NAVIGATION BEHAVIOR DESIGN AND REPRESENTATIONS FOR A PEOPLE AWARE MOBILE ROBOT SYSTEM

A Thesis
Presented to
The Academic Faculty

by

Akansel Cosgun

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Computing

Georgia Institute of Technology
May 2016

NAVIGATION BEHAVIOR DESIGN AND REPRESENTATIONS FOR A PEOPLE AWARE MOBILE ROBOT SYSTEM

Approved by:

Professor Henrik Iskov Christensen,
Advisor
College of Computing
Georgia Institute of Technology

Professor Ayanna Howard
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Andrea Thomaz
College of Computing
Georgia Institute of Technology

Dr Emrah Akin Sisbot
Toyota InfoTechnology Center

Professor Irfan Essa
College of Computing
Georgia Institute of Technology

Date Approved: 1/11/2016

To my parents,

Zeynep and Huseyin Coşgun,

who are my pillars of strength;

and to my wife Gamze,

who always supports me, rain or shine.

ACKNOWLEDGEMENTS

First, I would like to thank my colleagues at IRIM, especially to Alex Trevor, Victor Emeli, Baris Akgun and Can Erdogan, for their friendship and useful research discussions.

Thanks to my committee for their helpful comments and feedback: Andrea Thomaz, Ayanna Howard and Irfan Essa. Thanks to Akin Sisbot, who also is on my committee, for his close collaboration at Toyota and setting the standard in this line of research. Thanks to Dinei Floriencio his mentorship during my time at Microsoft. Moreover, thanks to the Korea Industrial Technology Foundation, The Boeing Company, BMW, Peugeot S.A. and ThyssenKrupp for financially supporting the body of work in this thesis.

I would like to remember Mike Stilman, whose passing was a shock to all of us. I wrote my first research paper under his helpful guidance and his enthusiasm was contagious.

Finally, special thanks to my thesis advisor Henrik Christensen, for his to-the-point comments and foresighted advice, and being a role model to me with his management style and charisma.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

There are millions of robots in operation around the world today, and almost all of them operate on factory floors in isolation from people. However, it is now becoming clear that robots can provide much more value assisting people in daily tasks in human environments. Perhaps the most fundamental capability for a mobile robot is navigating from one location to another. Advances in mapping and motion planning research in the past decades made indoor navigation a commodity for mobile robots. Yet, questions remain on how the robots should move around humans. This thesis advocates the use of semantic maps and spatial rules of engagement to enable non-expert users to effortlessly interact with and control a mobile robot.

A core concept explored in this thesis is the Tour Scenario, where the task is to familiarize a mobile robot to a new environment after it is first shipped and unpacked in a home or office setting. During the tour, the robot follows the user and creates a semantic representation of the environment. The user labels objects, landmarks and locations by performing pointing gestures and using the robot's user interface. The spatial semantic information is meaningful to humans, as it allows providing commands to the robot such as "bring me a cup from the kitchen table". While the robot is navigating towards the goal, it should not treat nearby humans as obstacles and should move in a socially acceptable manner.

Three main navigation behaviors are studied in this work. The first behavior is the point-to-point navigation. The navigation planner presented in this thesis borrows ideas from human-human spatial interactions, and takes into account personal spaces as well as reactions of people who are in close proximity to the trajectory of the

robot. The second navigation behavior is person following. After the description of a basic following behavior, a user study on person following for telepresence robots is presented. Additionally, situation awareness for person following is demonstrated, where the robot facilitates tasks by predicting the intent of the user and utilizing the semantic map. The third behavior is person guidance. A tour-guide robot is presented with a particular application for visually impaired users.

CHAPTER I

INTRODUCTION

1.1 Robots in Human Environments

The vision promised by Karel Čapek in his 1921 science fiction play R.U.R, in which the word *"robot"* was used the first time, featured intelligent autonomous systems that co-existed and worked for humans. The reality almost a century later today is that robots do exist, however almost all of them operate in factories, physically separated from human workers. With this separation, the safety of humans is ensured. However, it is now becoming clear that robots can provide much more value if they operate in human environments, assisting people in daily tasks. With the recent improvements in hardware and software systems and in robotics research, development of such robots is not science fiction anymore. According to International Federation of Robotics (IFR), $2.2 billion worth of personal robots were sold in the year 2014, including robotics vacuum cleaners, lawn mowers, educational and entertainment robots. This disruptive technology will make fascinating new applications possible, in contexts such as homes, hospitals, offices and factories, including but not limited to: delivery, elderly care, collaborating on an assembly line and household tasks such as cleaning. The feasibility and reliability of such applications will determine their business value, while potentially generating a new industry. Therefore, any development in this field is a step toward realization of *intelligent* and *social* robots.

1.2 Robots that Navigate in Human Environments

The most important difference of a robot from a computer is actuation: the ability to move in the physical world. Perhaps the most fundamental capability for a mobile

robot is *navigation* from one location to another. Advances in mapping, localization and path planning research in the past decades made indoor navigation a common capability for mobile robots.

Robots occupy the same physical space with us, therefore should be aware of the fact that their movements would be observed and reacted by humans. As we start having more and more mobile robots in human environments, we will notice that robots can move in a way that is not socially acceptable by people. For example, robots can cut people's paths, navigate uncomfortably close to people and move in a way that doesn't signal where it is intended to go. Investigating the context of robots in human environments gave rise to the sub-field of robotics: Human-Robot Interaction (HRI). This relatively new research field extends in many directions with a common goal: robots that will perceive its environment, reason and act in a safe way to facilitate people's lives. Thus, a robot that will serve to people should not only be a machine, but also respect social rules and protocols.

1.3 The Science of Personal Spaces

How should mobile agents adjust their spatial relationships with respect to humans? In fact, humans already have a framework for this problem. It is called the *proxemics*, coined by Hall [40], which studies of our use of space and the spatial separation individuals naturally maintain. One of Hall's main findings is that humans adjust the amount of distance between them in four distance levels: intimate, personal, social and public distances, depending on the intimacy level between individuals. Another important research work in human-human is Kendon's F Formations [53], which studies the formations people tend to take in groups settings.

Personal spaces are influenced by other factors such as cultures, the range of their sensory systems, and postures of the two actors. Although it is not known whether it is best to model robot navigation behaviors after humans, applying the concepts from

human-human spatial studies is a good start. This is especially true given that non-expert users today never interacted with a mobile robot and thus are likely expect the robot to behave within human social norms.

1.4 Semantic Maps

Robots keep a representation of its environment to enable navigation behaviors, and this representation is usually in the form of a discrete metric *map*, where each grid cells represent whether that position is occupied with an obstacle or not. Mobile robots that use such a map can accept navigation goals in metric coordinates (i.e. go to coordinate (5.2, -1.3)), however exact coordinates may not be the most intuitive way to communicate goals for human users. Robots that navigate in human environments will need to accept human-friendly navigation goals. A map for such robots should facilitate establishment of a *common ground* between the robot and its user and enable referencing to the same spatial elements. For example, the robot should be able to understand commands such as "go to the kitchen table" or "wait outside Joe's room". This is only possible with a richer map representation that includes *semantic* information. In particular, if the humans and robots need to communicate over navigation goals that involve objects, landmarks or rooms, a map should include information about these.

1.5 Tour Scenario

One method to familiarize the robot to a new human environment is the *Tour Scenario* [129], by which the robot interactively collects semantic information. Here is how the Tour Scenario is defined: After the robot product is shipped and unpacked in a home or office setting, a human user takes the robot on a tour so the robot can create a representation of the environment. In this interactive scenario, the robot learns basic information about the surroundings, while following the user. During the tour, the user can point out and identify relevant places and objects which the robot

can utilize for future tasks. It is likely that the a commercial robot product that has the capability of the familiarizing task will include both human-interactive and automatic components (i.e. automatic object recognition). In this thesis we focus on the interactive components.

1.6 Challenges

There are four primary challenges in developing a practical robotic system that navigates using semantic information. From the robot's perspective, the interesting research questions are:

- Where are the people?
- How should I move around humans?
- How can I acquire semantic information about this environment?
- How can I utilize this semantic information to improve navigation?

The challenges laid out by these questions will be addressed for the remainder of this section, and the answers will be discussed throughout this thesis.

1.6.1 Robust People Tracking

In order to be engaged in any kind of meaningful spatial interaction with humans, a robot has to be able to track people around it. Robust tracking of people is necessary to ensure high success rates for HRI tasks.

We use a tracking system that estimates the positions and velocities of humans, using three distinct body part detectors. These detectors are the leg detector, torso detector and upper body detector. Use of multiple modalities is leveraged to increase the robustness of tracking and provide more coverage around the robot. The detections are consolidated into position estimations using a Kalman Filter.

1.6.2 Social Navigation

Path planing for mobile robots has traditionally been considered a shortest-path problem. While this leads to sound solutions and collision-free paths, the behavior of the robot may be perceived as unsafe or unnatural by human observers. When a robot is navigating in an human environment, the most important consideration is the safety of people. Moreover, even if a robot's motions are safe, they can still cause discomfort. For example, sudden appearance of a robot can bother people or cutting in between two people while they are in a conversation may be considered rude behavior. Therefore, social factors should also be taken account along with effectiveness of robot motions. Borrowing the idea from Sisbot [119], we attach personal spaces and social constraints as costs to map cells while planning a path. Our planner also takes into account people's reaction to the robot's motion. Humans routinely use anticipation during navigation. For example, in a hallway encounter, one person may lean towards the left of the corridor, expecting the other to take right. We aim to give this anticipation ability to robots. Furthermore, we develop interactive navigation applications where the task of the robot is to follow a person or to a guide a person to a location.

1.6.3 Interactive Map Annotation

The question of how semantic maps could be useful for various tasks is addressed in Section 1.4. People may want to provide custom labels to landmarks, objects and waypoints instead of generic ones. By using unique labels, ambiguities can be resolved. For example, if an automatic object recognizer is utilized, the robot could be confused if it is given the command "bring my laptop" and there are multiple laptops in the environment. Moreover, the ability to label objects and locations would complement automatic recognizers. For example, if an object recognition system makes an error, a user would be able to change the object's label. Our solution to annotate maps is

to maintain a common ground between a human and and the robot is via the Tour Scenario, as described in Section 1.5.

1.6.4 Situation Awareness for Navigation

Metric maps provide a single type of information: whether a grid cell is occupied with an obstacle or not. While this kind of a simplistic map representation may be sufficient for some navigation tasks, if the robot needs to communicate with human users about its goals, it should have a richer map representation of the environment. The robot can leverage the semantic information presented in the map and increase its usefulness for navigation tasks.

Simply put, Situation Awareness is knowing what is going on around you. It can be achieved in various contexts for robot navigation. Person following with a robot is a widely studied research topic, however in the literature it is not usually considered *why* the robot is following the person and how the task context can be used to increase effectiveness. For example, during the Tour Scenario, the robot may employ a more intelligent following strategy if it knows the user will be pointing at landmarks and places. Furthermore, consider handling of the doors while following. If the robot does not know that it is standing on a doorway, it can block the path for others. If the door is added to the semantic map, the robot can have the opportunity to handle the door passing graciously. Moreover, the robot can utilize the floor layout of the environment to adjust its speed. For example, if the robot is in a region where the chance to encounter a person is high, slowing down may be a sensible action to take.

1.7 Scope and Context

In this section, we provide an overview of the assumptions of our research. At a high level, autonomous robots perceive its environment, reason the situation, act in the world. In the HRI context, the research in perception is focused on improving the robot's ability to "see" its environment, identify objects and humans. The reasoning

part is focused on deciding on "what to do" and "how to do" in a given situation. Acting part is focused on the execution and control of motions, as well as the design of mechanisms. This thesis mainly focuses on the reasoning part while also offering a solution to the perception part.

