static gaze feature, good class separation can be achieved with $\cos\left(\theta^d\right)$ values when a person's gaze is successfully detected.

values of $\cos\left(\theta^d\right)$ when the gaze detector fails to detect a person's gaze.

Similar to the static gaze feature, when the gaze detector successfully detects a person's gaze, $\cos\left(\theta^d\right)$ provides good information for classifying interacting and non-interacting pairs, as shown in Figure 6.14a. Paired people have $\cos\left(\theta^d\right)$ values close to 1, meaning their detected gaze is close to the expected direction. Unpaired people have a much wider distribution of $\cos\left(\theta^d\right)$ values. As expected, this indicates more variation in the gaze of ungrouped people.

Figure 6.15 shows normalised 2D histograms for the two velocity features, μ_v and Δ_v. Figure 6.15a shows a 2D histogram for pairs of people, whilst Figure 6.15b is a 2D histogram for unpaired people.

The high density of samples at the origin of both histograms represents features derived from static participants. For moving participants, a clear separation can be seen between paired and unpaired people, using both velocity features. Paired people have a low velocity difference, Δ_v, with a varying mean velocity, μ_v. Unpaired people, on the other hand, have Δ_v values with much greater variance.



(a) Paired (b) Unpaired

Figure 6.15: Normalised 2D histograms for the two velocity features, μ_v and Δ_v. Features from pairs of people are shown in (a). Features from unpaired people are shown in (b). The large number of samples at the origin of both graphs represents static subjects. Outside this area, the two histograms are visually separable. This suggests that, for moving subjects, accurate classification of interactions can be made using both features.

6.3.2 Results Analysis

In the following results, random subsampling cross-validation was used during classifier training. For each one of 10 random splits, 40% of the data was used as training data, and the remaining 60% was used for testing. The final classification accuracy, displayed in the following ROC curves and accuracy tables, was the average performance over all random splits. To generate the ROC curves for the logistic classifier, the threshold parameters λ^l_{p-p} and λ^l_{p-g} are varied for *pairwise* and *person-to-group* detection respectively. The equivalent parameters for the model-based method are λ^m_{p-p} and λ^m_{p-g}.

For each random split of the training data, the pairwise group detection methods from Section 6.2.2 were trained on groups of size 2. Scenarios with 3 people were used to train the person to group merging methods from Section 6.2.3. To fit the regression coefficients of the logistic classifier, maximum likelihood estimation was used [151]. When determining the optimal model-based parameters, $\mathbf{p}_{p-p}$ and $\mathbf{p}_{p-g}$, for a particular training dataset, the Nelder-Mead simplex algorithm [153] was employed.

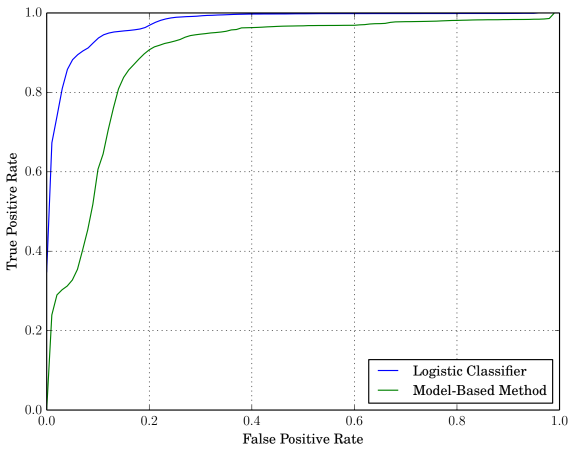
To accomplish this, an error function is created that takes as input: the training features, $\mathbf{f}_{p-p}$ or $\mathbf{f}_{p-g}$, their corresponding ground truth values, and a set of current parameter values, $\mathbf{p}_{p-p}$ or $\mathbf{p}_{p-g}$. The function output is the total number of positive and negative training misclassifications produced.

The Nelder-Mead method is based around a simplex that lies in the 7 dimensional parameter space, centred around an initial estimate of the function minimum. The vertices of the simplex are transformed, based on their function values, until they finally contract on the function minimum. This method was chosen as it does not require the derivatives of the model-based functions. Because of their parameterised piecewise limits, calculation of the function derivatives would be non-trivial. Sensible initial values were provided, based on sociological information and human knowledge of group definitions: $u_d = 1.20, \sigma_d = 0.50, \tau = 4.0, c = 0.16, u_v = 0.50, \sigma_v = 0.25, \lambda^m = 0.001$.

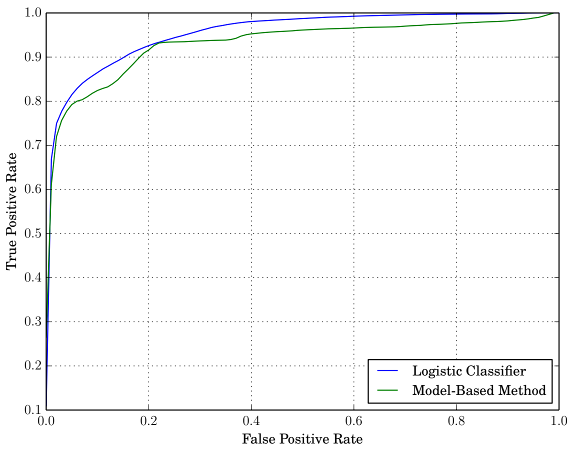
Figure 6.16 displays ROC curves comparing the results of the model-based method and the logistic classifier. The two methods were separately trained and tested for both pairwise group detection, Figure 6.16a, and person to group merging, Figure 6.16b. Whilst both methods can be see to provide adequate group detection accuracy, the logistic classifier has improved performance in both detection tasks. In particular, very high pairwise group detection accuracy is achieved with the logistic classifier.

In explaining the higher performance of the logistic classifier, the analysis of the group detection features from Section 6.3.1 shall be referred to. Specifically, the combination of μ_v, Δ_v, $\cos(\theta^s)$ and $\cos\left(\theta^d\right)$ can be seen to provide good class separation. The model-based method provides a group detection algorithm that is based on general sociological principles. The logistic classifier however, constructs a group detection method based entirely on the underlying data. As a consequence of the employed features providing robust group information, the logistic model of the data is very accurate. Additionally, in all experiments, cross-validation was used to ensure that the optimal logistic model is not overfitted, and is capable of performing in a wide range of scenarios.

An additional reason for the logistic classifier's higher performance is the large amount of



(a) Pairwise Group Detection



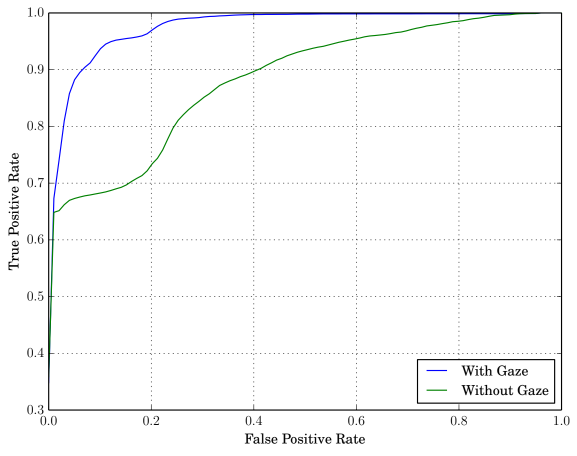
(b) Person to Group Merging

Figure 6.16: Figure showing ROC curves for both the logistic classifier and the model-based method. (a) shows pairwise group detection results. (b) shows results for the person to group merging stage. The logistic classifier has superior performance for both detection stages, especially pairwise group detection.

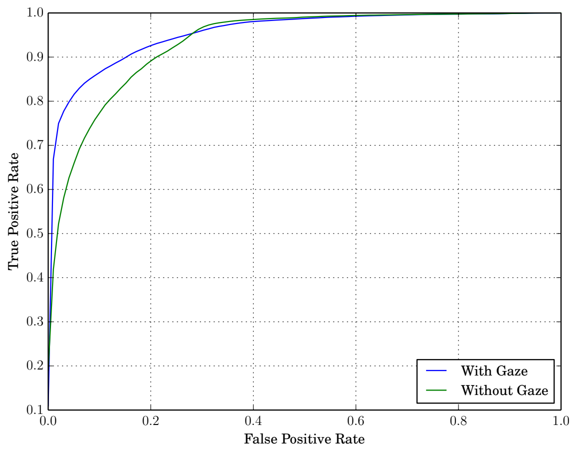
training data available. As the model-based method is constructed from human knowledge of group definitions, only a small amount of training data is needed to fine-tune the model parameters. The logistic classifier has a large amount of training data available to construct a group detection model that is sufficiently general, but tailored to the provided dataset. With less training data available, the accuracy and generality of the logistic model would be reduced. The model-based method would not be affected however, as it is not constructed from the underlying data.

To evaluate the importance of the gaze detector to the group detection algorithm, a modified feature vector was implemented that contains no gaze information. The gaze-less features used to generate the *pairwise* interaction probability are thus: $\mathbf{f}_{p-p}^{\neg g} = \{\|\Delta_{\mathbf{x}}\|, \mu_v, \Delta_v\}$. Similarly, the gaze-less features used to generate the *person-to-group* probability are: $\mathbf{f}_{p-g}^{\neg g} = \{M_D(\mathbf{x}), \mu_v, \Delta v\}$. These features were extracted from all datasets using the same number of cross-validation splits, and training and test ratios, as before.

A logistic classifier for detecting interacting pairs was trained using $\mathbf{f}_{p-p}^{\neg g}$; a corresponding person-to-group merging classifier was trained using $\mathbf{f}_{p-g}^{\neg g}$. Figure 6.17 displays the decrease in performance when compared to the logistic classifiers that use gaze information. As can



(a) Pairwise Group Detection



(b) Person to Group Merging

Figure 6.17: ROC curves that present logistic classification results with, and without, information from the gaze detector. (a) shows pairwise group detection results. (b) shows person to group merging results. In both cases, the inclusion of gaze information greatly increases the group detection results.

be seen, the inclusion of gaze estimates greatly increases the results of both pairwise group detection, Figure 6.17a, and person to group merging, Figure 6.17b. This clearly indicates the importance of the gaze detector in the group detection process.

In explaining the classification improvements that gaze information provides, the analysis of the group detection features from Section 6.3.1 shall again be referred to. Paired and unpaired people were shown to have overlapping $\|\Delta_{\mathbf{x}}\|$ values. Indeed, the more crowded the environment, the less information $\|\Delta_{\mathbf{x}}\|$ will provide when separating groups. Figure 6.15 shows that static paired and unpaired people cannot be clearly separated by the velocity features, μ_v and Δ_v. Thus, knowing a static person's gaze direction is essential for accurately classifying whether or not they are in a group, especially in crowded environments.

Table 6.4 further details the performance difference between the logistic classifier, with and without gaze information, and the model-based method. When generating the results of each of the 10 cross validation splits, each of the three tested method's thresholds was chosen to give a false positive rate of 0.1. Equivalent to the ROC curves, the values quoted in Table 6.4 are the average accuracy over all 10 splits. To test the methods' ability to scale to larger groups, training data did not include samples from four person groups. For this case, each method's

Table 6.3: Group detection accuracy for the logistic classifier, with and without gaze information, and the model-based method. As shown, for both the pairwise group detection, and the people to group merging classifiers, the threshold of each algorithm was set to produce a false positive rate of 0.1. The best results are emboldened, and are produced by the logistic classifier using gaze information.