We assume that we have a wheeled mobile robot platform. Even though some of our algorithms are derived for a non-holonomic robot with differential drive actuation, it is a technical limitation rather than a conceptual one, as they can be applied to mobile robots with different mechanisms with minor changes. We further assume that the robot is self-contained: equipped with on-board sensors, is capable of creating a map of the environment with an existing Simultaneous Localization and Mapping (SLAM) package, and then able to localize itself in this created map as it moves in the environment.

Furthermore, we focus on scenarios where people in the environment are standing or walking, because that's the most common body pose the robot would encounter.

A user can control the robot via tablet or smartphone apps. Similar functionality could be achieved with a speech dialog system, however we chose touch-based interfaces due to their ubiquity and reliability.

1.8 Thesis Overview

1.8.1 Thesis Statement

Non-expert users can effortlessly interact with and control a mobile robot through the use of semantic maps and spatial rules of engagement.

1.8.2 Contributions

- A method for a human to take a mobile robot on a tour and interactively add labeled objects, landmarks and locations to the robot's map using natural deictic gestures and later use these labeled entities as end goals for navigation

- Development of a navigation planner that takes into account social factors, and people's reactions to robot's motions
- Development of a person guidance behavior and its application to indoor wayfinding for blind users
- Development of a person following behavior and its usage for telepresence robots
- Demonstration of situation awareness for person following behavior, targeted at the Tour scenario

1.8.3 Document Outline

The structure of this thesis is shown in Figure 1.

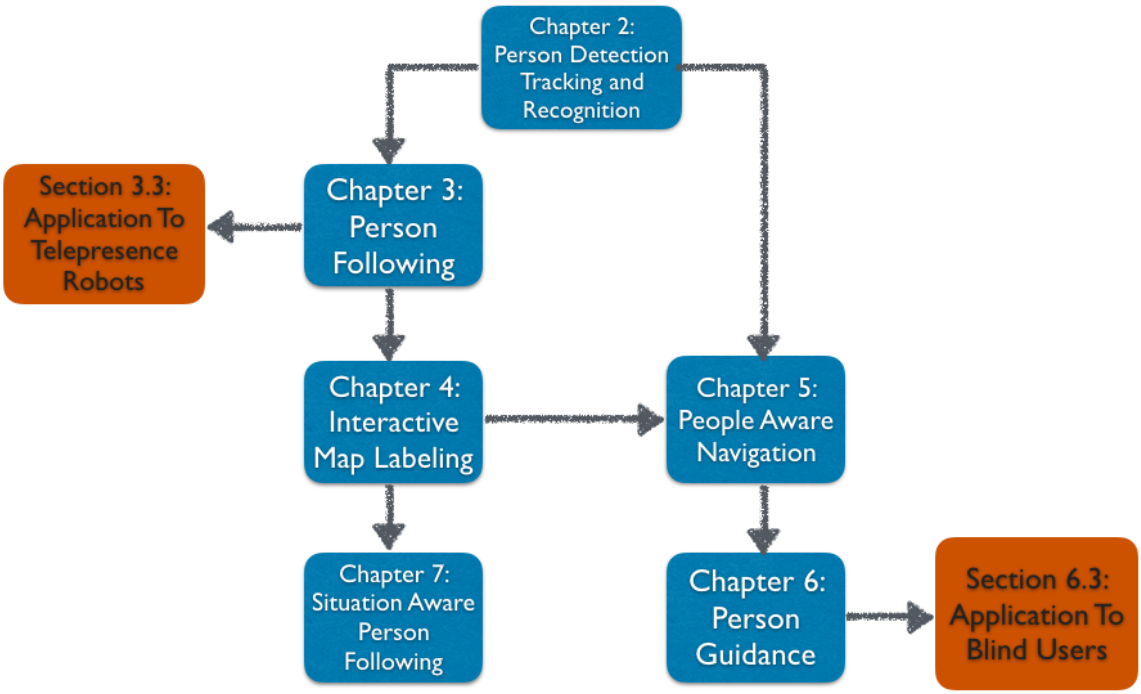


Figure 1: Thesis Outline

The first prerequisite for a socially aware robot is detecting and tracking people. **Chapter 2- Person Detection, Tracking and Recognition** presents our method of leg detection, torso detection and face recognition and how the robot estimates the positions of nearby people using these detections.

When a robot is able to keep track of the positions of nearby people, it can follow a user. **Chapter 3 - Person Following** describes how the robot follows a person. Additionally, an application of person following for telepresence robots is presented in Section 3.3.

If a robot has the capability to follow a person, a user can take the robot on a tour around the house and teach landmarks to the robot. **Chapter 4 - Interactive Map Labeling** describes our semantic map representation, the process of interactively labeling landmarks, the user interface and how to detect pointing gesture targets.

The ability to track nearby people also gives the robot the capability to conduct socially aware navigation toward labeled landmarks. **Chapter 5 - People-Aware Navigation** discusses how we extract goal points from semantic maps, and a people-aware planner for navigating from a start position to a goal position.

If a robot is able to navigate toward a goal position or a landmark, it can also guide people to the same goals. **Chapter 6 - Person Guidance** presents the behavior design of a tour guide robot and how the robot can guide a person to a goal. Furthermore, we present an application of person guidance for visually impaired users in Section 6.3.

The person tracking and following capability, combined with the existence of labeled landmarks, enables the robot to react to social situations during following. **Chapter 7 - Situation Aware Person Following** discusses some problematic scenarios that could be encountered during following and presents our design of robot behaviors for these situations.

Chapter 8 - Conclusion and Discussion discusses the contributions of this thesis, lessons learned during the research program and future work in this area of research.

Appendix A lists telepresence user study survey questions.

Appendix B gives details of the user study on haptic belts for guiding blind

people.

CHAPTER II

PERSON DETECTION, TRACKING AND RECOGNITION

The ability to detect, track and recognize a person is an important prerequisite for human-robot interaction. The challenge is to robustly track humans in the vicinity of the robot considering the robot's movements, sensing capabilities and occlusions. The scope of how much information is needed from the human perception module depends on the task objective. First, the robot should determine if there are people nearby. If the robot senses people around, the robot should find out *where* they are. Representing people as points (x,y) in maps is common practice for robot path planning. If the task requires the robot to face a person, the robot should be able to detect the orientation θ of the person. The robot can further determine *who* the detected person is. Identification of humans is necessary for enabling non-generic service. Finally, the robot can interpret *what* the person is doing by analyzing the motion features and through gesture analysis. Tracking body parts of humans over time give significant information about human activity.

We focus on tracking people who are either walking or standing, as these are the two most common human poses around a mobile robot. Many camera-based full-body or body part detectors have been developed in the literature, reviewed in Section 2.1. We aim to robustly track a person 360° around the robot. However, most sensors have a limited field of view and using only a single detector can lead to a system with a single point of failure. Therefore, for our use cases, a multimodal system is better suited the tracking people using on-board sensors.

Using laser scanners for people detection is a natural choice as state-of-the-art

mobile robots are already equipped with an ankle-height laser scanner that is mainly used for navigation. The laser scanners we used on our robot are Hokuyo UTM 30-LX, which has 270° Field of View (FOV), 0.25° angular resolution, $40Hz$ refresh rate and $30m$ maximum range. We are only interested in detections in close range (less than $5m$). In that range interval, and the accuracy of each laser reading is $\pm 3cm$, which is sufficient for our use cases. The relatively higher accuracy and resolution are the two advantages of laser scanners over cameras and RGB-D cameras. Cameras, on the other hand, have the advantage of providing richer information, which can be used to extract body parts.

After a literature survey in Section 2.1, we give details on our body part detectors in Section 2.2. Each detector produces a point measurement, which is then used to keep a list for person hypotheses. The state of the hypotheses are maintained using a Kalman Filter, explained in Section 2.3. We present our face recognition method in Section 2.4.

2.1 Related Work

Person detection was first addressed by the computer vision community as an object detection problem. Early research on person detection using vision is surveyed by Moeslund [82]. Face detection is a common method for detecting people, with the work of Viola and Jones [137] being the most popular approach. See Zhang [145] for a survey on contemporary approaches on vision based face detection. Another popular topic has been pedestrian detection in crowded scenes Leibe [69] and Tuzel [135].

In the past two decades, laser scanners has been the de-facto sensor for localization and mapping. For this reason, leg detection in laser scans became common practice. Legs are typically distinguished in laser scans using geometric features such as arcs [140] and boosting can be used to train a classifier on a multitude of features [3]. Early works by Montemerlo [83] and Schulz [111] used particle filters for tracking legs

in laser scan measurements. Topp [127] demonstrates that leg tracking in cluttered environments is prone to false positives. Glas [34] uses a network of laser sensors at torso height in hall-type environments to track the position and body orientation of multiple people. Several works used different sensor modalities to further improve the robustness. Carballo [15] uses a second laser scanner at torso level to improve the robustness of detection. Kleinehagenbrock [60] and Bellotto [6] combine leg detection and face tracking in a multi-modal tracking framework. Other examples include combining sound localization and vision [9] and combining RFID tracking and vision [33].

Tracking of the body parts has long been a topic of interest in vision [5, 115]. With the introduction of 3D sensors such as the Velodyne, Swissranger and Kinect, robust tracking of body parts became possible. Spinello [120] trains geometrical features at different height levels in the 3D point cloud for pedestrian detection. Ganapathi [30] estimates body part locations with a probabilistic model. One of the well-known skeleton tracking algorithms is the Microsoft Kinect SDK by Shotton [114], which trains decision forests using simple depth features and a large database. This software is not suitable to work continuously on a mobile robot as it is designed to work on a stationary sensor. In the robotics community, there are efforts to develop skeleton trackers that are better suited to work on mobile robots and in unstructured scenes [13].

Face recognition is a widely used application as surveyed by Phillips [98]. One of the approaches in face recognition uses a set of patch masks for features that doesn't necessarily correspond to eyes, ears or noses [134]. Zhao [146] combines PCA (Principal Component Analysis) and LDA (Linear Discriminant Analysis) to improve the recognition when only few samples are available. There has been some work to identify humans using 3D data, such as the head-to-shoulder signature [59] and body motion characteristics [86]. Biometric person identification techniques, such speaker

recognition [57], 3D ear shape [141] and multi-modal cues [31] have potential to be more accurate than face recognition. However, these approaches are better suited to work in controlled environments.

2.2 Person Detection

In this section, we present our body part detectors, namely leg detection (Section 2.2.1) and torso detection (Section 2.2.2). In addition, we use an implementation of an upper body detector by Mitzel [80], which uses a template and the depth information of a RGB-D camera to identify upper bodies (shoulders and head), designed to work for close range human detection using head mounted cameras.

2.2.1 Leg Detection

A front-facing laser scanner at ankle height is used for leg detection. The output of a laser scanner at each iteration is an array of range measurements, represented in the polar coordinate system. We first convert the range data to Cartesian coordinate system:

$$x_i = \sum_{\phi=\phi_{start}}^{\phi_{end}} r_i \cos(\phi)$$

$$y_i = \sum_{\phi=\phi_{start}}^{\phi_{end}} r_i \sin(\phi)$$

Then we cluster the scan into segments, with the assumption that nearby laser points belong to the same object. Two adjacent distance measurements are considered to be in the same segment if the Euclidean distance between them is below a threshold value. Starting from one end of the array of measurements, a new segment is started if $|r_i - r_{i+1}| > d_{cluster}$. Although some approaches use a variable segmentation threshold that is a function of the range, we use a fixed clustering threshold $d_{cluster} = 0.1m$. The segmentation process results in a set of segments $\mathbf{S}$. A set of geometric features are extracted from each laser segment in $\mathbf{S}$.

In a laser scan, legs can appear in different patterns [127]. We look only single leg and person-wide blob patterns as these two cover all the ways legs can be seen in a laser scan. Depending on the application, we accept either only the single leg pattern or both of the patterns. This is explained in more detail in Section 2.3.

A number of geometric features can be extracted from a laser segment, as detailed by Arras [3]. We use three geometric features: segment width, circularity, and Inscribed Angle Variance (IAV):

1. Segment Width: Measures the Euclidean distance between the first and last point of a segment S_i

2. Segment Circularity: For this feature we use the ratio of the perpendicular distance from the middle point to the line segment that connects start and end points, to the segment width. For example, in a perfect half circle in Figure 2, the circularity criterion is $|\overline{P_0P_n}|/d_{mid} = 0.5$. In case of a laser scan, as can be seen in Figure 3, we again consider the ratio of d_{mid} to segment width. For this calculation, we only use the start, end and middle points of a laser scan as it gives a fast measure on the circularity of the laser segment.

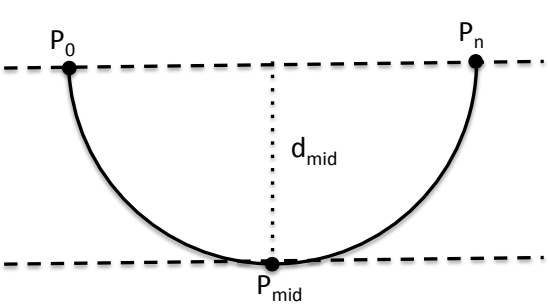


Figure 2: Circularity criterion in a perfect circle is: $|\overline{P_0P_n}|/d_{mid} = 0.5$

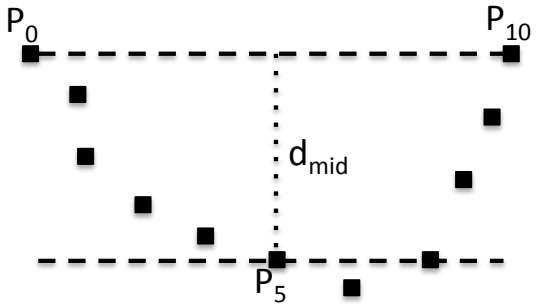


Figure 3: Circularity criterion in a this laser segment is: $|\overline{P_0P_{10}}|/d_{mid} = 0.5$

3. Inscribed Angle Variance (IAV): This feature is originally proposed by Xavier [140], in order to detect circles. For shapes that are not perfect circles but are

similar to circles, IAV feature should be consistent. Laser segments from a leg usually resemble a circle, therefore we use IAV as one of the features for leg detection. As an example, inscribed angles on a circle is shown in Figure 4. As a geometric property of the circle, angles $\angle P_0P_1P_4$ and $\angle P_0P_2P_4$ are equal. IAV for a given set of points is the average of all inscribed angles:

$$IAV_S = \sum_{P=P_1}^{P_{n-1}} \frac{\angle P_0 P P_n}{n}$$

where $IAV_S = 90^\circ$ for a perfect circle.

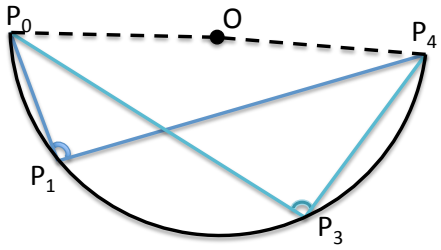


Figure 4: Inscribed angles of an arc are shown in the figure. Inscribed Angle Variance (IAV) is calculated by taking the average of all inscribed angles on a laser segment.

We captured a set of laser scans data while the robot followed a person through an office environment. The following method used for this experiment will be discussed in detail in Section 3.2. About 17×10^3 Single Leg and 0.6×10^3 person-wide blob patterns were manually labeled in the data. In addition, 120×10^3 segments were labeled as 'other'. We first pre-computed the mean and standard deviation of these features, and then used these values for detection. The mean and standard deviations of geometric features for single leg, personwide blob, as well as other segments are given in Table 1.

For every segment S_i in a test laser scan, we first extract the geometric features f_1^i, f_2^i, f_3^i. We then calculate the weighted Mahalanobis distance to the average leg parameters for the each leg pattern:

$$D^i_{mah} = \sum_{j=1}^{n_{features}} \sqrt{w_j \frac{(f_j^i - \mu_j)^2}{\sigma_j^2}} \quad (1)$$

Table 1: Table shows the mean and standard deviation of geometric leg features training set.

Segment type	Width(m)		Circularity		IAV($radians$)	
	μ	σ	μ	σ	μ	σ
Single Leg	0.13	0.03	0.25	0.15	2.23	0.4
Personwide blob	0.33	0.07	0.14	0.09	2.61	0.16
Other	0.22	0.12	0.1	0.11	2.71	0.38

where w_j is the weight for each feature and μ_j and σ_j are pulled from Table 1. The resulting Mahalanobis distance $Dmah_{leg}$ is then compared with a detection threshold. If $Dmah^i_{leg} < Threshold_{leg}$, the segment S_i is considered a detection. $Threshold_{leg}$ defines how many standard deviations away from the average features are allowed. In our implementation, we empirically set the feature weights as: $\mathbf{W}_{leg} = (0.35, 0.26, 0.39)$, with the feature order given in Table 1. For normal operation, we set $Threshold_{leg} = 1.5$, which accounts for about %95 of the detections. If only one person is being tracked, we use a higher threshold. The reasoning behind this will be explained in Section 2.3.

2.2.1.1 Associating Leg Segments

After single leg patterns are detected, we attempt matching the leg segments by determining whether each leg pair are connected. The method described in this section applies to configurations where a RGB-D camera is pointing to the lower body of the human. For each leg segment pair, if both of them are within the FOV of the RGB-D sensor, we use our algorithm to determine whether there is a connectivity between two candidate leg segments. If connectivity is found, then the leg segments pair is qualified to represent a person. If this method is not applicable due to robot configuration, we use solely the distance between leg pairs for association. See Figure 5 as an example result. Figure 6 shows the flow chart of the leg segment association algorithm. Here we describe the steps of this algorithm. First, the centroids each of the two candidate leg segments are found. These points are projected onto the depth

Figure 5: Two person detections are seen in this figure. Our leg segment association algorithm propagates pixels vertically from candidate leg segments and connects leg pairs.

image acquired from the RGB-D camera. At each iteration for each leg segment, our algorithm first propagates horizontally to both directions in the depth image to determine the center pixel, and it propagates 1 pixel vertically ($+z$ direction). If there are no connectivity after a number of iterations, then we conclude that the candidate leg pair does not represent a person. If there is a connectivity at some point, we then assign a likelihood score to the pair. The score is a function of the total vertical propagation height. If this score is higher than a threshold, then the algorithm concludes that the leg candidate segments represent a person. In our experience, we observed that the propagation scoring eliminates most of the false positives.

2.2.2 Torso Detection

For this detector, we used another Hokuyo UTM 30-LX laser scanner, placed at torso height ($1.27m$), directed towards the back of the robot. Our approach relies on fitting an ellipse to laser segments and interpreting the axis lengths. By using the shape of the ellipse, the body orientation of the person is estimated (Figure 7).

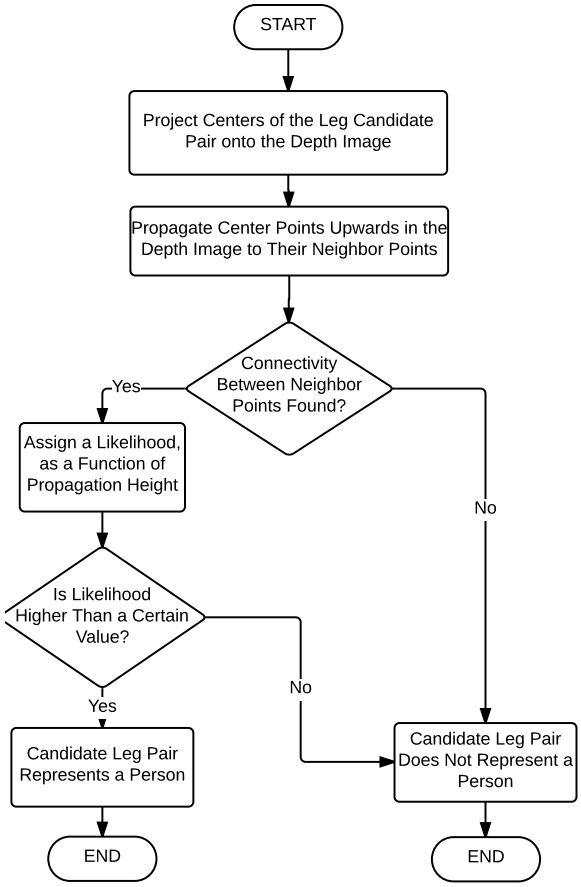


Figure 6: Flow chart for determining if two leg segment candidates belong to a single person.

The first step to detect torsos in a laser scan is to segment the laser scan into clusters. We use the same segmentation technique used for leg detection, as explained in Section 2.2.1. We then fit an ellipse to each laser segment. We use a numerical ellipse fitting method that solves the problem with a generalized eigensystem, introduced by Fitzgibbon [27]. This fitting method is robust, efficient and ellipse-specific, so that even very noisy sensor data will always return an ellipse. Compared to iterative methods, it is computationally more efficient.

To detect a torso in a laser segment, we use the minor and major axis lengths, in addition to the three geometric features introduced in Section 2.2.1 for detection of legs. We collected 450 laser scans while a person stood $2m$ away from the sensor and made a one full turn around. We calculated the mean and standard deviation

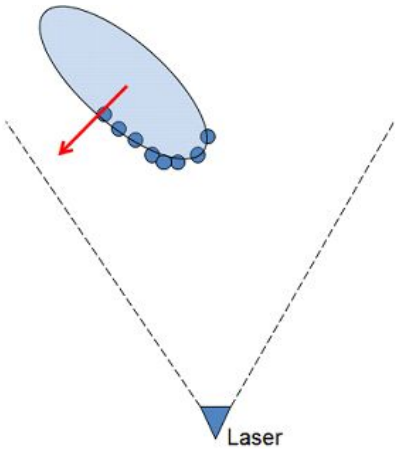


Figure 7: Our torso detector fits an ellipse to the human torso and estimate its position using the ellipse centroid and orientation using axis lengths.

Table 2: Table shows mean and standard deviations of geometric features for a human torso in the training data set acquired by a torso-height laser scanner.

Torso Features	μ	σ
Width(m)	0.44	0.12
Circularity	0.32	0.18
IAV$(radians)$	2.57	0.38
Major axis length(m)	0.39	0.08
Minor axis length(m)	0.17	0.06

of the all five features, which is given in Table 3. For a given laser segment, we find the weighted Mahalanobis distance in Equation 1 to the averaged parameters. If $Dmah^{i}_{torso} < Threshold_{torso}$, the segment is considered a detection. The feature weight constants we used was $\mathbf{W}_{torso} = (0.19, 0.09, 0.35, 0.24, 0.13)$, in respective order given in Table 2. These values were empirically determined, although it is possible that optimizing the feature weights would yield better results.