Ground Truth	**Logistic Accuracy** (%)	**Model-Based Accuracy** (%)	**Logistic Accuracy Without Gaze** (%)
	Pairwise Group Detection		
Pairs	**93.59**	59.75	68.26
Non-Pairs	90.00	90.00	90.00
	Person to Group Merging		
Merge	**86.50**	82.39	77.33
Non-Merge	90.03	90.03	90.02

training parameters were set to the average of those generated in each cross validation split. These optimised parameters for the model-based pairwise group classifier were: $u_d = 0.30, \sigma_d = 0.40, \tau = 8.70, c = 0.10, u_v = 0.69, \sigma_v = 0.25, \lambda^m_{p-p} = 0.001$. Those for the person-to-group merging classifier were: $u_d = 0.69, \sigma_d = 0.44, \tau = 4.63, c = 0.17, u_v = 0.55, \sigma_v = 0.27, \lambda^m_{p-g} = 0.001$.

The tabular results mirror those of the ROC curves in Figures 6.16 and 6.17: The logistic classifier has better accuracy than the model-based method for pairwise group detection, and the use of gaze information improves the accuracy of both group detection stages. To further justify the stated explanation for this, Table 6.4 breaks down the results into the separate datasets categories of positive group samples.

It can be seen that both logistic classifiers detect dynamic groups with a high accuracy. However, when gaze information is removed, static group detection reduces dramatically. These results corroborate the explanation that velocity features, μ_v and Δ_v, well separate dynamic groups, but gaze is required for static group detection. The poor performance of the model-based method for the group of 4 persons, is explained by the fact that this dataset was not used in any training data. For this 4 person dataset, 22.64% of frames had a group of size 3 detected using the model-based method. The pairwise model-based parameters, $\mathbf{p}_{p-p}$, were thus probably overfitted to the training datasets of paired people, and did not allow for the

Table 6.4: A breakdown of the group detection accuracy for the categories of groups recorded as positive samples. Results are displayed for the logistic classifier, with and without gaze information, and the model-based method. The threshold of each algorithm was set to produce a false positive rate of 0.1. The best results are emboldened, and are mainly produced by the logistic classifier using gaze information.

Group Size	**Movement**	**Logistic Accuracy** (%)	**Model-Based Accuracy** (%)	**Logistic Accuracy Without Gaze** (%)
2	Static	**85.64**	46.28	14.34
2	Dynamic	**97.90**	67.08	97.77
3	Static	85.00	**85.12**	74.50
3	Dynamic	**96.46**	64.08	96.19
4	Static	**79.66**	0.78	29.99

increased spread of the 4 person group formation.

6.3.3 Results in Crowded Environments

To assess group detection performance in crowded, unconstrained environments, the robot was placed outside a lecture hall, recording people as they leave. As people tend to form natural groups as they exit lectures, this dataset is used to demonstrate the proposed group detection method's real-world practicality.

Figure 6.18 shows the results of the group detection algorithm, using the logistic classifier. Figures 6.18d and 6.18e show a person looking down at their phone and notes. The subject's gaze is thus not detected, and they are not merged with the group in front of them. In Figure 6.18f one of the subjects then looks up, towards the direction of this group. His detected gaze, and corresponding features, result in him correctly being merged into the group.

As part of a collaborative work on an unpublished paper [154], the results of the logistic group detector were compared against published state-of-the-art results using four external datasets. Two "CoffeeBreak" datasets from Cristani et al. [8] were used, each 2 minutes long, with 91 and 75 frames of manually annotated ground truth groups, respectively. Additionally, a "CocktailParty" dataset from Setti et al. [9] was used, containing 320 frames of ground truth groups, which were manually annotated every 5 seconds over a 30 minute video. Images from

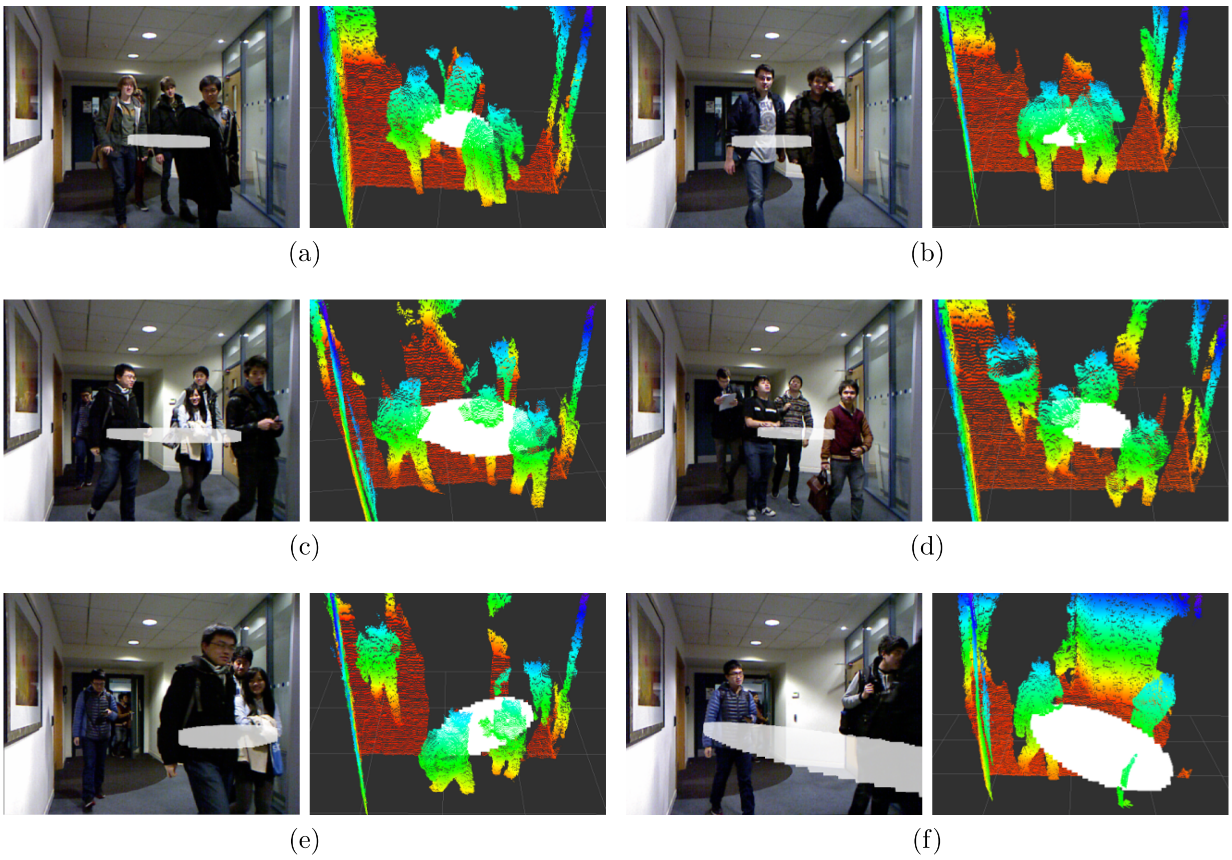
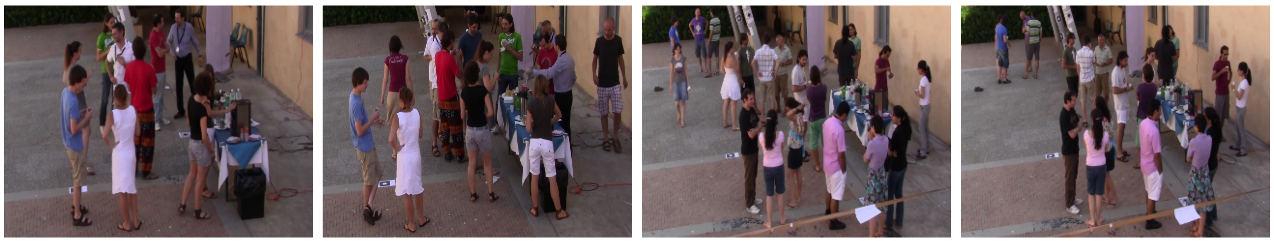


Figure 6.18: Images of group detection results, using logistic classification, on a recording of students exiting a crowded lecture theatre. Each subfigure has an RGB image and a rotated point cloud from the same time instance. A white ellipse is overlaid on the detected groups, so that the the outermost participants lie on its edges. Note that in (d) and (e), the use of the gaze detection method from Chapter 5 means that the subjects looking down are correctly separated from the nearby groups. When one of the subjects then looks at the people in front of him, shown in (f), he is then included in the group.

the datasets are shown in Figure 6.19. Each dataset contains the positions of people in the scene, their gaze orientation, and ground truth annotated groups. Comparison results were also generated for a synthetic dataset from [8], containing 100 frames of ground truth annotated people positions, orientations and group membership.

For each dataset, a single random split of 40% of the frames were used to train the logistic classifier with the remaining 60% used to evaluate it. The pairwise classifier was trained only on groups of size 2, whilst the people to group merging classifier was trained on all permutations of larger groups. Using metrics from [9], a group of size N is correctly detected if it contains at least $\frac{2}{3}N$ correct members, and no more than $\frac{1}{3}N$ false participants are included. Correctly


(a) “CoffeeBreak” Scenarios

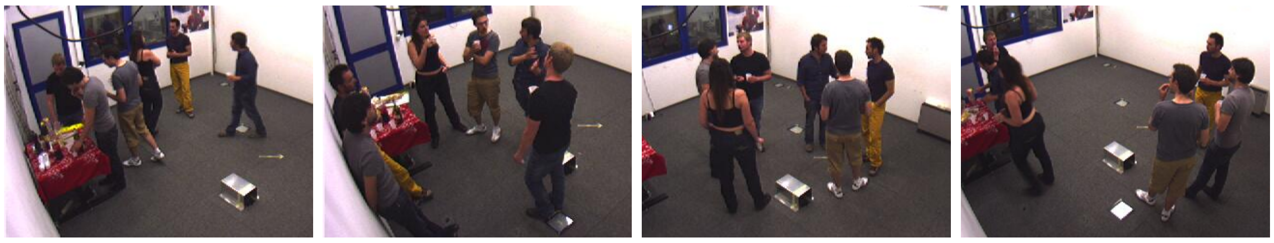

(b) “CocktailParty” Scenario

Figure 6.19: Images from publicly available group detection datasets, used to compare the logistic group classifier against the state-of-the-art. The “CoffeeBreak” scenario, (a), was published in [8], and the “CocktailParty” scenario, (b), was published in [9]. All images kindly supplied by the authors.

detected groups are denoted as true positives, TP, mistakenly detected groups as false positives, FP, and missing groups are denoted as false negatives, FN. Precision and recall can then be defined as:

$$\text{precision} = \frac{TP}{TP + FP} \qquad \text{recall} = \frac{TP}{TP + FN} \tag{6.9}$$

The results of four different methods, using these datasets, were presented in [9]. Table 6.5 compares these presented results with that of the proposed group detection method. As shown, the proposed logistic group detector gives a very high precision and recall for all datasets except the “CocktailParty” scenario.

The accurate results of the “Synthetic” and “CoffeeBreak” scenarios can be attributed to the method’s descriptive features, extracted from the training set. These features generalise well in describing the groups found in the test data, allowing for accurate group classification. In comparison, the advantage of the competing methods is that any internal parameters are set using prior knowledge. Thus, they dont require explicit training data. An additional reason for the high performance in the “CoffeeBreak” scenario is due to its short length, which reduces

Table 6.5: Table comparing the precision and recall of the proposed method to four state-of-the-art techniques, using four publicly available datasets. The largest values in each category are emboldened. The logistic group detector gives accurate results for all but one of the datasets.