Figure 8 shows how the torso detection success rate changes with respect to the Mahalanobis Distance Threshold in our training dataset. What is not displayed in the plot is that a higher detection threshold leads to more false positive rates. For normal operation, we set $Threshold_{torso} = 1.25$, which accounts for about %90 detection rate. If the tracker is dedicated to track only a single person, then we use a higher threshold,

$Threshold_{torso} = 2$, to account for about %99 of the detections. The reasoning behind the higher threshold is that false positives would be dismissed anyway if only a single person is tracked.

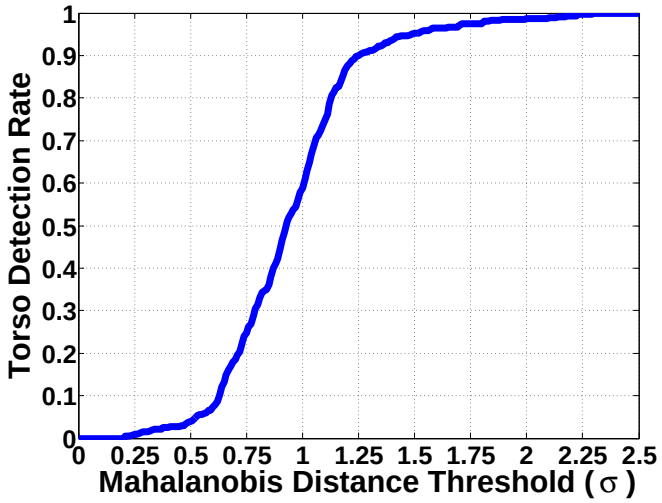


Figure 8: Torso detection rate vs the detection threshold

2.2.2.1 Evaluation of Torso Detection

The ellipse fitting algorithm output is the centroid of the ellipse, and the minor and major axes. We assume the orientation of detected humans is aligned with the minor axis. This provides two solutions: either the front or the back of the person is facing towards the sensor. While this is a significant limitation our current system, one can potentially utilize face detection as will be described in Section 2.4 to estimate if the person is facing the front of the sensor or not.

In order to evaluate the accuracy of the position and orientation estimations of our torso detection method, we collected torso data from 23 people. Subjects were instructed to stand on 4 targets at different distances with 8 different orientations on each target. Experimental setup from the sensor's point of view is shown in Figure 9. For each pose at every target, we logged the position and orientation estimate of the torso detector and compared it with ground truth, which is defined by the markings

on the floor.

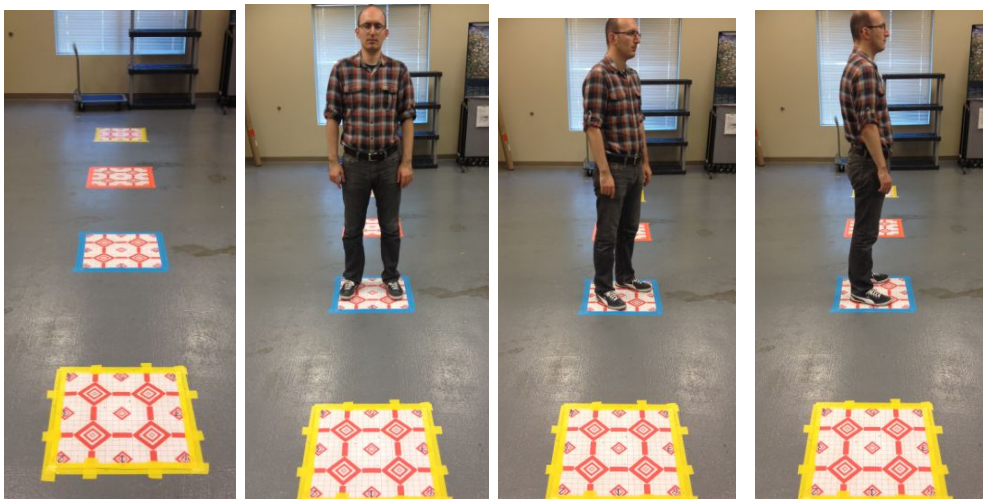


Figure 9: Experimental setup for measuring the position and orientation estimation errors of torso detection. The pictures are taken from the laser scanner's point of view.

Table 3 shows the angular error at every target distance and human orientation with respect to the laser scanner.

Table 3: Mean orientation error of the torso detector with respect to distance from sensor and body pose is shown. Data from 23 individuals is used.

Distance To Laser	N	NE	E	SE	S	SW	W	NW	ALL
$1.0m$	4°	12°	22°	13°	5°	7°	26°	17°	13°
$2.5m$	5°	16°	19°	10°	3°	6°	14°	17°	11°
$4.0m$	4°	10°	30°	16°	7°	11°	21°	17°	15°
$5.5m$	5°	11°	41°	18°	10°	6°	38°	23°	19°
ALL	4°	12°	27°	14°	6°	7°	24°	18°	14.5°

The average position error was about $5cm$ regardless of the distance and the orientation of the human. The average orientation error throughout all the experiments was 14.5°. Error in orientation, however, varied greatly by the pose of the person with respect to the laser scanner. Average error in orientation differed slightly with respect to the distance from the sensor and was the least with 11° when the humans were $2.5m$ away from the sensor. We attribute to the fact that when humans are

closer than $2.5m$ to the laser scanner, it captures more of the arms, which makes the fitted ellipse slightly worse. The orientation of the human with respect to the sensor had a significant effect on orientation error. Least error was achieved when people faced the sensor (4°) or the opposite way (6°). On the other hand, average orientation error was between 24° and 27° when humans were perpendicular to the sensor, because a large portion of the torso was not visible to the laser scanner in that configuration.

2.3 Person State Estimation

The position and velocity of the person can not be determined by direct observation due to measurement noise and false detections. Therefore there is a need for a filtering algorithm in order to estimate the state of a person. Using a state predictor for human movement has two advantages. First, the predicted trajectories are smoother than raw detections. Smooth tracking helps the robot maintain consistent trajectories for high-level applications such as person following (Section 3). Second, it provides a posterior estimate that can be used for data association when there is a lack of matching detections. This allows the tracker to handle temporary occlusions. We use a Kalman Filter [50] to predict the position and velocity of a person. There are other types of filtering techniques available in the literature, such as Particle Filters [54]. Since the results of the person state estimator is used by time-critical higher level applications, the tracker should come up with an estimate in real time. Therefore the choice of using Kalman Filters was motivated by its computational efficiency. Efficient person state estimation also increases the safety of the robot, as the robot can react faster if there are people in close proximity.

Accoding to Hicheur [44], humans tend to maintain a constant speed when they are walking straight and reduce speed while turning. Using this observation, we used constant velocity model which assumes people will maintain their speed. Even though

this assumption is not always true, it provides a simple model without sacrificing too much from tracking performance.

The Kalman filter estimates a process as a predictor-corrector cycle using feedback control. The process has two cycling states: time update and measurement update. Time update projects the state forward by using the current state and error covariance. Measurement update is responsible for the feedback and corrects the previous estimate.

The Kalman Filter is governed by two linear stochastic difference equations:

$$s_k = As_{k-1} + Bu_{k-1} + w \tag{2}$$

$$z_k = Hs_k + v \tag{3}$$

Where s_k represents the process state at time step k, A is the state propagation matrix, B relates the optional control input u, z_k is a measurement, H is the measurement observation matrix. w and v represent the process and measurement noises, respectively, drawn from normal probability distributions with zero mean $N(0, Q)$ and $N(0, R)$.

We define the state of a person s_k at time step k as:

$$s_k = \begin{bmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \end{bmatrix} \tag{4}$$

where (x_k, y_k) is the position and $(\dot{x}_k, \dot{y}_k)$ is the velocity of the person in Cartesian Coordinates. With the constant velocity model, the time update equations are:

$$x_k = x_{k-1} + \dot{x}_{k-1}\Delta t_k \tag{5}$$

$$y_k = y_{k-1} + \dot{y}_{k-1}\Delta t_k \tag{6}$$

$$\dot{x}_k = \dot{x}_{k-1} \tag{7}$$

$$\dot{y}_k = \dot{y}_{k-1} \tag{8}$$

where Δt_k is the time difference from the previous detection. This results in the following Kalman Filter matrices:

$$A = \begin{bmatrix} 1 & 0 & \Delta t_k & 0 \\ 0 & 1 & 0 & \Delta t_k \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{9}$$

A track is lost if there are no detections for a fixed amount of time. At every time update of a filter, if Δt_k is larger than a fixed threshold, the track is deleted from the list.

The reason B vector is zero is that we track people in the world frame and robot motion is already accounted for with robot localization. For this reason, we assume there are no control inputs to our system. The noise matrices we used are:

$$Q = qI_4 \qquad\qquad R = rI_2 \tag{10}$$

where we used $q = 0.02$ and $r = 1.0$ in practice.

Our approach is multimodal in the sense that asynchronous measurements are accepted from different sources as long as they provide a position estimate in the respective sensor frames. Using the latest localization information, this position is converted to the world frame. Before the measurement is accepted as valid, we execute one more check. We check if a new detection is in collision with the static map, and if it is in collision, we reject that detection. The check against the static map is fast and helps reduce false positives in practice. We use Nearest Neighbor (NN) data association [4], which is a reasonable compromise between performance and computational cost.

Depending on the task, a single person or multiple people must be tracked. For example, for following a person or guiding a person, tracking a specific user is sufficient. However, for point-to-point navigation or when searching a specific person, the robot should track multiple people. We examine each case below:

- **Single target tracking:** For some tasks, such as person following, dedicated tracking of a single specific user is required and tracking bystanders is optional. In this case, our goal is to keep tracking the specific user, so we significantly relax the detection thresholds of the detectors. Even though doing so results in more spurious detections, we do not start more than a single track. This approach improves the tracking performance of a single person.

- **Multi-target tracking:** When the robot is navigating from a point to another among people, the robot should keep track of bystanders and respect their personal spaces. In multi-target case, we keep a separate Kalman filter for each tracked person. If a detection is matched to multiple tracks, only the closest track is associated with the detection and the other filters are considered to have no detections for that time step.

2.4 Face Recognition

For certain interactive navigation tasks such as finding a specific person, a robot needs to have person recognition capability. Our person recognition approach uses face recognition and optionally shirt color features. We detect faces in RGB images using the popular face detector by Viola and Jones [137], and use the $eigenface$ method by Turk and Penland [134] for extracting features. With the $eigenface$ approach, faces are represented in a lower-dimensional space. Sirovich and Kirby [118] showed that dimension reduction method Principal Component Analysis (PCA) can be used on face images to form a set of basis features. The main idea of PCA for faces is to find vectors that best account for variation of face images in all training images. These vectors are called $eigenvectors$. Then a face space called $eigenfaces$ is constructed and the images are projected onto this space. Our approach of face recognition has the following steps:

1. A person previously unknown to the robot initiates training.

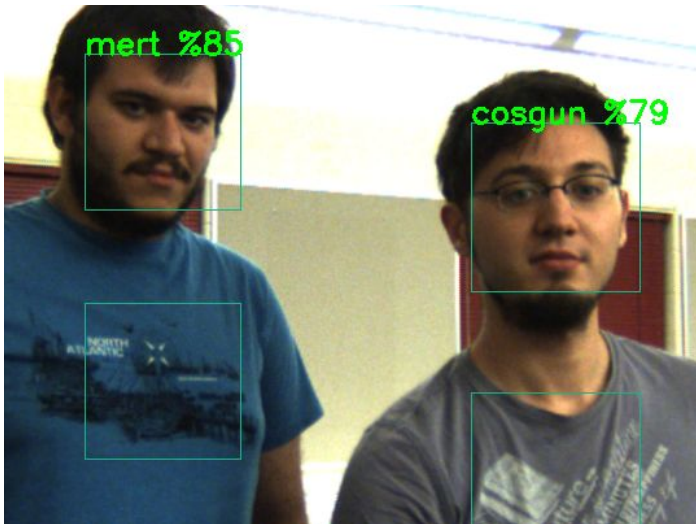


Figure 10: Example results of our person recognition method is shown in the image. We use $eigenfaces$ and PCA face recognition method and optionally shirt color recognition.

2. Robot asks the person to turn his face one side to another, and takes M number of face and shirt pictures

3. Eigenfaces from the entire training set is calculated, and every known face is projected to the corresponding M-dimensional weight $facespace$.

4. After training is completed, face recognition is reactivated.

5. When face recognition is active, a distance value from face recognition and optionally from shirt color matching is received and it is thresholded for a decision.

An example recognition result is shown in Figure 10. Our approach allows new faces to be trained on-the-fly. Using the GUI of the robot, a user can start training and adjust the information in the person database. The person data is managed by a SQLite database hosted locally on the robot.

Shirt color matching can be used when there is not much time between the training and recognition. Activating the shirt color matching should improve recognition and

reduce false positive detections. We take a rectangular region below the face as the shirt area (1.5 times below the the face rectangle size). A histogram with a number of bins is generated for each shirt pattern. Each pixel is pushed into bins according to their normalized RGB value. We calculate a distance value between two histograms using the Earth Mover Distance [105]. In addition, the trained histogram can adaptively be updated at every high confidence detection in order to account for illumination changes. The overall person score is calculated by a weighted average of face and shirt distances.

CHAPTER III

PERSON FOLLOWING

There are many scenarios in which a person following robot could be useful. For example, a robot can carry luggage of travelers in airports, or groceries in a supermarket. Person following is also the enabling capability for interactive label acquisition during the *Tour Scenario* discussed in Section 1.5. For the tour scenario, the robot needs to know how to follow a person before building an environment representation and providing services to the user. There are two properties that a person following robot should have: robustness and social awareness. Typically, a service robot operates in a dynamic and populated environment, therefore the robot must be able to keep track of a single person even when they are temporarily occluded. Multimodal person tracking that is presented in Chapter 2 helps the robot to have an estimate of the user's position. For the person following task, the robot has to track a designated user, and the detection thresholds of detectors are relaxed for robust tracking at the expense of more false positive detections.

For intelligent following, the robot not only has to keep appropriate distance to the user, but also has to recognize *what* the user is trying to do. For example, during the tour scenario, when the user stops, the robot should predict when the user is going to annotate a landmark, and it should come beside the user instead of standing behind. Moreover, the robot should be smarter when passing doors or following a person who is cutting a corner. In order to be able to handle these scenarios, the robot should act beyond purely reactive following behavior. It is desirable for the robot to anticipate what the user is going to do and take appropriate action. The current chapter will discuss the basic following behavior whereas Chapter 7 will

discuss situation awareness and improved behavior for person following.

In this chapter, after referring previous works on person following robots in Section 3.1, we present a basic person following method in 3.2. In Section 3.3, we present an application of person following for telepresence robots.

3.1 Related Work

A robot that follows a person is a widely studied scenario in robotics. A relevant body of work is pursuit evasion [17], in which the target is trying to evade the follower. In our applications, we assume that the target user is cooperating with the robot.

In one of the earliest works in this area by Sidenbladh [116], robot keeps the person centered in the camera image using a P controller. Prassler [99] also offers a reactive approach using the *Velocity Obstacles* concept, which uses the velocity of the target to find allowable velocities of the robot that guarantees avoiding collision if both the target and the robot moved with constant speed for future steps. The approach is applied on a wheelchair, however social constraints were not considered. Gockley [35] observed how people walk together. It is reported that partners who were conversing tend to look forward with occasional glances to each other. Ohya [91] presents a following method that escorts a target on the side while avoiding obstacles. It was assumed that the target would move with the same acceleration and velocity. Murakami [87] presents a method to first estimate the sub-goal of the leading person and then following as if the robot knows the goal. Park [96] models the problem as a control problem and offers an algorithm based on Model Predictive Control. Miura [81] employs randomized tree expansion and biases the calculated paths towards a sub-goal, which is the current position of the person. Hoeller [45] adopts the virtual targets idea and selects a goal position in a circular region around the person. Stein [122] proposes choosing and following a leader to handle navigation in crowds.

Some of the relevant works considered the social side of the interaction. Gockley

compared two elementary following methods in [36]: direction following, where the robot always attempts to drive towards the tracked person, and path following, in which the robot follows the exact path the person took. It was shown that direction following behavior was perceived as more human-like and natural than path following. Yuan's approach [142] switches the following behavior between parallel, direction and path following depending on the layout of the obstacles. Zender [143] emphasizes situation awareness for following and studies handling of doors and corridors. To handle the doors, the robot increases its following distance and that leads the robot to wait for a while. Following in a corridor is handled with an approach similar to Pacchierotti [94], and the robot's speed is adjusted. Loper [72] presents a system that is capable of responding to verbal and non-verbal gestures and following a person. Granata [37] presents behaviors such as going towards, following and searching a user. Ota [92] touches upon the recovery functions whenever the robot loses tracks of the leading person.

3.2 Basic Person Following

In this section, we describe our basic person following method, that is the default following mode. When the following behavior is initiated from a higher level process, first the guide person must be tracked. The robot looks for the closest person in the vicinity of the robot (within $2m$). If no person is detected for a period of time, then the command is invalidated. We use the approach described in Section 2 to detect and track users.

In the basic person following mode, the robot has three different strategies depending on the distance towards the followed person. The distance to the user is calculated as the distance from the center of the robot base to the person's current location estimation. We used Hall's characterization of personal spaces in order to determine the distance limits. The three distinct zones and corresponding robot

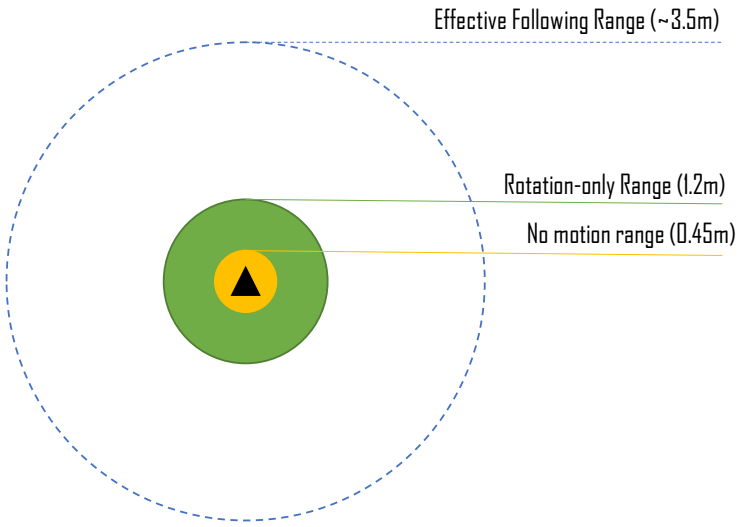


Figure 11: Overhead sketch of the robot and relevant ranges for person following. Robot is represented as the triangle in the middle.

behavior are defined in Hall's work [39] as follows:

Intimate Zone $[0 - 0.45m]$**:** The person is very close to the robot, therefore any motion of the robot may be potentially unsafe. The robot comes to a complete halt in this zone ($v = 0, w = 0$). A full stop also allows the user to safely interact with the on-board tablet GUI.

Personal Zone $[0.45 - 1.2m]$**:** When the robot is in the personal zone of the user, the robot stops and rotates towards the followed person ($v = 0, w = w_{rotation}$). The rotational velocity is determined with a P controller, with the error term defined as the difference between the robot's current orientation and the tracked person's orientation. The rotational speed is capped at a fixed value, so that the person feels comfortable with the rotation speed of the robot.

Social Zone $[1.2m - 3.5m]$**:** In this range interval, the robot executes the main following behavior. At every time step, a goal pose that is $1.2m$ away from the and headed towards the user is calculated (see Figure 12 for an illustration). A collision-free path is found and the robot executes this path until a new measurement from the target person is received. The path is found using Dynamic Window Approach

(DWA) local planing method. We use the ROS implementation of DWA for the basic following behavior.

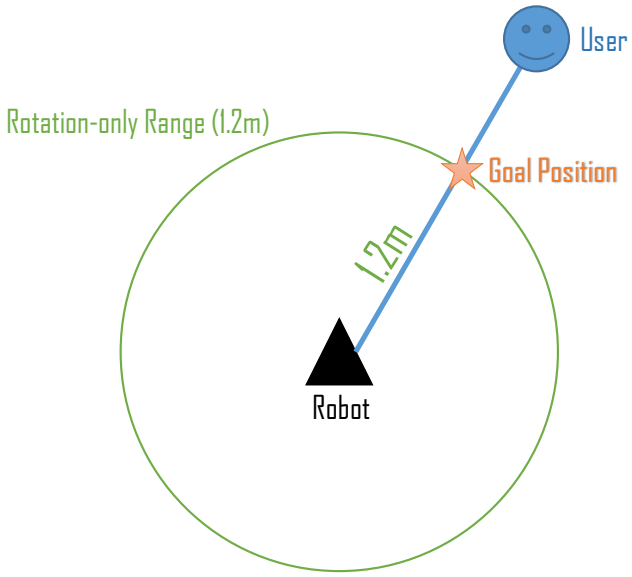


Figure 12: An illustration of how the goal position is calculated when the user is in the social space $[1.2m - 3.5m]$.

Sometimes it is inevitable that the person tracking system loses the target, particularly when the person is consistently faster than the robot or the person goes outside the range of the sensors (further than $\sim 3.5m$ in our case). When this happens, the robot will attempt to go to its last calculated goal position and look for the person. With this behavior, the robot attempts to keep up with the lost person as far as possible with the hope that the person will re-appear in the vicinity of the last seen position. After the robot reaches this goal, it stops and waits for an amount of time. If the user is saved to the database, or the robot already knows that he/she is in the database, then the face recognition system that will be described in Section 2.4 is activated. Otherwise, the robot continues following the closest person that appears in this position. If no person is detected within a fixed amount of time (5 seconds in our implementation), then the robot declares that the person is lost and starts listening for a new following command.

3.3 Application To Telepresence Robots

A telepresence robot can be described as *Skype on wheels*, where a remote user teleconferences in a physical environment while having the motion control of a robotic system. Telepresence robots constitute a promising area in the consumer robotics industry as evidenced by recent start-up companies working on telepresence products. However, currently all the telepresence robots that are available in the market are controlled by manual driving by the remote controller - usually via the keyboard or a joystick. In this section, we present an implementation of person following on a telepresence robot and a user study that evaluates effect of having person following capability on a telepresence robot.

Telepresence robots are a level above video conferencing since the robot is used as the communication medium and the remote user can now control the movement. Therefore, the spatial interaction between people and a telepresence robot in social situations is worth investigating. One of those situations is moving with a group of people. In an effort to analyze the spatial and verbal interaction, we focus on engagement with one person where the remote user interacts with the person while following him/her in a corridor. This is situation is very likely to happen in office environments, for example when the remote user is having a discussion with a coworker while walking to his office after a meeting. As telepresence robots become more common, there will be need to have the functionality of autonomous following of a person so that the remote user doesn't have to worry about controlling the robot.

We evaluate our system by conducting a user study, where there are two following conditions:

1. Manual Person Following: Robot is controlled with an Xbox controller
2. Autonomous Person Following: Following is initiated by clicking on a user in RGB-D image

The aim of the user study is to measure how remote users like using the autonomous following feature compared to the manual. For the study, the remote user has a task that consists of listening to a passage the guiding person reads, and answering related questions after the interaction. We also observe subjects' experiences using the system, get useful feedback to pinpoint future challenges that can be helpful designing new generations of telepresence robots.