Method	**Synthetic**		**CoffeeBreak 1**		**CoffeeBreak 2**		**CocktailParty**	
	Prec.	**Rec.**	**Prec.**	**Rec.**	**Prec.**	**Rec.**	**Prec.**	**Rec.**
Proposed	**0.98**	**0.98**	**0.81**	**0.89**	0.92	**0.99**	0.31	0.33
Multi-Scale [9]	0.86	0.94	0.63	0.76	**0.94**	0.78	**0.69**	**0.74**
Cristani et al. [8]	0.73	0.83	0.42	0.59	0.58	0.49	0.59	**0.74**
Dominant Sets [149]	-	-	0.62	0.54	0.72	0.71	-	-
IRPM [150]	0.71	0.54	0.63	0.54	0.55	0.19	0.50	0.46

the variance between the training and test sets.

The poor performance in the "CocktailParty" scenario can be attributed to several factors. Some groups found in this dataset have arrangements very different to the normal F-formation. The proposed group features were based on the sociological principles defined in Section 6.2.1. If these are heavily contravened by the ground truth, performance will suffer. For example, one ground truth group consisted of a cluster of three people in an F-formation and an extra two people in the "R"-space. These outlying people are positioned at 1.5 m and 2 m from the F-formation centre, with one looking away from everyone else. The subjectivity of including the extra two people in the group could thus be argued, but the results of Table 6.5 do not take this into account. An additional reason for the poor performance in the "CocktailParty" dataset is that people's gazes are estimated using a visual-based method, which can give coarse gaze granularity. The proposed method, however, was designed around the availability of accurate gaze estimates, such as provided by the sensor-fusion method of Chapter 5. As the "CocktailParty" scenario has different sensor inputs than the system was designed for, this could account for the reduced performance.

6.4 Conclusions

The contribution of this chapter has been a method of detecting static and dynamic groups, which incorporates real-time gaze estimates to enhance gaze detection accuracy. Group detection is an important aspect of HRI, enabling robots to understand social context in human environments. Many previous group detection methods do not explicitly account for dynamic groups. However, it was shown that group arrangements vary drastically between static and dynamic interactions, necessitating distinct solutions to classify them. To account for this, a novel set of features was defined, based on sociological concepts. These features were extracted from person displacement, velocity and gaze information. The importance of accurate gaze estimates was highlighted when detecting static interactions.

Gaze estimates were obtained through the method in Chapter 5 with position and velocity information obtained from the people detection method of Chapter 3. In using this information, the proposed group detector firstly identifies pairs of interacting people in a scene. A subsequent recursive algorithm is used to merge unassigned people into existing groups, completing the detection process. Two classification methods were evaluated: a model-based method, which uses a combination of likelihood functions to estimate interaction probability, and a logistic regression model.

Classifier performance was compared using more than twenty minutes of recorded groups, of various sizes and at various speeds. A detailed analysis of features chosen for group classification was presented, along with their suitability for static and dynamic groups. Logistic regression was shown to classify groups more accurately than the model-based method. Due to the robust features employed, performance compared favourably to four state-of-the-art methods, using four external datasets.

Group detection has many important applications in HRI. One of the most promising is that of socially aware mobile robot navigation. An introductory work in this topic was carried out [13], using the group detection method from this chapter. Detected group were assigned a shared interaction space, which a moving robot should not enter. Using this information, a navigation algorithm was devised that would find alternate paths around a detected group's interaction space. In this way, a moving robot is better able to conform to human social norms when navigating an environment. To evaluate the proposed navigation framework, scenarios were recorded where a group lay between the robots initial pose and the goal. It was shown that a commonly used trajectory planner, with standard obstacle avoidance, frequently violated the group's shared interaction space. However, using the group-aware navigation algorithm, the group was mostly undisturbed by the robot moving to its goal.

Chapter 7

Conclusions

The contribution of this thesis has been to provide mobile robots with sensing methods, which are capable of operating in crowded and dynamic environments, in order to facilitate HRI. Accurate sensing methods for HRI are important, as autonomous robots will play an increasing role in improving quality of life in the future. To cope with problems such as global ageing populations, robot assistants will become a more viable and prevalent solution.

As medical advances give rise to longer life expectancy, it becomes increasingly important to develop methods by which quality of life can be improved. Innovation in mobile robotics holds much untapped potential for continuous monitoring of the elderly, providing assistive care, and managing chronic diseases. Although HRI has been the focus of much recent research, the deployment of a robot capable of seamless operation in crowded, human-centric environments, has still not been realised. It is this problem that has been investigated in this thesis, as it is a notable hurdle for the adoption of mobile robots in healthcare environments.

Effective HRI techniques allow a robot to understand the intentions of humans in an environment, and act accordingly. As has been shown, there are many research areas in this multidisciplinary field. This thesis has specifically addressed the HRI problems of detection and tracking of people and their associated hands, human attention detection through gaze estimation, and group identification. Many existing methods to these problems have been developed in the context of general computer vision. However, their suitability for HRI is reduced by such research challenges as occlusions from potentially distant subjects in crowded environments, requirements of computational efficiency, dynamics of moving people with unconstrained pose, a moving camera and illumination changes. The HRI methods presented in this thesis used multiple input sensors to mitigate problems caused by these research challenges, thereby furthering the state-of-the-art.

The main limitations of the work presented in this thesis come from sensor limitations, specifically the accuracy of the Microsoft Kinect depth camera. It was found that reliable depth values were only obtainable within distances of approximately 1.2 m to 2.5 m. At further distances, pixels near the edges of objects tend to exhibit a lot of noise, and frequently have no obtainable depth information. The SwissRanger SR3000 time-of-flight depth camera was evaluated as an alternative method. Whilst its depth values were more accurate than those of the Kinect, it's low resolution of 144×176 pixels and poor performance on a moving robot reduced its applicability for HRI. Additionally, the presented methods were not tested in outdoor environments, as the performance of both depth cameras is negatively affected by bright sunlight.

Due to the narrow nature of the fingers, resultant noise from edge pixels in the depth camera significantly alters the hand appearance at far distances. As such, hand detection rates in Chapter 3 were greatly reduced after 2.5 m. Depth camera noise was also the limiting factor in the range of the gaze estimation method of Chapter 5. Principally, the forehead estimation method, described in Section 5.2.3, displayed poor performance at distances further than 2.5 m. Whilst the employed infrared LEDs were found to illuminate pupils accurately at over 3 m, in the absence of depth camera noise they would also limit the maximum range of the system.

In both chapters, a more accurate depth camera would overcome these problems. In July 2014, a new "Kinect v2" sensor was released by Microsoft, which promises greater depth accuracy at further distances. If it is shown to have good performance on a moving robot, it could easily increase the detection rates and maximum range of both the hand and body detectors from Chapter 3 and the gaze estimation method of Chapter 5.

7.1 Technical Contributions of Thesis

Four main contributions were presented in this thesis.

Chapter 3 presented a hand and body association algorithm designed for crowded and dynamic environments. Results of the hand detector highlighted its ability to recognise unseen gestures and subjects in a crowded environment. This is due to the use of geodesic distances in isolating information local to the hand. Results of the proposed body detector showed comparable performance to state-of-the-art visual methods, but at a fraction of the computational cost. Finally, it was shown that the hand-body association algorithm effectively used past association outcomes to reduce transient misassociations, due to noise or misdetections.

Chapter 4 presented an RGB-D sensor-fusion algorithm for tracking hands in crowded and dynamic environments. Tracking results in crowded environments demonstrated that the RGB update step successfully compensated for misdetections from the depth-image based hand detector, caused by sensor noise, occlusions and the highly articulate nature of the hand. Due to the RGB update step's efficiency, velocity reconstruction of a gesturing subject was shown to have increased accuracy.

Chapter 5 proposed a sensor-fusion gaze estimation system that combined infrared and depth cameras on a mobile robot. The sensor-fusion techniques allowed the calibration-free method to provide better accuracy than traditional visual methods, without the constraints of traditional IR methods. This was proved experimentally under multiple conditions: using a static subject, using multiple moving people and using a moving robot. Unlike traditional IR techniques, the proposed method is not limited by the number of people in a scene and has a long range: accurate gaze estimates were achieved between 1.2 m and 2.5 m.

Chapter 6 presented a group detection algorithm that uses gaze estimates to enhance the detection accuracy of static and dynamic interactions. Existing sociological theory was used to extract a novel set of features for detecting static and dynamic groups. This was validated experimentally, where gaze-based information was shown to be especially important in static group detection. Using these robust features, a logistic regression classifier was shown to give better performance than a pre-defined model-based method.

7.2 Future Work

This thesis has presented a number of mobile-robot sensing methods to facilitate HRI. These methods have numerous potential applications, providing most benefit in crowded environments, due to the difficulties posed for many existing techniques. However, the work in this thesis can be extended and, to compensate for previously highlighted limitations, its results improved in many ways.

One problem common to virtually all HRI sensing methods, including those proposed in this thesis, is that a user must be continuously in the field of view (FOV) of all cameras during interaction. The Microsoft Kinect has a 57° horizontal FOV, which is already larger than many of its competitors. With modern navigation systems, robots tend to make quick horizontal rotations. Thus, not only does a user have to be in front of the robot for the sensing methods to work, but any interaction attempt can be cancelled by somewhat unpredictable robot motion. Additionally, as the methods proposed in this thesis have a maximum range of 2.5 m, the user will have to be relatively close to the robot to initiate interaction.

This problem could be solved using active vision, which would involve manipulating the robot's location in order to better view salient information. This would require the incorporation of additional sensors, such as a microphone array or an omni-directional camera. For example, a user could audibly call to a nearby robot, which would use acoustic source localisation to detect where the sound is coming from. Alternatively, the user could perform a predefined gesture, recognised and located using the robot's omni-directional camera. In both examples, the robot would then navigate to the detected person and continue interaction using its conventional sensors, and the methods proposed by this thesis.

One of the most significant results of this thesis is the sensor-fusion hand and body association system from Chapters 3 and 4. Despite this system having many potential HRI applications, none were explored in this thesis. A gesture recognition module would greatly complement the presented work, allowing users to command and control the robot. One of the main limitations of existing gesture recognition systems is the hand tracking component. As highlighted in Section 2.4, most methods rely on simple techniques, such as monocular skin colour detection. This frequently imposes restrictions such as a static camera, single-user operation, or uncluttered environments. The body pose estimation methods reviewed in Section 2.3.2 would impose their own limitations, such as requiring standing users, if applied to this problem.

By applying the presented sensor-fusion hand and body association system to the problem of gesture recognition, many restrictions of existing methods could be overcome. Special attention should be given to multi-user scenarios, crowded environments and performance on a moving robot. The hand-body association system is uniquely suited for these situations, affording the potential for a large improvement over the state-of-the-art. Using the hand-body association algorithm for gesture input, the most appropriate machine learning algorithm would have to be investigated. An important, related avenue of investigation would be to detect the start and end of a gesture issued via the hand-body association system. Many previous methods discussed in Section 2.4 expect a continuous stream of gestures; this assumption will not hold for real-world HRI. A final area of investigation would be how the robot could separate background gestures issued in human-human interaction from intentional gestures in HRI. The proposed group detection method of Chapter 6 could form a strong basis in solving this problem.

Despite the presented work being evaluated in challenging scenarios, other environments can be identified where HRI could be of more practical use. Specifically, patients could benefit greatly from the introduction of HRI in healthcare environments. Thus, the results presented in this thesis should be further explored by applying them in hospitals and nursing homes. However, these environments introduce specific problems that will require overcoming some of the constraints imposed by the presented methods. For example, it was noted in Chapter 5 that the gaze estimation method assumes an upright person. In order for the robot to be used by patients lying in bed, this restriction will have to be overcome. Additionally, the sociological group definitions used in Chapter 6 assumed healthy, standing people. The suitability of these definitions would have to be re-evaluated for wheelchair users, and those with reduced mobility.

So far, such complicating factors have received little attention in HRI literature.