3.3.1 Robot Platform

Figure 13: The telepresence robot platform we used for our experiments.

The system described in this paper is implemented on an experimental telepresence robot platform shown in Figure 13. The robot has a differential drive base and can be used for about 8 hours with full charge. The remote user connects to the robot via wireless internet and communicates with others using Skype. The robot is also equipped with an omni-directional microphone and a high-end speakerphone. A wide-angle camera is placed on top of the monitor and tilted slightly downward to help the remote user to see the floor, robot base and people's faces at the same time. A Kinect Sensor is also placed above the monitor. Person following is initiated using

the user interface shown in Figure 14.

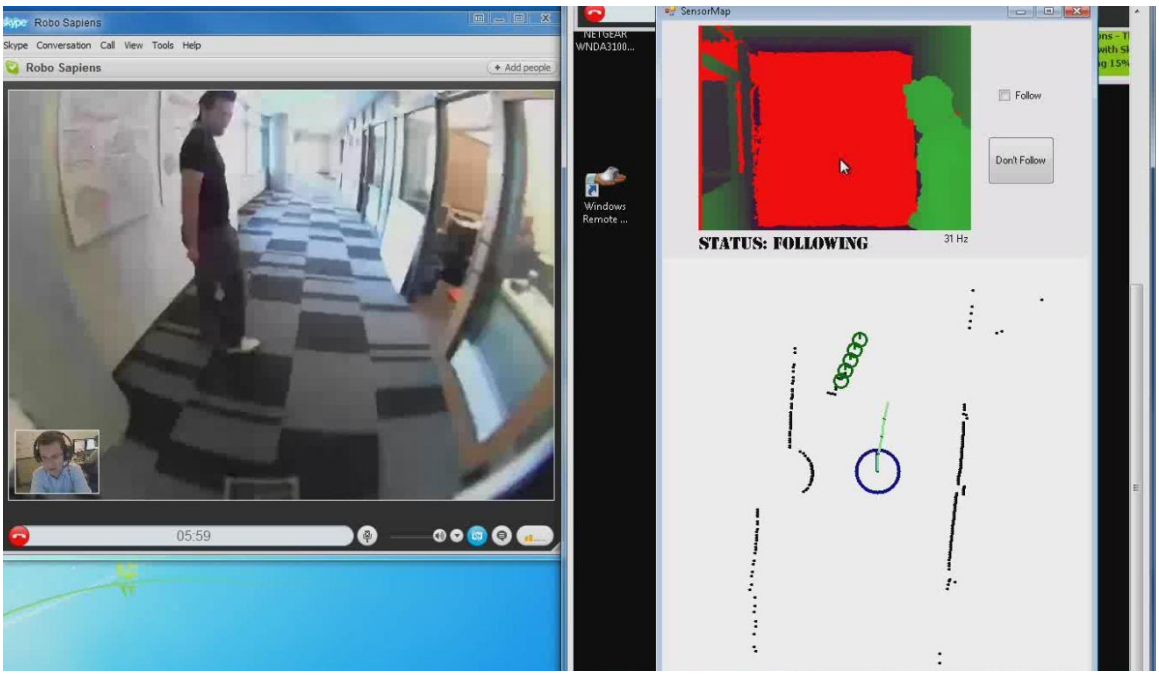


Figure 14: User Interface of the robot for the remote user.

A modified version of the local planner used in Section 5.5.2 is used for person following. A utility function consisting multiple factors, including the respective position to the person, is optimized over multiple steps using Breadth-First Search. Details of this path planning method can be found in [21].

3.3.2 User Study

In this study, remote user is the subject and the followed person is the experimenter. To investigate the effectiveness of using autonomous person following for an interaction task, we ran a controlled experiment and varied manual vs. autonomous following within subjects.

Design: The experiments were conducted in working hours and bypassers were free to walk across the experiment area or talk. The subjects were given the task of following the experimenter through the course for a lap and listen to the passage

he is reading. In the first run, the subject used one of the autonomous following or teleoperation methods to follow a person and complete the lap. In the second run, the subject used the other method. At the end of each run, the subject was asked to complete a 4-question quiz about the passage. At the end of both runs, the user was asked which method he/she will prefer over the other for this type of a scenario. The exact questionnaire can be found in Appendix A. The passages and quiz questions were taken from Test of English as a Foreign Language (TOEFL) listening section examples. The passages were chosen so that they are at a similar difficulty level. The time it takes to read a passage corresponded approximately to the same time a lap is completed. We also asked numbered 7 point Likert scale questions, administered after each run, about how *Understandable* the experimenter was, if the UI was *Easy to Use*, if the robot exhibited *Natural Motions*, how *Safe* the remote user felt, if the subject was able to *Pay Attention* to the passage, how *Fast* the robot was and how much *Fun* the subject had.

Participants: 10 volunteers participated in the study (6 male and 4 female between the ages of 25-48). 5 of the participants had little knowledge, 4 had average knowledge and 1 had above average knowledge on robotics. The participants weren't gamers: 4 participants never played console games, 4 played rarely, 1 sometimes played and 1 often played. 6 of the participants often used video conferencing software, while 2 sometimes and 2 rarely used. 9 of the participants were not native English speakers and all of them had taken the TOEFL before.

Procedure: The participants were first greeted by the experimenter and instructed to complete a pre-task questionnaire regarding their background. The robot was shown to the participant and basic information about its capabilities was told. The experimenter explained the task while walking with the participant in the experiment area and showing the course to be followed. Participants were told that they should

stay close to the experimenter while he is walking and there will be a quiz regarding the passage afterwards. The participant was informed that there are 2 operation modes: manual and autonomous person following.

Before the experiment started, the participant went through training for about 15 minutes. First, the participant learned the basic controls for the Xbox controller when he/she was nearby the robot. Then the participant was taken to the remote station, which was about 20 meters away from the corridor area. The participant was informed about the UI and was shown how the autonomous following is activated. Then a test run was executed, where the remote user followed the experimenter via teleoperation and had a conversation.

After the training, the actual run was executed using one of the manual or autonomous methods. We switched the starting method for every other experiment in order not to bias the subjects' opinions about one particular method. When the lap was completed, first the passage quiz, then the survey questions were answered by the subject. Then, the second experiment using the other method was executed and the second passage quiz and survey questions were given to the subject. As the last question, the subject was asked to state his/her method of preference. Lastly, the participants were debriefed about the study and engaged in a discussion.

Measures: We had three measurement criteria to compare manual vs autonomous following:

1. Number of correct answers to passage quizzes: Assuming the standardized TOEFL exercises were of same difficulty, we ran a paired $t-test$ on two groups of autonomous and manual.

2. Survey questions: We ran a paired $t-test$ using 7-point Likert Scale on each of the seven questions.

3. Preferred Method: We looked at which method subjects chose over the other one.

Results: Out of the 4 quiz questions, the correct answers for autonomous group ($\mu = 2.9$, $\sigma = 0.9$) were more than the manual group ($\mu = 2.2$, $\sigma = 1.2$) but the statistical difference was not statistically significant ($t(9) = 1.48$, $p = 0.17$ on $t - test$).

Table 4 summarizes the survey results. For *Understandable* and *Fun*, the scores slightly favored autonomous method but the difference wasn't statistically significant. Manual method was found to be easy to use ($\mu = 5.0$, $\sigma = 2.2$), but the UI for autonomous method (clicking) was found to be marginally easier ($\mu = 6.5$, $\sigma = 0.9$), ($t(9) = 2.13$, $p = 0.06$). The motions of the robot was found to be significantly more *Natural* to have a conversation for autonomous method ($\mu = 5.4$, $\sigma = 1.0$) than manual method ($\mu = 3.5$, $\sigma = 1.9$), ($t(9) = 2.52$, $p = 0.03$). Participants thought they were able to *Pay more Attention* to the passage the experimenter is reading when the robot was following the him autonomously ($\mu = 5.3$, $\sigma = 1.8$) compared to manual method ($\mu = 3.4$, $\sigma = 1.5$) and the statistical difference was significant ($t(9) = 2.63$, $p = 0.02$). Participants have found the autonomous method ($\mu = 5.1$, $\sigma = 1.7$) safer than manual method ($\mu = 2.3$, $\sigma = 1.4$) and there was a significant difference between two groups ($t(9) = 3.09$, $p = 0.01$). The speed of the robot was found to be neither fast nor slow for both methods ($\mu = 3.9$, $\sigma = 0.3$) and ($\mu = 4.3$, $\sigma = 0.8$).

All 10 subjects chose autonomous person following over teleoperation for this task.

3.3.3 Design Implications

Our user study showed that a person following behavior is desirable for telepresence robots when there is interaction. The follow-up discussions also agreed with the survey results, as one subject (R10) stated: *"It just gives me more focus and concentration."*

Table 4: Survey results of the user study for person following for telepresence robots. Table displays survey question average and standard deviations for the two conditions: Autonomous Person Following and Manual Person Following.

Question	Autonomous		Manual Drive		$t-test$	
	μ	σ	μ	σ	p	t
1. Understandable	4.0	1.5	3.6	1.7	0.47	0.73
2. Easy UI	6.5	0.9	5.0	2.2	0.06	2.13
3. Natural Motion	5.4	1.0	3.5	1.9	0.03	2.52
4. Safe	5.1	1.7	2.3	1.4	0.01	3.09
5. Pay Attention	5.3	1.8	3.4	1.5	0.02	2.63
6. Fast	3.9	0.3	4.3	0.8	0.10	-1.8
7. Fun	5.3	1.5	5.1	1.7	0.66	0.45

Below, we list our observations and implications for future research and design for telepresence robots:

Motor Noise: Even though the motors on the robot were relatively quiet, 8 out of 10 participants expressed that the motor noise made communication harder. This justifies the close scores we collected in the survey question asking if the subject was able to understand what the experimenter was saying. (R8) was disturbed by the noise: *"When I was driving, it was always this constant sound. It was worse for the autonomous one. It was constantly adjusting and compensating for the movement."* On the other hand, (R5) found the motor noise useful: *"I actually like it because it gives me the feedback whether I'm driving faster or slower. It also gives me a little bit feeling of life."* Thus, although excessive motor noise should be avoided, some noise might be useful.

Wireless Connection: Second most cited problem for video conferencing was the video quality and time lags. (R8) clearly expressed why it was hard to walk with the experimenter using the manual method: *"The frame rate drops all of a sudden and you have no choice but to stop."* Another subject (R9) made use of the displayed sensor data when the video conferencing quality went bad: *"Because of the lag, I just switched to the Kinect (depth image) and the overhead view (laser)."* This was

possible because the wide angle camera image was coming from Skype whereas sensor displays were received from the Windows Remote Assistance. Clearly, a big challenge for telepresence systems is to deal with wireless connection problems.

Natural Interaction: Even though the participants thought the motions of the robot were natural to have a conversation ($\mu = 5.4$, $\sigma = 1.0$), some didn't feel it was a natural way to communicate. As seen in Figure 13, the screen displaying the remote user's face is flat and it introduced problems when the robot was traveling side-by-side to the person. (R5), when asked about walking side by side: *"..we don't have face-to-face. It is not really a conversation."* This raises design considerations on how the remote user's face is brought out. One of the subjects (R5) discovered that the microphone characteristics are different than human hearing: *"I don't have a distance sense if the experimenter is further away or close. If you have the fading audio, then I'll immediately notice."* Whether a telepresence robot should exhibit the same characteristics of human perception or not is an open question and needs further investigation.