Each new environment that the proposed methods are evaluated in introduces the prospect of reduced performance due to environmental factors. This is because each method has a number of chosen parameters that may not work well if the assumed environment changes significantly. For example, the optimal blob detection threshold in Chapter 5 could change under large lighting variances, as could the number optimal of colour histogram bins in Chapter 4. The presented results should be further explored in different environments to see how they are affected.

To combat any performance reduction due to manually tuned parameters, a system of adaptive parameter selection could be introduced. This would involve automatically finding optimal parameters for the current situation. The system could be extended to re-optimise parameters if the employed hardware changes. For example, if more computational power were available then the number of samples used in Chapter 4 could be increased, probably leading to improved results. Alternatively, if the employed depth camera was upgraded, depth-based parameters in Chapters 3 and 5 could benefit from re-optimisation.

There are many further research areas that could be combined with the presented work, to form a holistic HRI system of more benefit to its users. The people detector in Chapter 3 does not allow the robot to identify detected people. However, if combined with a face recognition system, the robot would be able to perform such HRI tasks as fetching items for a specific user. Audio recognition could be used to issue natural commands to a robot. A facial expression recognition component would allow a robot to tailor any HRI to a subject's emotional state. This could entail finishing interaction if a user is annoyed, or extending it if they are happy. Future HRI systems will no doubt leverage many such components, to solve problems of ever increasing complexity. It is hoped that the presented work will bring that future a step closer.

Bibliography

[1] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun, “Real-time identification and localization of body parts from depth images,” in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2010, pp. 3108–3113.

[2] 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 single depth images,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, Jun. 2011, pp. 1297–1304.

[3] R. Y. Wang and J. Popović, “Real-time hand-tracking with a color glove,” *Graphics, ACM Transactions on*, vol. 28, no. 3, pp. 63:1–63:8, Jul. 2009.

[4] S. McKeague, J. Liu, and G.-Z. Yang, “Hand and body association in crowded environments for human-robot interaction,” in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2013, pp. 2161–2168.

[5] J. Kovac, P. Peer, and F. Solina, “Human skin color clustering for face detection,” in *EUROCON. Computer as a Tool, International Conference on*, vol. 2, Sep. 2003, pp. 144–148.

[6] S. McKeague, J. Liu, and G.-Z. Yang, “An asynchronous rgb-d sensor fusion framework using monte-carlo methods for hand tracking on a mobile robot in crowded environments,” in *Social Robotics*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2013, vol. 8239, pp. 491–500.

[7] B. Benfold and I. Reid, “Guiding visual surveillance by tracking human attention,” in *Proceedings of the British Machine Vision Conference (BMVC).* BMVA Press, 2009, pp. 14.1–14.11.

[8] M. Cristani, L. Bazzani, G. Paggetti, A. Fossati, D. Tosato, A. D. Bue, G. Menegaz, and V. Murino, “Social interaction discovery by statistical analysis of f-formations,” in *Proceedings of the British Machine Vision Conference (BMVC).* BMVA Press, 2011, pp. 23.1–23.12.

[9] F. Setti, O. Lanz, R. Ferrario, V. Murino, and M. Cristani, “Multi-scale f-formation discovery for group detection,” in *Image Processing (ICIP), IEEE International Conference on*, Sep. 2013, pp. 3547–3551.

[10] E. Wright, “2008-based national population projections for the united kingdom and constituent countries,” *Population Trends*, no. 139, pp. 91–114, 2010.

[11] S. Valibeik, “Human robot interaction in a crowded environment,” Ph.D. dissertation, Imperial College London, 2010.

[12] S. McKeague, J. Liu, A. Vicente, and G.-Z. Yang, “Human gaze estimation using a sensor-fusion based long range infrared eye detector on a mobile robot,” in *Submission*, Feb. 2015.

[13] J. Correa, S. McKeague, J. Liu, and G.-Z. Yang, “A study of socially acceptable movement for assistive robots concerning personal and group workspaces,” in *Proccedings of the 7th Hamlyn Symposium on Medical Robotics*, 2014.

[14] C. Wong, S. McKeague, J. Correa, J. Liu, and G.-Z. Yang, “Enhanced classification of abnormal gait using bsn and depth,” in *Wearable and Implantable Body Sensor Networks (BSN), International Conference on*, May 2012.

[15] C. Wong, Z. Zhang, S. McKeague, and G.-Z. Yang, “Multi-person vision-based head detector for markerless human motion capture,” in *Wearable and Implantable Body Sensor Networks (BSN), International Conference on*, May 2013.

[16] J. Liu, J. Correa, S. McKeague, E. Johns, C. Wong, A. Vicente, and G.-Z. Yang, "A healthcare mobile robot with natural human robot interaction," in *Proccedings of the 5th Hamlyn Symposium on Medical Robotics*, 2012.

[17] R. Simmons, J. Fern, R. Goodwin, and S. Koenig, "Xavier: An autonomous mobile robot on the web," in *IEEE Robotics and Automation Magazine*, 1999, pp. 43–48.

[18] C. Schroeter, S. Mueller, M. Volkhardt, E. Einhorn, C. Huijnen, H. van den Heuvel, A. van Berlo, A. Bley, and H.-M. Gross, "Realization and user evaluation of a companion robot for people with mild cognitive impairments," in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2013, pp. 1153–1159.

[19] J. Ballantyne, S. Valibeik, A. Darzi, and G.-Z. Yang, "Robotic navigation in crowded environments: Key challenges for autonomous navigation systems," in *Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems.* ACM, 2008, pp. 306–312.

[20] K. S. Ong, Y. H. Hsu, and L. C. Fu, "Sensor fusion based human detection and tracking system for human-robot interaction," in *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, Oct. 2012, pp. 4835–4840.

[21] L. Chen, "Pairwise macropixel comparison can work at least as well as advanced holistic algorithms for face recognition," in *Proceedings of the British Machine Vision Conference (BMVC).* BMVA Press, 2010, pp. 5.1–5.11.

[22] M. Enzweiler and D. Gavrila, "Monocular pedestrian detection: Survey and experiments," *Pattern Analysis and Machine Intelligence (PAMI), IEEE Transactions on*, vol. 31, no. 12, pp. 2179–2195, Dec. 2009.

[23] C. Papageorgiou and T. Poggio, "A trainable system for object detection," *Computer Vision (IJCV), International Journal of*, vol. 38, no. 1, pp. 15–33, 2000.

[24] P. Viola, M. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," in *Computer Vision (ICCV), IEEE International Conference on*, vol. 2, Oct. 2003, pp. 734–741.

[25] P. Viola and M. Jones, "Robust real-time face detection," *Computer Vision (IJCV), International Journal of*, vol. 57, no. 2, pp. 137–154, 2004.

[26] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, vol. 1, Jun. 2005, pp. 886–893.

[27] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan, "Fast human detection using a cascade of histograms of oriented gradients," in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, vol. 2, 2006, pp. 1491–1498.

[28] I. Haritaoglu, D. Harwood, and L. Davis, "Hydra: multiple people detection and tracking using silhouettes," in *Image Analysis and Processing, International Conference on*, 1999, pp. 280–285.

[29] F. Porikli, "Integral histogram: a fast way to extract histograms in cartesian spaces," in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, vol. 1, Jun. 2005, pp. 829–836.

[30] O. Tuzel, F. Porikli, and P. Meer, "Region covariance: A fast descriptor for detection and classification," in *Computer Vision (ECCV), European Conference on*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006, vol. 3952, pp. 589–600.

[31] ——, "Human detection via classification on riemannian manifolds," in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, Jun. 2007, pp. 1–8.

[32] D.-Y. Chen, K. Cannons, H.-R. Tyan, S.-W. Shih, and H.-Y. Liao, "Spatiotemporal motion analysis for the detection and classification of moving targets," *Multimedia, IEEE Transactions on*, vol. 10, no. 8, pp. 1578–1591, Dec. 2008.

[33] R. Mosberger and H. Andreasson, "An inexpensive monocular vision system for tracking humans in industrial environments," in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2013, pp. 5850–5857.

[34] B. Choi, C. Mericli, J. Biswas, and M. Veloso, “Fast human detection for indoor mobile robots using depth images,” in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2013, pp. 1108–1113.

[35] Y. Freund and R. Schapire, “A short introduction to boosting,” *Japanese Society for Artificial Intelligence*, vol. 14, no. 5, pp. 771–780, 1999.

[36] C. Zhang and Z. Zhang, “A survey of recent advances in face detection,” Microsoft Research, Tech. Rep. MSR-TR-2010-66, Jun. 2010.

[37] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, “Real-time plane segmentation using rgb-d cameras,” in *RoboCup Symposium*, 2011.

[38] D. Comaniciu, V. Ramesh, and P. Meer, “Real-time tracking of non-rigid objects using mean shift,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, vol. 2, 2000, pp. 142–149.

[39] D. Lowe, “Distinctive image features from scale-invariant keypoints,” *Computer Vision (IJCV), International Journal of*, vol. 60, no. 2, pp. 91–110, 2004.

[40] L. Spinello and K. Arras, “People detection in rgb-d data,” in *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, Sep. 2011, pp. 3838–3843.

[41] H. Hashimoto, A. Sasaki, S. Yokota, Y. Ohyama, and C. Ishii, “A study on degree of freedom in hand modeling,” in *Proceedings of the SICE Annual Conference*, Sep. 2011, pp. 2492–2493.

[42] S. Koo, D. Lee, and D.-S. Kwon, “Gmm-based 3d object representation and robust tracking in unconstructed dynamic environments,” in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2013, pp. 1114–1121.

[43] T. Oggier, M. Lehmann, R. Kaufmann, M. Schweizer, M. Richter, P. Metzler, G. Lang, F. Lustenberger, and N. Blanc, “An all-solid-state optical range camera for 3d real-time imaging with sub-centimeter depth resolution (swissranger),” *Proceedings of the International Society for Optics and Photonics (SPIE)*, vol. 5249, pp. 534–545, 2004.

[44] A. Kolb, E. Barth, R. Koch, and R. Larsen, "Time-of-Flight cameras in computer graphics," *Computer Graphics Forum*, vol. 29, no. 1, pp. 141–159, 2010.

[45] M. Ruhnke, B. Steder, G. Grisetti, and W. Burgard, "Unsupervised learning of 3d object models from partial views," in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2009, pp. 801–806.

[46] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the Alvey Vision Conference.* Alvety Vision Club, 1988, pp. 23.1–23.6.

[47] M. N. Dailey and N. Bo Bo, "Towards real-time hand tracking in crowded scenes," in *Industrial Automation and Robotics (ACIAR), Asian Conference on*, 2005, pp. F–70.

[48] S. Ikemura and H. Fujiyoshi, "Real-time human detection using relational depth similarity features," in *Computer Vision (ACCV), Asian Conference on*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, vol. 6495, pp. 25–38.

[49] Z. Li and D. Kulic, "Local shape context based real-time endpoint body part detection and identification from depth images," in *Computer and Robot Vision (CRV), Canadian Conference on*, May 2011, pp. 219–226.

[50] M.-H. Yang, N. Ahuja, and M. Tabb, "Extraction of 2d motion trajectories and its application to hand gesture recognition," *Pattern Analysis and Machine Intelligence (PAMI), IEEE Transactions on*, vol. 24, no. 8, pp. 1061–1074, Aug. 2002.

[51] L. Bretzner, I. Laptev, and T. Lindeberg, "Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering," in *Automatic Face and Gesture Recognition, IEEE International Conference on*, May 2002, pp. 423–428.

[52] C. Shan, Y. Wei, T. Tan, and F. Ojardias, "Real time hand tracking by combining particle filtering and mean shift," in *Automatic Face and Gesture Recognition, IEEE International Conference on*, May 2004, pp. 669–674.

[53] S. Carbini, J.-E. Viallet, O. Bernier, and B. Bascle, "Tracking body parts of multiple people for multi-person multimodal interface," in *Computer Vision in Human-Computer*

Interaction, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, vol. 3766, pp. 16–25.

[54] S. E. Ghobadi, O. E. Loepprich, K. Hartmann, and L. O., "Hand segmentation using 2d/3d images," in *Image and Vision Computing New Zealand (IVCNZ), International Conference on*, Dec. 2007, pp. 64–69.

[55] K. Nickel and R. Stiefelhagen, "Visual recognition of pointing gestures for humanrobot interaction," *Image and Vision Computing*, vol. 25, no. 12, pp. 1875–1884, 2007.

[56] S. Valibeik and G.-Z. Yang, "Segmentation and tracking for vision based human robot interaction," in *Web Intelligence and Intelligent Agent Technology, IEEE/WIC/ACM International Conference on*, vol. 3, Dec. 2008, pp. 471–476.

[57] B.-J. Chen, C.-M. Huang, T.-E. Tseng, and L.-C. Fu, "Robust head and hands tracking with occlusion handling for human machine interaction," in *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, Oct. 2012, pp. 2141–2146.

[58] D. Comaniciu and V. Ramesh, "Mean shift and optimal prediction for efficient object tracking," in *Image Processing, International Conference on*, vol. 3, 2000, pp. 70–73.

[59] L. Latecki, R. Lakamper, and T. Eckhardt, "Shape descriptors for non-rigid shapes with a single closed contour," in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, vol. 1, 2000, pp. 424–429.

[60] D. Zhang and G. Lu, "A comparative study on shape retrieval using fourier descriptors with different shape signatures," in *Intelligent Multimedia and Distance Education, International Conference on*, 2001.

[61] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *Pattern Analysis and Machine Intelligence (PAMI), IEEE Transactions on*, vol. 24, no. 4, pp. 509–522, Apr. 2002.

[62] R. Jain, R. Kasturi, and B. G. Schunck, *Machine vision.* McGraw-Hill, Inc., 1995.

[63] E. Kalogerakis, A. Hertzmann, and K. Singh, "Learning 3d mesh segmentation and labeling," in *Special Interest Group on Graphics (SIGGRAPH), ACM Conference on.* ACM, 2010, pp. 102:1–102:12.

[64] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Machine Learning, International Conference on.* Morgan Kaufmann Publishers Inc., 2001, pp. 282–289.

[65] D. Anguelov, B. Taskarf, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng, "Discriminative learning of markov random fields for segmentation of 3d scan data," in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, vol. 2, Jun. 2005, pp. 169–176.

[66] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun, "Real time motion capture using a single time-of-flight camera," in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, Jun. 2010, pp. 755–762.

[67] V. Lepetit, P. Lagger, and P. Fua, "Randomized trees for real-time keypoint recognition," in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, vol. 2, Jun. 2005, pp. 775–781.

[68] D. Parks and S. Fels, "Evaluation of background subtraction algorithms with post-processing," in *Advanced Video and Signal Based Surveillance (AVSS), IEEE International Conference on*, Sep. 2008, pp. 192–199.

[69] C. Stauffer and W. Grimson, "Adaptive background mixture models for real-time tracking," in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on.*, vol. 2, 1999, pp. 252–258.

[70] P. W. Power and J. A. Schoonees, "Understanding background mixture models for foreground segmentation," in *Image and Vision Computing New Zealand (IVCNZ), International Conference on*, 2002, p. 267.

[71] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *Pattern Recognition (ICPR), International Conference on*, vol. 2, Aug. 2004, pp. 28–31.

[72] A. Störmer, M. Hofmann, and G. Rigoll, "Depth gradient based segmentation of overlapping foreground objects in range images," in *Information Fusion (FUSION), International Conference on*, Jul. 2010, pp. 1–4.

[73] B. Langmann, S. E. Ghobadi, K. Hartmann, and O. Loffeld, "Multi-modal background subtraction using gaussian mixture models," in *Photogrammetric Computer Vision and Image Analysis, ISPRS Technical Commission Symposium on*, vol. XXXVIII, 2010, pp. 61–66.

[74] G. Mori and J. Malik, "Estimating human body configurations using shape context matching," in *Computer Vision (ECCV), European Conference on*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2002, vol. 2352, pp. 666–680.

[75] S. Knoop, S. Vacek, and R. Dillmann, "Sensor fusion for 3d human body tracking with an articulated 3d body model," in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2006, pp. 1686–1691.

[76] J. Darby, B. Li, N. Costen, D. Fleet, and N. Lawrence, "Backing off: Hierarchical decomposition of activity for 3d novel pose recovery," in *Proceedings of the British Machine Vision Conference (BMVC).* BMVA Press, 2009, pp. 11.1–11.11.

[77] L. Raskin, M. Rudzsky, and E. Rivlin, "3d human body-part tracking and action classification using a hierarchical body model," in *Proceedings of the British Machine Vision Conference (BMVC).* BMVA Press, 2009, pp. 12.1–12.11.

[78] Y. Zhu and K. Fujimura, "A bayesian framework for human body pose tracking from depth image sequences," *Sensors*, vol. 10, no. 5, pp. 5280–5293, 2010.

[79] M. Siddiqui and G. Medioni, "Human pose estimation from a single view point, real-time range sensor," in *Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE Conference on*, Jun. 2010, pp. 1–8.

[80] A. Moutzouris, J. Martinez-del Rincon, M. Lewandowski, J. Nebel, and D. Makris, "Human pose tracking in low dimensional space enhanced by limb correction," in *Image Processing (ICIP), IEEE International Conference on*, Sep. 2011, pp. 2301–2304.

[81] A. Moutzouris, J. del Rincon, J.-C. Nebel, and D. Makris, “Human pose tracking by hierarchical manifold searching,” in *Pattern Recognition (ICPR), International Conference on*, Nov. 2012, pp. 866–869.

[82] M. Sigalas, H. Baltzakis, and P. Trahanias, “Gesture recognition based on arm tracking for human-robot interaction,” in *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, Oct. 2010, pp. 5424–5429.

[83] P. Buehler, M. Everingham, D. P. Huttenlocher, and A. Zisserman, “Upper body detection and tracking in extended signing sequences,” *Computer Vision (IJCV), International Journal of*, vol. 95, no. 2, pp. 180–197, 2011.

[84] N. K. Iason Oikonomidis and A. Argyros, “Efficient model-based 3d tracking of hand articulations using kinect,” in *Proceedings of the British Machine Vision Conference (BMVC).* BMVA Press, 2011, pp. 101.1–101.11.

[85] Y. Wu and T. S. Huang, “Vision-based gesture recognition: A review,” in *Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction.* Springer-Verlag, 1999, pp. 103–115.

[86] L. Rabiner, “A tutorial on hidden markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[87] H.-I. Suk, B.-K. Sin, and S.-W. Lee, “Robust modeling and recognition of hand gestures with dynamic bayesian network,” in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, Dec. 2008, pp. 1–4.

[88] A. Wilson and A. Bobick, “Parametric hidden markov models for gesture recognition,” *Pattern Analysis and Machine Intelligence (PAMI), IEEE Transactions on*, vol. 21, no. 9, pp. 884–900, Sep. 1999.

[89] Z. Ghahramani and M. Jordan, “Factorial hidden markov models,” *Machine Learning*, vol. 29, no. 2-3, pp. 245–273, 1997.

[90] M. Brand, N. Oliver, and A. Pentland, “Coupled hidden markov models for complex action recognition,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, Jun. 1997, pp. 994–999.

[91] A. Just, O. Bernier, and S. Marcel, “Hmm and iohmm for the recognition of mono- and bi-manual 3d hand gestures,” in *Proceedings of the British Machine Vision Conference (BMVC).* BMVA Press, 2004, pp. 28.1–28.10.

[92] B.-W. Min, H.-S. Yoon, J. Soh, Y.-M. Yang, and T. Ejima, “Hand gesture recognition using hidden markov models,” in *Systems, Man, and Cybernetics, Computational Cybernetics and Simulation, IEEE International Conference on*, vol. 5, Oct. 1997, pp. 4232–4235.

[93] S. Eickeler, A. Kosmala, and G. Rigoll, “Hidden markov model based continuous online gesture recognition,” in *Pattern Recognition (ICPR), International Conference on*, vol. 2, Aug. 1998, pp. 1206–1208.

[94] Y. Iwai, H. Shimizu, and M. Yachida, “Real-time context-based gesture recognition using hmm and automaton,” in *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, International Workshop on*, 1999, pp. 127–134.

[95] F.-S. Chen, C.-M. Fu, and C.-L. Huang, “Hand gesture recognition using a real-time tracking method and hidden markov models,” *Image and Vision Computing*, vol. 21, no. 8, pp. 745–758, 2003.

[96] T. Mori, Y. Segawa, M. Shimosaka, and T. Sato, “Hierarchical recognition of daily human actions based on continuous hidden markov models,” in *Automatic Face and Gesture Recognition, IEEE International Conference on*, May 2004, pp. 779–784.

[97] K. S. Huang and M. M. Trivedi, “3d shape context based gesture analysis integrated with tracking using omni video array,” in *Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE Conference on*, Jun. 2005, pp. 80–88.

[98] C. Joslin, A. El-Sawah, Q. Chen, and N. Georganas, “Dynamic gesture recognition,” in *Instrumentation and Measurement Technology (IMTC), IEEE Conference on*, vol. 3, May 2005, pp. 1706–1711.

[99] M. B. Holte, T. B. Moeslund, and P. Fihl, “View invariant gesture recognition using the csem swissranger sr-2 camera,” in *Intelligent Systems Technologies and Applications, International Journal of*, vol. 5, 2005, pp. 295–303.

[100] Z. Li and R. Jarvis, “Real time hand gesture recognition using a range camera,” in *Robotics and Automation, Australasian Conference on*, 2009, pp. 529–534.

[101] Y. Gu, H. Do, Y. Ou, and W. Sheng, “Human gesture recognition through a kinect sensor,” in *Robotics and Biomimetics (ROBIO), IEEE International Conference on*, Dec. 2012, pp. 1379–1384.

[102] Y. Gu, W. Sheng, Y. Ou, M. Liu, and S. Zhang, “Human action recognition with contextual constraints using a rgb-d sensor,” in *Robotics and Biomimetics (ROBIO), IEEE International Conference on*, Dec. 2013, pp. 674–679.

[103] S. P. Priyal and P. K. Bora, “A robust static hand gesture recognition system using geometry based normalizations and krawtchouk moments,” *Pattern Recognition*, vol. 46, no. 8, pp. 2202–2219, 2013.

[104] P. Neto, D. Pereira, J. Norberto Pires, and A. Moreira, “Real-time and continuous hand gesture spotting: An approach based on artificial neural networks,” in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2013, pp. 178–183.

[105] G. Chen, F. Zhang, M. Giuliani, C. Buckl, and A. Knoll, “Unsupervised learning spatio-temporal features for human activity recognition from rgb-d video data,” in *Social Robotics*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2013, vol. 8239, pp. 341–350.

[106] Y. Rui and P. Anandan, “Segmenting visual actions based on spatio-temporal motion patterns,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, vol. 1, 2000, pp. 111–118.

[107] G. S. Schmidt, D. H. House, and M. Miller, “Model-based gesture recognition,” Texas A & M University, Tech. Rep., 1999.

[108] M. Bdiwi, A. Kolker, J. Such, and A. Winkler, “Automated assistance robot system for transferring model-free objects from/to human hand using vision/force control,” in *Social Robotics*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2013, vol. 8239, pp. 40–53.