Assisted Teleoperation: Telepresence robots should possess a layer to assists the remote user to avoid obstacles and collisions. *Safety* ratings for the manual method were very low ($\mu = 2.3$, $\sigma = 1.4$) and (R8) expressed the concern: *"I was especially worried about running into the experimenter."* This suggests that scenarios involving interaction would demand more attention of the remote users. The teleoperation should also be intuitive and be similar to driving modalities that people are already used to. (R4) stated: *"I was thinking about Manual mode compared to driving a car."* before suggesting *".. maybe something like a cruise control might be good."*

Gaming Experience: Since the robot was controlled by a gaming console controller, some participants likened the manual mode to gaming. (R9) said: *"Manual is like playing video games."* and (R5) said: *"I don't play video games so controlling*

those consoles is not natural to me." Thus, it is possible that gamers are less likely to have trouble driving the robot. The same observation was made by Takayama [124].

Long Term Interaction: None of the subjects who participated in our study had used a telepresence robot before. (R6) justified the inability to use the manual method: *"Maybe if I have some more practice for about several hours of driving the robot, I can use manual as well as autonomous."* (R8) on having fun using teleoperation: *"It was fun because it was the first time I did it but I can imagine that over time, I'll get bored of it."* The *Fun* question in the survey received similar scores for autonomous and manual, possibly because using a telepresence robot was a new experience for the subjects. Studies regarding long term interaction for telepresence robots can yield interesting results, as in [68].

Error recovery: When the person was lost during following, the UI displayed a text that the person was lost so that the remote user can re-initiate the following by clicking on the person. None of the subjects complained about the robot losing the person. When asked explicitly about the robot losing the experimenter, (R10) answered: *"That's not a big deal in comparison to me driving the robot."* Therefore, applications developed for telepresence robots can take advantage of the human being in the loop and does not have to be error-free for deployment.

3.3.4 Discussion

User studies showed that autonomous person following is a desired capability for telepresence robots and it was favored over direct teleoperation for an accompanying task. Autonomous following was found to be safer, easier to use and helped the remote users to pay more attention to the conversation instead of the robot control. From the experience earned from user studies, there are still interesting challenges to explore in terms of human-robot interaction for telepresence robots.

CHAPTER IV

INTERACTIVE MAP LABELING

Robots that co-exist with humans will need to accept commands from human users. As mentioned in Chapter 1, the robot keeps a metric map for autonomous navigation. Requiring users to understand the robot's internal representation of the world and to provide goals in terms of metric coordinates may not be a convenient way of interaction. World representation of robots should enable effortless communication between users and robots. For example, it is not easy for someone to quickly interpret the floor plan and provide a goal with explicit coordinates $(1.2, 4.5, 0.0)$. Maps should include spatial information to support tasks that requires more than simple obstacle avoidance. For example, if the task involves interaction with planar surfaces, landmarks, objects, or people; a robot's map should be able to represent these information. For these reasons, adding semantic information to robot maps is essential. There are several methods to support the annotation of entities in a robot map. For example, while the robot is building its representation of the environment, it can recognize objects or landmarks such as doors, tables, rooms and automatically add these features to its map. Even though such a system would be useful, it may wrongly label some objects. In that case, the correct label can be provided by a human with an interactive system. Custom labels would also allow unique annotations such as *"John's Room"*.

We use a map representation that includes objects, 2D waypoints, 3D polygons that defines the boundary of planar surfaces along with user-appointed labels. We provide details about waypoint representations in Section 4.2.1, planar surfaces in Section 4.2.2 and objects in Section 4.2.3. To add labeled landmarks and objects

to the semantic map, we use an interactive system that uses a combination of a graphical user interface (GUI) and natural pointing gestures. Using the GUI, users can command the robot to follow them, add waypoints along the robot's path and add landmarks or objects to the map by pointing at them and entering a label. Our GUI is described in Section 4.3 and our approach to determine the pointing gesture targets is presented in Section 4.4.

4.1 Related Work

One of the key concepts in semantic mapping is establishing a *common ground* [19]. By referring to same structures and objects with the same reference, the communication about labeled entities is grounded.

One of the closely related works to our interactive labeling approach Human Augmented Mapping, introduced by Topp and Christensen in [126] and [128]. This approach involves labeling two types of entities: regions and locations. Regions are meant to represent areas such as rooms and hallways, and serve as containers for locations. Locations are meant to represent specific important places in the environment, such as a position at which a robot should perform a task. This approach was applied to the Cosy Explorer system, described in [144], which includes a metric map, a topological map, as well as detected objects. While our goal is similar, we use a different map representation, method of labeling and interaction design. Kuipers [67] introduced Spatial Semantic Hierarchy, which is a method of organizing semantic information for spatial regions. Another application using semantic maps is direction following, studied by Kollar [61]. This work describes a system capable of following directions in natural language, i.e. "go past the computers".

4.2 Semantic Maps

Semantic mapping aims to create maps that include various types of semantic information to allow the robot to have a richer representation of the environment. In

this section, we briefly present the semantic information we use to enrich the occupancy grid maps: waypoints, planar landmarks and objects. Although we describe three types of entities for storing semantic information, it is possible to include more features.

4.2.1 Navigation Waypoints

We define navigation waypoint as a navigable pose represented in the map frame. A pose (x, y, θ) represents the position and orientation of with respect to a coordinate frame. We will refer a navigation waypoint simply as a waypoint for the remainder of this text. Using the GUI, a user can save the robot's current pose along with a label, assuming that the robot is properly localized in the map. This method is the most straightforward way to save a pose and use it later for a task. When the robot is asked to navigate to a labeled waypoint, the goal of the robot is simply the saved pose.

Even though the waypoint method is easy to use and practical, it has shortcomings. It fails to capture the shape and extent of the structure designated by the label, which might be important for some tasks, such as object search. Moreover, point based references are not sufficient to represent a region or volume in a map, such as a hallway or a room. For example, if the user wants to save hallway as a landmark, it is useful to have a representation of the location and the extent of the hallway. Another example is that the user can label a coffee table, which is a movable object. Instead of saving a fixed coordinate location for the landmark, robot can save the planar surface and can potentially have a model to handle moving landmarks. This gives the robot robustness in its operations and a method to potentially use the planar surfaces as features in localization [133].

4.2.2 Planar Surfaces

In this type of semantic information, we keep track of a set of observed planar surfaces as part of the map representation. We use a front-facing RGB-D camera to acquire the data. Planes are extracted from the point cloud by an iterative RANdom SAmple Consensus (RANSAC) method, which allows us to find all planes meeting constraints for size and number of inliers. A clustering step is performed on extracted planes to separate multiple coplanar surfaces, such as two tables with the same height, but at different locations. We make use of the Point Cloud Library (PCL) [106] for much of our point cloud processing. Planes are represented in the hessian normal form. Since the observed planes do not have infinite extent, we bound the observed area with a polygon - in this case, the convex hull of all the observed points on the plane. The convex hull is accompanied with a user-provided label.

In order to label a planar landmark, the user goes through the following sequence:

1. User activates basic person following behavior (Section 3.2) and comes near a planar surface of interest.

2. Robot adjusts its pose so it can perceive both the surface and the user.

3. User enters the label using the GUI and activates pointing gesture recognition

4. User performs a pointing gesture towards the landmarks of interest

The act of labeling a planar surface with a label "Table" from the view of the RGB-D camera planted on the robot is shown in Figure 15. After the labeling process, the planar landmark is added to the map representation.

4.2.3 Objects

Our approach allows interactively building object models for selected objects, and then annotate these models with a label. This enables the robot to quickly build a

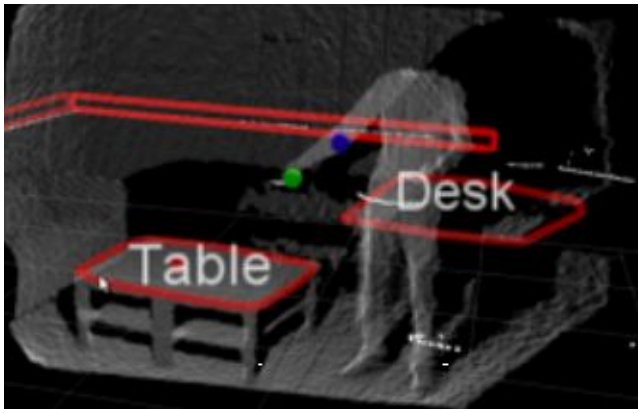


Figure 15: A user is pointing at a table to add it to the semantic map

small object database of the specific objects it needs to interact with. The interactive system works as described in the previous section. Different from labeling planar landmarks, however, we keep a bag of features for each object in the database. Once a user has performed a pointing gesture to label an object, the robot moves to a favorable position and activates its its pan-tilt unit to aim the sensors at this location. Object segmentation is performed using point cloud data from an RGB-D sensor. First, the large planar surface corresponding to the table is detected. This is removed from the point cloud, and point clusters above this are detected. The robot calculates the likelihood of each object being the target, as will be explained in Section 4.4 and confirms with user if ambiguity is detected. After the target is confirmed, the cluster points are projected into the camera image, and are used to generate a region of interest. SURF features are found in the region of interest, and are stored as an object model along with the provided label.

4.3 Graphical User Interface

We developed an on-board tablet interface to enable interaction between the user and the robot. The interface has been implemented on a Nexus 7 Tablet running on Android Operating System. The tablet is equipped with WiFi and talks with the robot using a TCP-based server-client communication model. The client must have

the the IP address of the server machine to establish the connection. The messages are sent back-and forth as via XML files. The XML files are serialized and de-serialized using *libXML++* package for *C++* (server) and *XMLPullParser* (client) for Android OS. With the tablet interface, a user can do the following:

- Add/Delete a Labeled or Generic waypoint
- Label a Planar Surface or Object
- Initiate Person Following
- Initiate Person Guidance to a Labeled Planar Surface/Object/Waypoint
- Stop Robot
- Change IP Address and Port of the TCP connection

A screenshot of our tablet user interface is shown in Figure 16.

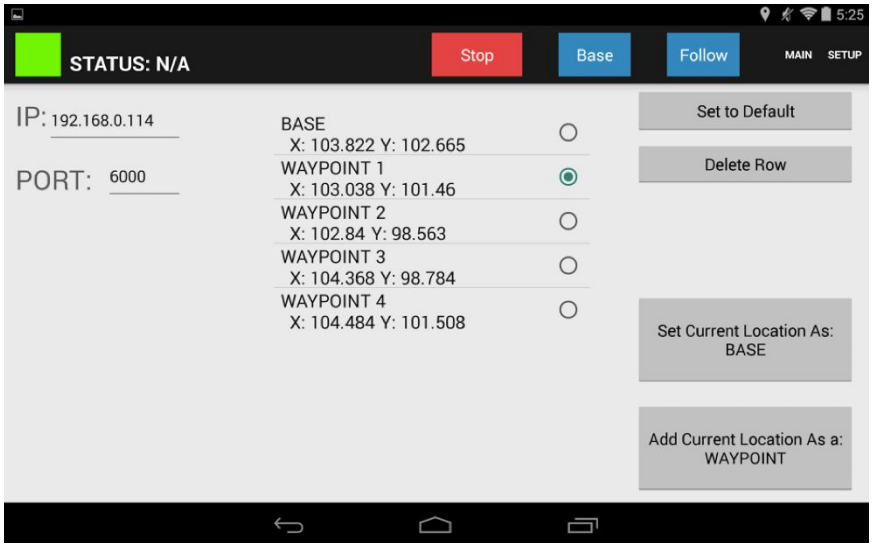


Figure 16: The GUI that runs on the robot

4.4 Pointing Gestures for Human-Robot Interaction

A natural way to refer to objects and landmarks is to point at them. In this work, we analyze the performance of our pointing gesture recognition, and present an uncertainty model that enables us to reason about ambiguity in pointing gestures, when gesture targets are too close to one another. We model the uncertainty of pointing gestures in a spherical coordinate system, use this model to determine the correct pointing target, and detect when there is ambiguity. Two pointing methods are evaluated using two skeleton tracking algorithms: elbow-hand and head-hand rays, using OpenNI NITE and Microsoft Kinect SDK [113]. A data collection with 6 users and 7 pointing targets was performed, and the data was used to model users' pointing behavior. The resulting model was evaluated for its ability to distinguish between potential pointing targets, and to detect when the target is ambiguous. An example scenario is shown in Figure 17.