[109] A. Haro, M. Flickner, and I. Essa, “Detecting and tracking eyes by using their physiological properties, dynamics, and appearance,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, vol. 1, 2000, pp. 163–168.

[110] C. Morimoto, A. Amir, and M. Flickner, “Detecting eye position and gaze from a single camera and 2 light sources,” in *Pattern Recognition (ICPR), International Conference on*, vol. 4, 2002, pp. 314–317.

[111] Z. Zhu and Q. Ji, “Eye and gaze tracking for interactive graphic display,” *Machine Vision and Applications*, vol. 15, no. 3, pp. 139–148, 2004.

[112] D. H. Yoo and M. J. Chung, “A novel non-intrusive eye gaze estimation using cross-ratio under large head motion,” *Computer Vision and Image Understanding*, vol. 98, no. 1, pp. 25–51, 2005.

[113] B. Benfold and I. Reid, “Colour invariant head pose classification in low resolution video,” in *Proceedings of the British Machine Vision Conference (BMVC).* BMVA Press, 2008, pp. 49.1–49.10.

[114] A. Czyzewski, B. Kunka, M. Kurkowski, and R. Branchat, “Comparison of developed gaze point estimation methods,” in *Signal Processing Algorithms, Architectures, Arrangements, and Applications (SPA)*, Sep. 2008, pp. 133–136.

[115] C. W. Cho, J. W. Lee, E. C. Lee, and K. R. Park, “Robust gaze-tracking method by using frontal-viewing and eye-tracking cameras,” *Optical Engineering*, vol. 48, no. 12, pp. 127 202:1–127 202:15, 2009.

[116] J. W. Lee, C. W. Cho, K. Y. Shin, E. C. Lee, and K. R. Park, “3d gaze tracking method using purkinje images on eye optical model and pupil,” *Optics and Lasers in Engineering*, vol. 50, no. 5, pp. 736–751, 2012.

[117] S. Sheikhi and J.-M. Odobez, “Recognizing the visual focus of attention for human robot interaction,” in *Human Behavior Understanding*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, vol. 7559, pp. 99–112.

[118] D.-C. Cho and W.-Y. Kim, “Long-range gaze tracking system for large movements,” *Biomedical Engineering, IEEE Transactions on*, vol. 60, no. 12, pp. 3432–3440, Dec. 2013.

[119] T. Axenbeck, M. Bennewitz, S. Behnke, and W. Burgard, “Recognizing complex, parameterized gestures from monocular image sequences,” in *Humanoid Robots, IEEE/RAS International Conference on*, Dec. 2008, pp. 687–692.

[120] K. Funes Mora and J. Odobez, “Gaze estimation from multimodal kinect data,” in *Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE Conference on*, Jun. 2012, pp. 25–30.

[121] J. Choi, B. Ahn, J. Parl, and I. S. Kweon, “Appearance-based gaze estimation using kinect,” in *Ubiquitous Robots and Ambient Intelligence (URAI), International Conference on*, Oct. 2013, pp. 260–261.

[122] D. Kim, J. Park, and A. Kak, “Estimating head pose with an rgbd sensor: A comparison of appearance-based and pose-based local subspace methods,” in *Image Processing (ICIP), IEEE International Conference on*, Sep. 2013, pp. 3637–3641.

[123] G. Fanelli, J. Gall, and L. Van Gool, “Real time head pose estimation with random regression forests,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, Jun. 2011, pp. 617–624.

[124] P. Padeleris, X. Zabulis, and A. Argyros, “Head pose estimation on depth data based on particle swarm optimization,” in *Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE Conference on*, Jun. 2012, pp. 42–49.

[125] R. Stiefelhagen, “Tracking focus of attention in meetings,” in *Multimodal Interfaces, IEEE International Conference on*, 2002, pp. 273–280.

[126] I. J. Hodgkinson, I. J. Hodgkinson, and A. C. B. Molteno, “Point-spread function for light scattered in the human ocular fundus,” *Journal of the Optical Society of America*, vol. 11, no. 2, pp. 479–486, Feb. 1994.

[127] A. Gullstrand, *Helmholtz's Treatise on Physiological Optics.* The Optical Society of America, 1924.

[128] I. Guyon, V. Athitsos, P. Jangyodsuk, B. Hamner, and H. Escalante, “Chalearn gesture challenge: Design and first results,” in *Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE Conference on*, Jun. 2012, pp. 1–6.

[129] Y. Zhu and K. Fujimura, “Constrained optimization for human pose estimation from depth sequences,” in *Computer Vision (ACCV), Asian Conference on*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, vol. 4843, pp. 408–418.

[130] D. Grest, J. Woetzel, and R. Koch, “Nonlinear body pose estimation from depth images,” in *Pattern Recognition, DAGM Conference on*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, vol. 3663, pp. 285–292.

[131] D. Young, U. of Sussex. School of Cognitive, and C. Sciences, *Straight lines and circles in the log-polar image*, ser. Cognitive science research papers. School of Cognitive and Computing Sciences, University of Sussex, 2000.

[132] M. Chen *et al.*, “Priority queues and dijkstras algorithm,” The University of Texas at Austin, Department of Computer Sciences, Tech. Rep., 2007.

[133] G. Welch and G. Bishop, “An introduction to the kalman filter,” University of North Carolina at Chapel Hill, Tech. Rep., 1995.

[134] S. Mitra and T. Acharya, “Gesture recognition: A survey,” *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 37, no. 3, pp. 311–324, May 2007.

[135] M. Donoser and H. Bischof, "Real time appearance based hand tracking," in *Pattern Recognition, International Conference on*, Dec. 2008, pp. 1–4.

[136] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[137] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, pp. 83–97, 1955.

[138] H. Stern and B. Efros, "Adaptive color space switching for face tracking in multi-colored lighting environments," in *Automatic Face and Gesture Recognition, IEEE International Conference on*, May 2002, pp. 249–254.

[139] M. Van den Bergh, D. Carton, R. de Nijs, N. Mitsou, C. Landsiedel, K. Kuehnlenz, D. Wollherr, L. Van Gool, and M. Buss, "Real-time 3d hand gesture interaction with a robot for understanding directions from humans," in *Robot and Human Interactive Communication (RO-MAN), IEEE International Symposium on*, Jul. 2011, pp. 357–362.

[140] M. Van den Bergh and L. Van Gool, "Combining rgb and tof cameras for real-time 3d hand gesture interaction," in *Applications of Computer Vision (WACV), IEEE Workshop on*, Jan. 2011, pp. 66–72.

[141] E. Hall, *The hidden dimension*, ser. Doubleday Anchor Books. Doubleday, 1966.

[142] M. Walters, K. Dautenhahn, K. Koay, C. Kaouri, R. Boekhorst, C. Nehaniv, I. Werry, and D. Lee, "Close encounters: spatial distances between people and a robot of mechanistic appearance," in *Humanoid Robots, IEEE/RAS International Conference on*, Dec. 2005, pp. 450–455.

[143] S. Caraian and N. Kirchner, "Head pose behavior in the human-robot interaction space," in *Human-Robot Interaction (HRI), ACM/IEEE International Conference on.* ACM, 2014, pp. 132–133.

[144] D. Droeschel, J. Stuckler, D. Holz, and S. Behnke, "Towards joint attention for a domestic service robot - person awareness and gesture recognition using time-of-flight cameras,"

in *Robotics and Automation (ICRA), IEEE International Conference on*, May 2011, pp. 1205–1210.

[145] S. Satake, T. Kanda, D. Glas, M. Imai, H. Ishiguro, and N. Hagita, "How to approach humans? -strategies for social robots to initiate interaction," in *Human-Robot Interaction (HRI), ACM/IEEE International Conference on*, Mar. 2009, pp. 109–116.

[146] N. A. Dodgson, "Variation and extrema of human interpupillary distance," in *Storage and Retrieval for Image and Video Databases*, 2004.

[147] M. Mucientes and W. Burgard, "Multiple hypothesis tracking of clusters of people," in *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, Oct. 2006, pp. 692–697.

[148] B. Lau, K. Arras, and W. Burgard, "Multi-model hypothesis group tracking and group size estimation," *International Journal of Social Robotics*, vol. 2, no. 1, pp. 19–30, 2010.

[149] H. Hung and B. Kröse, "Detecting f-formations as dominant sets," in *Multimodal Interfaces, International Conference on*. ACM, 2011, pp. 231–238.

[150] L. Bazzani, M. Cristani, D. Tosato, M. Farenzena, G. Paggetti, G. Menegaz, and V. Murino, "Social interactions by visual focus of attention in a three-dimensional environment," *Expert Systems*, vol. 30, no. 2, pp. 115–127, 2013.

[151] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

[152] A. Kendon, "Spacing and orientation in co-present interaction," in *Development of Multimodal Interfaces: Active Listening and Synchrony*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2010, vol. 5967, pp. 1–15.

[153] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.

[154] J. Correa, S. McKeague, J. Liu, and G.-Z. Yang, "Gaze contingent group detection for socially aware mobile robot navigation," in *Submission*, Feb. 2015.

Appendix A

Support Vector Machines

Since its introduction in 1995, the SVM gained popularity for its performance as a non-probabilistic binary classifier. For linearly separable, two-class classification problems, many previous approaches were either dependent on an arbitrarily chosen parameter values or the order in which data is presented [151]. The SVM attempts to find the solution with the smallest classification error, by analysing the minimum margin of the data points. This is defined to be the smallest distance between the decision boundary and the closest data samples from both classes. The decision boundary is chosen so that this minimum margin is maximised. This section will present an overview of the standard theory behind the SVM, as it is used for hand classification in Chapter 3.

Input training data comprises N input vectors, $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$, with corresponding target values $\boldsymbol{t} = (t_1, \ldots, t_N)$, where $t_n \in (1, -1)$. Each $\mathbf{x}_n$ is a $D \times 1$ hand descriptor vector from Section 3.2.1, where D is the dimensionality of the descriptor. If $\mathbf{x}_n$ is a descriptor of a hand, then its corresponding $t_n = 1$; otherwise, for example, if the descriptor is from a background keypoint, it's corresponding $t_n = -1$.

New data points, $\mathbf{x}$, are classified as hands or background according to the sign of a function $y(\mathbf{x})$. This function will satisfy $y(\mathbf{x}_n) > 0$ for hands where $t_n = 1$, and $y(\mathbf{x}_n) < 0$ for non-hands where $t_n = -1$. The decision boundary will thus be a hyperplane defined by $y(\mathbf{x}) = 0$, shown

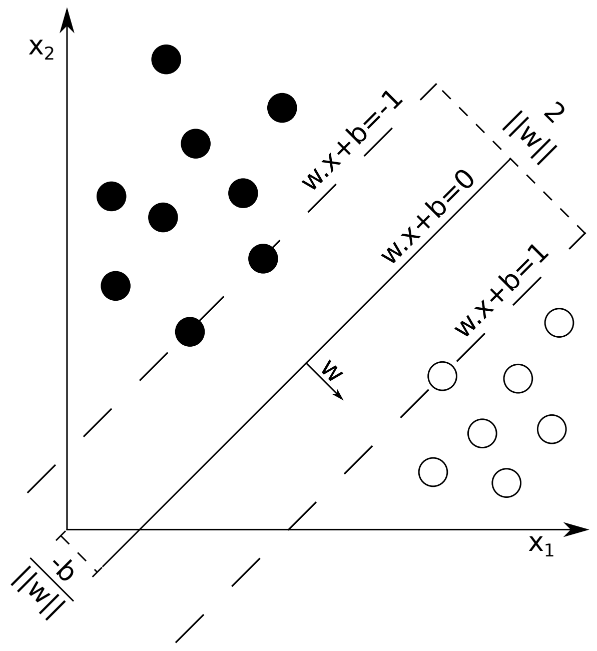


Figure A.1: Illustration of a SVM classifier, in 2D space, trained on two classes, shown by black and white data points. In the presented hand detection application, hands are represented by white circles, and non-hands are represented by black circles. The decision boundary is a solid line, mid-way between two margin hyperplanes, shown by dashed lines. The dashed margin hyperplanes intersect the nearest data points from each class, termed "support vectors", such that the distance between them is maximised.

in Figure A.1.

$$y\left(\mathbf{x}\right) = \mathbf{w}^T\mathbf{x} + b. \tag{A.1}$$

$\mathbf{w}$ is a $D \times 1$ vector that is normal to the decision boundary, and b is the offset of the decision boundary from the origin. To prove this, picking two points, $\mathbf{x}_A$ and $\mathbf{x}_B$, on the decision boundary gives: $y\left(\mathbf{x}_A\right) = y\left(\mathbf{x}_B\right) = 0$. Then, from the equation: $y\left(\mathbf{x}_A\right) - y\left(\mathbf{x}_B\right) = \mathbf{w}^T\left(\mathbf{x}_A - \mathbf{x}_B\right) = 0$, it can be said that vector $\mathbf{w}$ is orthogonal to every vector on the decision boundary. Additionally, the following can be formulated: $\mathbf{w}^T\mathbf{x}_A = -b$. Diving both sides by $||\mathbf{w}||$ shows that the distance from the origin, of a point on the decision surface, in the direction of its normal, is $\frac{-b}{||\mathbf{w}||}$.

Any point, $\mathbf{x}$, can be represented by its orthogonal projection onto the decision boundary, $\mathbf{x}_\perp$, and its orthogonal distance to the decision boundary, r, such that: $\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{||\mathbf{w}||}$. Multiplying both sides by $\mathbf{w}^T$ and adding b gives: $y\left(\mathbf{x}\right) = \mathbf{w}^T\mathbf{x}_\perp + \mathbf{w}^T r\frac{\mathbf{w}}{||\mathbf{w}||} + b$. Knowing that $y\left(\mathbf{x}_{\perp}\right) = \mathbf{w}^T\mathbf{x}_{\perp} + b = 0$ and $\mathbf{w}^T\mathbf{w} = \|\mathbf{w}\|^2$, gives the result: $r = \frac{y(\mathbf{x})}{\|\mathbf{w}\|}$. In other words, the perpendicular distance of a point $\mathbf{x}$ to the decision boundary is given by $\frac{y(\mathbf{x})}{\|\mathbf{w}\|}$. Again, these relationships are shown diagrammatically in Figure A.1.

The decision boundary parameters $\mathbf{w}$ and b will be chosen as those which maximise the distance from the decision boundary to the closest training set points in each class. These points are termed "support vectors", and will be denoted by the set $\mathcal{S}$. This optimisation problem can be formulated by firstly assuming that the support vectors will be at a perpendicular distance of $\frac{1}{\|\mathbf{w}\|}$ from the decision boundary. Thus:

$$\mathbf{w}^T\mathbf{x}_m + b = \begin{cases} 1 & \text{if } t_m = 1 \\ \\ -1 & \text{if } t_m = -1. \end{cases}, \quad \mathbf{x}_m \in \mathcal{S} \tag{A.2}$$

Subject to these constraints, we wish to maximise the minimum margin, by choosing values of $\mathbf{w}$ and b in order to maximise $\frac{1}{\|\mathbf{w}\|}$. This can be formulated by combining the cases in Equation A.2:

$$\underset{\mathbf{w},b}{\text{maximise}} \quad \frac{1}{\|\mathbf{w}\|} \tag{A.3}$$

$$\text{subject to} \quad g_n\left(\mathbf{w}, b\right) \geq 0\,, \quad 1 \leq n \leq N \tag{A.4}$$

$$\text{where} \quad g_n\left(\mathbf{w}, b\right) = t_n\left(\mathbf{w}^T\mathbf{x}_n + b\right) - 1 \tag{A.5}$$

Due to the square root involved in calculating Equation A.3, it is computationally easier to minimise $\frac{1}{2}\|\mathbf{w}\|^2$. Lagrange multipliers, $\boldsymbol{a} = (a_1 \ldots a_N)$, are used to solve this constrained optimisation problem, with one multiplier for each of the constraints in Equation A.4. The Lagrangian solution is then given by:

$$\underset{\mathbf{w},b}{\text{minimise}} \quad \underset{\boldsymbol{a}\geq 0}{\text{maximise}} \quad L(\mathbf{w}, b, \boldsymbol{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n\left(t_n\left(\mathbf{w}^T\mathbf{x}_n + b\right) - 1\right) \tag{A.6}$$

Because of the inequality in Equation A.4, Lagrange multipliers are subject to: $a_n \geq 0$. Many

a_n values will be zero, corresponding to $\mathbf{x}_n$ values for which $g_n(\mathbf{w}, b) > 0$ in Equation A.4. However the $\mathbf{x}_n$ values for which $g_n(\mathbf{w}, b) = 0$, will have a corresponding $a_n > 0$. These points are the "support vectors".

Setting to zero the derivative of the Lagrange function, with respect to $\mathbf{w}$ (Equation A.7) and b (Equation A.8), gives the desired decision boundary parameters; those which maximise the minimum margin. Using the fact that $\|\mathbf{w}\|^2 = \mathbf{w}^T\mathbf{w}$, these are:

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n \tag{A.7}$$

$$0 = \sum_{n=1}^{N} a_n t_n. \tag{A.8}$$

To calculate the optimal Lagrange multipliers, $\boldsymbol{a}$, Equations A.7 and A.8 can be substituted into Equation A.6. This eliminates $\mathbf{w}$ and b using their maximised conditions. Thus, under the constraints of Equation A.8 and $a_n \geq 0$, the optimal value of $\boldsymbol{a}$ is given by maximising the following equation with respect to $\boldsymbol{a}$:

$$\underset{\boldsymbol{a}\geq 0}{\text{maximise}} \quad \tilde{L}(\boldsymbol{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m \mathbf{x}_n^T \mathbf{x}_m \tag{A.9}$$

$$\text{subject to} \quad \sum_{n=1}^{N} a_n t_n = 0 \tag{A.10}$$

$$a_m \geq 0\,, \quad 1 \leq m \leq N \tag{A.11}$$

With optimal $\boldsymbol{a}$ values defined, by substituting Equation A.7 into Equation A.1, a new hand descriptor can be classified as hand or background according to the sign of:

$$y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n \mathbf{x}^T \mathbf{x}_n + b. \tag{A.12}$$

This equation reveals that only "support vectors", $\mathbf{x}_n$ with a corresponding $a_n > 0$, have any effect on the optimal decision boundary. However, an optimal value for b remains to be defined. Denoting the "support vector" indices in the training set as the set $\mathcal{S}$, any point $\mathbf{x}_n$ with $n \in \mathcal{S}$

satisfies $t_n y(\mathbf{x}_n) = 1$. Finally, with this observation, Equation A.12 can be used to give:

$$\begin{aligned} t_n \left(\sum_{m \in \mathcal{S}} a_m t_m \mathbf{x}_n^T \mathbf{x}_m + b \right) &= 1, \\ b &= \frac{1}{t_n} - \sum_{m \in \mathcal{S}} a_m t_m \mathbf{x}_n^T \mathbf{x}_m \end{aligned} \tag{A.13}$$

Appendix B

Mobile Robot Navigation Frameworks

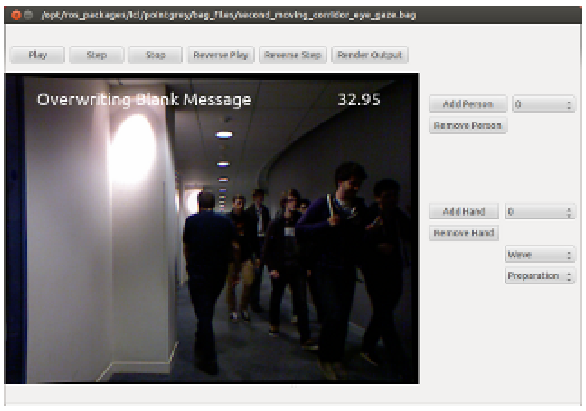
Two robot frameworks were investigated in the creation of this thesis: Player and ROS. However, due to its many advantages, all code was written in ROS. Real-time code was written in C++, although many tools, such as the two shown in Figure B.1, were written in Python.

B.1 Player

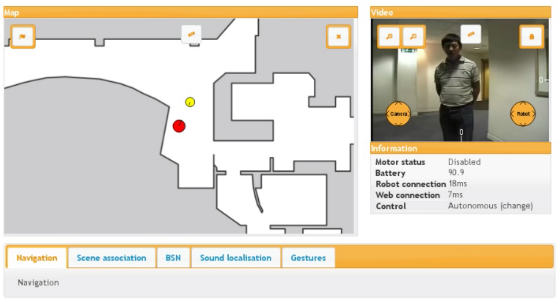
For many years Player was the industry standard robot framework. It provided a networked robot server along with an abstraction layer to handle data communication with the robots sensors and actuators.

The Player server is initialised with a configuration file detailing the sensors and actuators to be made available. For each device to be configured, the port to which the device is connected is specified, along with a driver by which is can be accessed. An device identifier is also specified, which is the unique identifier by which the device is accessed on the Player server. This is composed of an interface, which denotes the generic device type, and an index to facilitate unique addressing. A sample configuration file is shown in Figure B.2.

The Player server can also be loaded with drivers that refer to virtual devices. These virtual devices do not control any hardware themselves, but rather operate on sensor input from other



(a) Annotation tool



Motor status	Disabled
Battery	90.9
Robot connection	18ms
Web connection	7ms
Control	Autonomous (change)

(b) Web-based teleoperation programme

Figure B.1: Screenshot of a two tools used in this thesis, that were written in Python using the ROS libraries. (a) shows a ground truth annotation tool, used for manually tagging people, hands and gestures. (b) shows a teleoperation tool, that allows a user to remotely control the robot using a web interface. A live map, showing robot and detected people positions, along with a video feed from the robot are displayed in real-time on the client's screen. The robot's movement can be controlled with on-screen controls, keyboard commands or a joystick.

```
driver
(
name ''p2os_position''
provides [''position:0'']
)
```

```
driver
(
name ''sicklms200''
provides [''laser:0'']
port ''/dev/ttyS1''
)
```

Figure B.2: Implementation code of sample player driver.

devices. They are used to implement important predefined algorithms in real-time, making their output conveniently available through the Player server.

The AMCL virtual device driver, for example, performs the Adaptive Monte-Carlo Localization algorithm on sensor information from a robots odometry, a laser device and a predefined map, outputting an accurate estimate of the robots location within this map.

The devices made available through the Player server are accessed through the Player client libraries. The client libraries connect to a networked Player server and handle retrieval of device information and actuator control. Control programs can be written in a variety of languages to facilitate ease of robot interaction.

Multiple networked client programmes can simultaneously connect to a Player server. Communication between clients is facilitated by a thread-safe memory map, implemented as a device driver called "Blackboard". In this way a client program can be confined to operating on one aspect of the overall robot control, improving the modularity of the overall system.

B.2 Robot Operating System

Development on the Robot Operating System (ROS) was started by the Willow Garage company in 2007. It has quickly become the state of the art robot framework, finding use in many high profile projects such as those in the Surrey Space Centre. It provides a much richer development environment than Player, with a multitude of debugging and analysis tools. The framework is composed of a very modular system of message passing, facilitating far better transparency into the inner workings of the framework than provided by Player.

The ROS communication infrastructure is built on a graph architecture, where process are represented as nodes that can both publish messages and subscribe to messages. ROS ensures that all message passing dependencies are handled at run time, enabling the developer to individually design each component of the system with seamless modularity. For example, a people tracking system might be implemented using a foreground detection node that passes messages

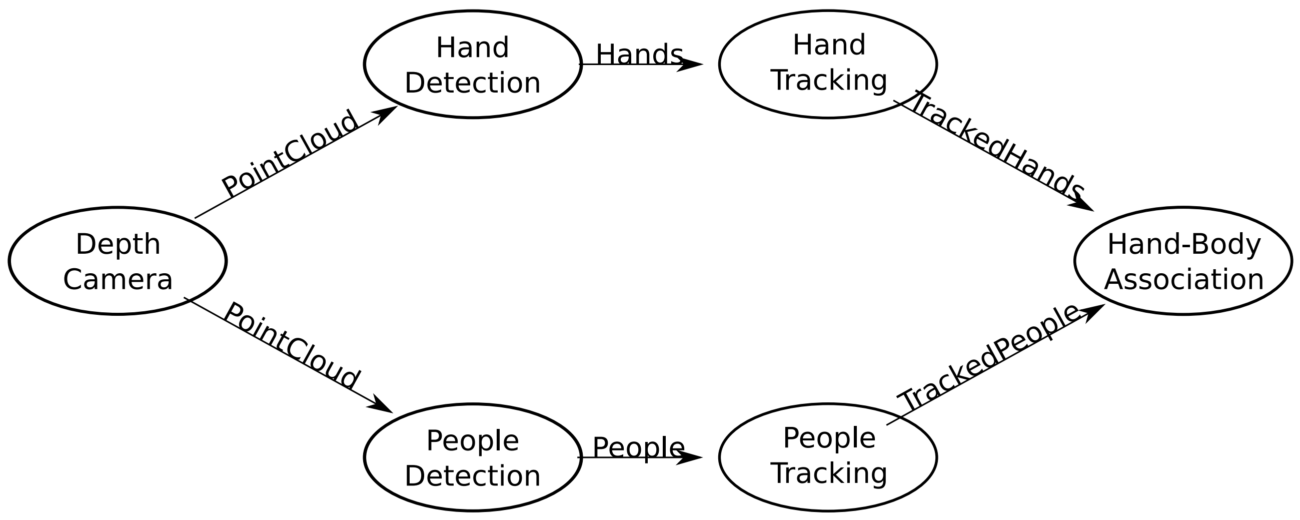


Figure B.3: Illustration graph of the ROS communication infrastructure for the hand and body association method described in Chapter 3.

to a point tracking node. At run-time ROS would enable the foreground detection node to be replaced with a feature detection node publishing the same type of messages, ensuring the function of the system remains the same. In this way ROS allows the developer to observe the output of swapping of algorithmic components, without any code changes necessary.

ROS contains invaluable tools for recording and playback of all sensor information, for visualising point cloud information published from 3D scanners and for plotting graphs of accelerometer data, to name but a few, that facilitate research in human-robot interaction.

Appendix C

Gaze Estimation Circuit Diagram

In Section 5.2.1, hardware was required that generated two pulse signals, so that in each camera frame only one set of LEDs is observable, with the opposite LED set being observable in the proceeding frame. The frequency of the pulse signals should thus be half the camera's frame rate, with a 50% duty cycle and a phase shift of 180°. Figure C.1 is a circuit diagram of the hardware used to alternate the power to the on and off-axis LEDs, using a square wave signal oscillator. This circuit diagram was designed in collaboration with the third author of the paper where the technique is published [12].

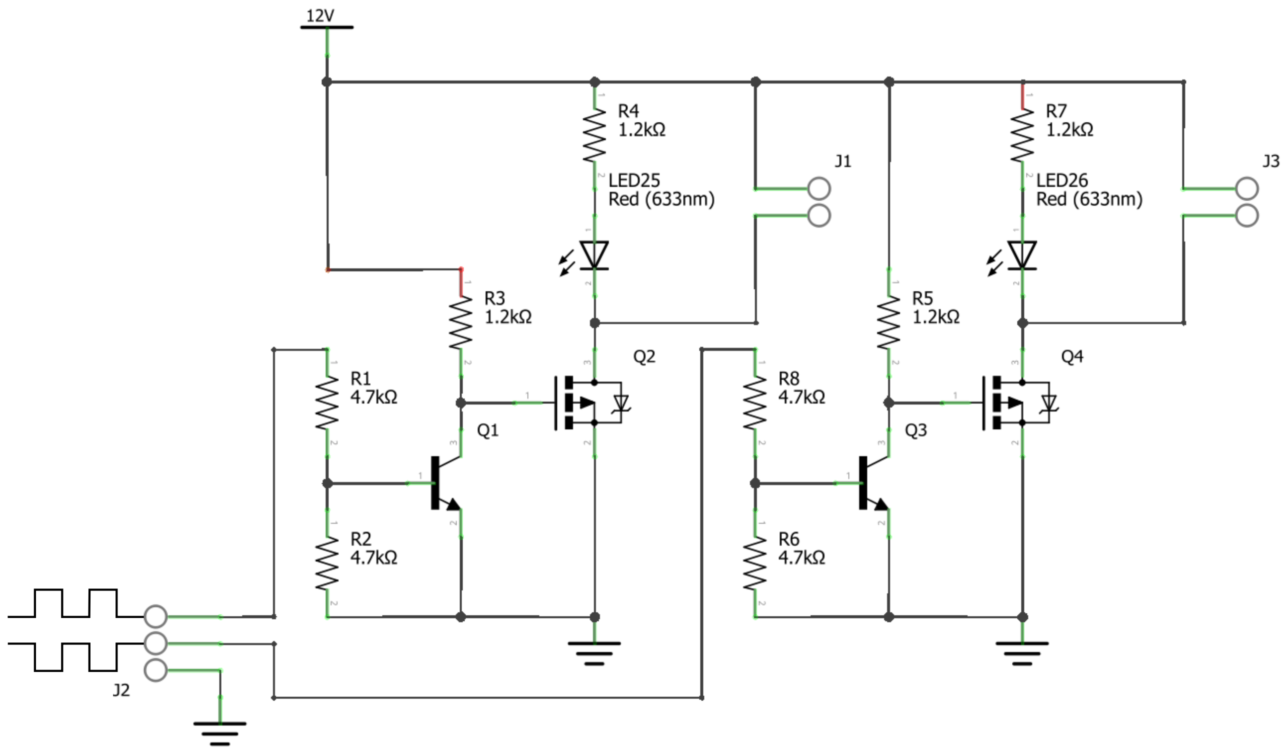


Figure C.1: Circuit diagram of the hardware used to alternate power to the on and off-axis LEDs. Screw terminal J1 is connected to the on-axis LEDs and screw terminal J3 is connected to the off-axis LEDs. Screw terminal J2 is connected to a square wave oscillator with frequency equal to half the camera's frame rate. As shown, the two pins propagate waves that have a 50% duty cycle and are separated by phase shifts of 180°. In this way only one set of LEDs is observable in each frame, and in the subsequent frame only the opposite set of LEDs is observable. The 633 nm wavelength LEDs provide a visual means of detecting if power is supplied to their connected set of IR LEDs.

Appendix D

Proof of Image Permissions

Figure 2.1d

From: Reid Simmons
Sent: 29 September 2014 15:12
To: McKeague, Stephen
Subject: Re: Image Permission

Yes, although obviously with appropriate attribution.

On 9/29/2014 9:41 AM, Stephen McKeague wrote:
>Hello Prof. Simmons,
>
>My name is Stephen McKeague, and I am a robotics Ph.D. student at Imperial College London. May I have permission to use an unaltered copy of Figure 1 from your paper "Xavier: An Autonomous Mobile Robot on the Web" in my thesis?
>
>Many thanks
>Stephen

Figure 2.1f

From: S Valibeik
Sent: 29 September 2014 14:46
To: McKeague, Stephen
Subject: Re: EO Robot

thanks , np.

On Mon, Sep 29, 2014 at 2:43 PM, Stephen McKeague wrote:
>Hi Salman, hope you're well.
>
>Could I have permission to use your picture of the RP-7 from Figure 2.4(a) in your Ph.D. thesis, in my own thesis
>
>Many thanks
>Stephen

Figure 2.1i

From: Christof Schrter
Sent: 29 September 2014 16:03
To: McKeague, Stephen
Subject: Re: Image Permission

Hi Stephen,

feel free to use that picture of the robot, I would only ask that you cite the mentioned paper for reference. [. . .]

On 29.09.2014 16:45, Stephen McKeague wrote:
>Hello
>
>My name is Stephen McKeague and I am a robotics Ph. D. student at Imperial College London. Can I have permission to use an unaltered copy of Figure 2 (left) from your paper "Realization and User Evaluation of a Companion Robot for People with Mild Cognitive Impairments" in my thesis?
>
>Many thanks,
>Stephen

Figure 2.3b

From: Stephen McKeague
Sent: 29 September 2014 15:00
To: thehavenresidentialhome
Subject: Image Permission

Hello,

My name is Stephen McKeague and I am a Ph. D. student at Imperial College London researching robot techniques for elderly care. Could I have permission to use an unaltered copy of the following image from your website in my Ph. D. thesis?
`http://www.thehavenresidentialhome.co.uk/images/caring-for-the-elderly.jpg`

Many thanks,
Stephen

Figure 2.5b

From: Stephen McKeague
Sent: 28 September 2014 19:32
To: Chris Coverley
Subject: Image Permission

Hello

My name is Stephen McKeague and I am a Ph. D. student at Imperial College London researching robot techniques for elderly care. Could I have permission to use an unaltered copy of the following image from your website in my Ph. D. thesis?
`http://www.clsgroup.org.uk/?q=sites/default/files/styles/home_slide/adaptive-image/public/homes/images/web1_10.jpg&itok=FnkIFzzo`

Many thanks
Stephen

Figure 2.7

From: knowlabs on behalf of Sebastian Thrun
Sent: 29 September 2014 01:54
To: McKeague, Stephen
Subject: Re: Image Permission

yes of course

On Sat, Sep 27, 2014 at 4:04 PM, McKeague, Stephen wrote:
>Dear Prof. Thrun,
>
>I am a robotics Ph.D. student at Imperial College London. May I have permission to include an unaltered copy of Figure 2 from your paper "Real-time Identification and Localization of Body Parts from Depth Images" in my Ph.D. thesis?
>
>Kind regards
>Stephen McKeague

Figure 2.9

From: Robert Wang
Sent: 28 September 2014 01:17
To: McKeague, Stephen
Subject: Re: Image Permission

Hi Stephen,
Yes. That's fine.

Best,
Rob

On 9/27/2014 4:08 PM, McKeague, Stephen wrote:
>Dear Dr. Wang
>
>I am a robotics Ph. D. student at Imperial College London. May I have permission to use an unaltered copy of Figure 1 (left) from your paper "Real-Time Hand-Tracking with a Color Glove" in my Ph.D. thesis?
>
>Kind regards
>Stephen McKeague

Figure 6.19

From: Marco Cristani
Sent: 16 September 2014 09:01
To: McKeague, Stephen
Subject: Re: Group Detection Dataset Pictures

Dear Stephen,

thanks for the interest for the group stuff! Francesco (in cc) will send you some images soon.
Best,
Marco

On 15/09/2014 17:57, McKeague, Stephen wrote:
>Hello Prof. Cristani

>My name is Stephen, and I am a Ph.D. student at Imperial College London. Regarding your group datasets at the URL below, could you please provide me with screenshots of the real datasets? I would like to include some results in my thesis, but need a comprehensive page of screenshots to be able to do so.
>http://profs.sci.univr.it/~cristanm/ssp/

>Many thanks in advance for your time
>Stephen