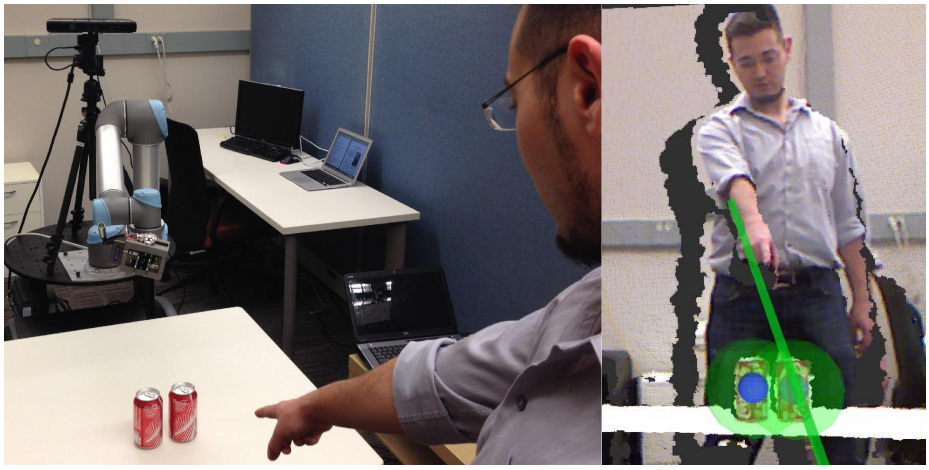


Figure 17: (Left) Our approach allows a robot to detect when there is ambiguity on the pointing gesture targets. (Right) The point cloud view from robot's perspective is shown. In this demonstration, both objects are identified as potential intended targets, therefore the robot decides that there is ambiguity.

4.4.1 Related Works

Pointing gestures are widely used in Human-Robot Interaction applications. Examples include interpretation of spoken requests [147], pick and place tasks [10], joint attention [25], referring to places [42] or objects [110], instructing [76] and providing navigation goals to a robot [101].

Early works on recognizing pointing gestures using stereo vision utilized background subtraction [18, 48, 49]. Other popular methods include body silhouettes [51], hand poses [47], motion analysis [78] and Hidden Markov Models [139, 7, 70, 88, 24, 2]. Matuszek [79] presented a method for detecting deictic gestures given a set of detected tabletop objects, by first segmenting the users' hands and computing the most distal point form the user, then applying a Hierarchical Matching Pursuit on these points over time.

After deciding if a pointing gesture occurred or not, an algorithm must estimate the direction of pointing. This is typically done by extending a ray from a body part to another. Several body part pairs are used in the literature, such as eye-fingertip [51] and shoulder-hand [46]; with the two of most commonly used methods being elbow-hand [101, 11, 10] and head-hand [7, 110] rays. Some studies found that head-hand approach is a more accurate way for estimating pointing direction than elbow-hand [100, 25]. Recent works made use of skeleton tracking data in depth images [100, 10, 101]. Other approaches, such as measuring head orientation with a magnetic sensor [88] and pointing with a laser pointer [16, 52] is reported to have a better estimation accuracy than the body part methods, but require additional hardware. We prefer not to use additional devices in order for the interaction to be as effortless as possible.

Given a pointing direction, several methods have been proposed to determine which target or object is referred by the gesture, including euclidean distance on a planar surface [16], ray-to-cluster intersection in point clouds [10, 100] and searching

a region of interest around the intersection point [110]. Droeschel [24] trains a function using head-hand, elbow-hand and shoulder-hand features with Gaussian Process Regression and reports a significant improvement on pointing accuracy. Some efforts fuse speech with pointing gestures for multi-modal Human-Robot Interaction [2, 63].

To our knowledge, only Zukerman [148] and Kowadlo [63] considered a probabilistic model for determining the referred object for a pointing gesture. In their approach, the probability that the user intended an object is calculated using the 2D distance to the object and the occlusion factor. Objects that reside in a Gaussian cone emanating from the user's hand are considered as candidates in the model. The approach is implemented in [148], where it is reported that due to high variance of the gesture recognition system, the Gaussian cone typically encompassed about five objects in cluttered settings. Our work addresses the confusion in such settings. In contrast to their work, we measure Mahalanobis distances to potential target objects using a prior error analysis.

4.4.2 Pointing Gesture Representation

4.4.2.1 Pointing Gesture Recognition

Our approach to pointing gesture recognition is to use a third party skeleton tracking package as implemented by OpenNI NITE 1.5 (OpenNI) or Microsoft Kinect SDK 1.5 (MS-SDK). Skeleton tracking software produces 3D positions for several important positions on the user's body, including hands, elbows, shoulders and head. We use the user's hands, elbows, and head points for the recognition of pointing gestures. We are primarily interested in pointing gestures generated by pointing with one's arm. We consider two rays for determining the pointing direction: elbow-hand and head-hand. Both of these methods were evaluated with the two skeleton tracking implementations. For each depth frame, this yields two rays for each of the OpenNI and MS-SDK trackers:

- $\vec{v}_{eh} := p_{elbow}\vec{p}_{hand}$
- $\vec{v}_{hh} := p_{head}\vec{p}_{hand}$

When a pointing gesture recognition request is received from a higher level process, the gesture is searched in a time window of T seconds. Two conditions must be met to trigger a pointing gesture:

- $\vec{v}_{eh}$ makes an angle more than ϕ_g with the vertical axis
- p_{elbow} and p_{hand} stays near-constant for duration Δt_g

The first condition requires the arm of the person to be extended away from his/her body, while the second ensures that the gesture is consistent for some time period. The parameters are empirically determined as: $T = 30s$, $\phi_g = 45°$ and $\Delta t_g = 0.5s$.

4.4.2.2 Representing Pointing Directions

We represent a pointing ray in two angles: a "horizontal" / "azimuth" sense we denote as θ and a "vertical" / "altitude" sense we denote as ψ. We first attach a coordinate frame to the hand point, with its z-axis oriented in either Elbow-hand $\vec{v}_{eh}$ or Head-Hand $\vec{v}_{hh}$ directions, depending on which method is used. The hand point was chosen as the origin for this coordinate system because both of head-hand and elbow-hand pointing methods include the user's hand. The transformation between the sensor frame and the hand frame ${}^{sensor}T_{hand}$ is calculated by using an angle-axis rotation. An illustration of the hand coordinate frame for Elbow-Hand method and corresponding angles are shown graphically in Figure 18.

Given this coordinate frame and a potential target point P, we first transform it to the hand frame by:

$$ {}^{hand}p_{target} = T_{hand} \cdot p_{target} $$

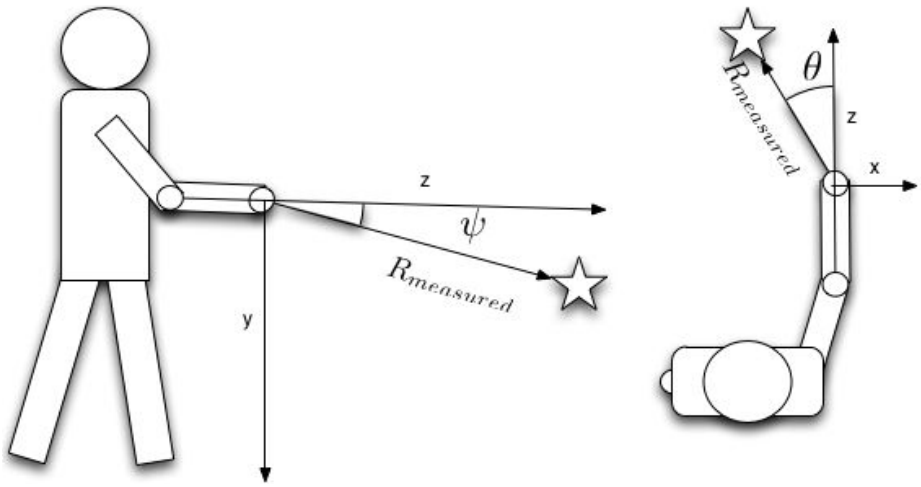


Figure 18: Vertical (ψ) and horizontal (θ) angles in spherical coordinates are illustrated. A potential intended target is shown as a star. The z-axis of the hand coordinate frame is defined by either the Elbow-Hand (this example) or Head-Hand ray.

We calculate the horizontal and vertical angles for a target point as ${}^{hand}p_{target} = (x_{targ}, y_{targ}, z_{targ})$ follows:

$$[\theta_{target}\ \psi_{target}] = [atan2(x_{targ}, z_{targ})\ atan2(y_{targ}, z_{targ})]$$

Where $atan2(y, x)$ is a function returns the value of the angle $\arctan(\frac{y}{x})$ with the correct sign. This representation allows us to calculate the angular errors in our error analysis experiments in Section 4.4.4.1. The angles for each object is then used to find the intended target, as explained in the following section.

4.4.2.3 Determining Intended Target

We propose a probabilistic approach to determine the referred target by using statistical data from previous pointing gesture observations. We observed that head-hand and elbow-hand methods, implemented using two skeleton trackers, returned different angular errors in spherical coordinates. Our approach relies on learning statistics of each of these approaches, and compensating for the error when the target object is searched. First, given a set of prior pointing observations, we calculate the mean and variance of the vertical and horizontal angle errors for each pointing method.

This analysis will be presented in Section 4.4.4.1. Given an input gesture, we apply correction to the pointing direction and find the Mahalanobis distance to each object in the scene.

When a pointing gesture is recognized, and the angle pair

$$[\theta_{target}\ \psi_{target}]$$

is found as described in the previous section, we first apply a correction by subtracting the mean terms from measured angles:

$$[\theta_{cor}\ \ \psi_{cor}] = [\theta_{target} - \mu_\theta \quad \psi_{target} - \mu_\psi]$$

We also compute a covariance matrix for angle errors in this spherical coordinate system:

$$S_{type} = \begin{bmatrix} \sigma_\theta & 0 \\ 0 & \sigma_\psi \end{bmatrix}$$

We get the values for $\mu_\theta, \mu_\psi, \sigma_\theta, \sigma_\psi$ from Tables 5 and 6 for the corresponding gesture type and target. We then compute the mahalanobis distance to the target by:

$$D_{mah}(target, method) = \sqrt{[\theta_{cor}\ \psi_{cor}]^T S^{-1}_{method} [\theta_{cor}\ \psi_{cor}]}$$

We use D_{mah} to estimate which target or object is intended. We consider two use cases: the objects are represented as a 3D point or a point cloud. For point targets, we first filter out targets that have a Mahalanobis distance larger than a threshold $D_{mah} > D_{thresh}$. If none of the targets has a D_{mah} lower than the threshold, then we conclude that the user did not point to any targets. If there are multiple targets that has $D_{mah} \leq D_{thresh}$, then we determine ambiguity by employing a ratio test. The ratio of the least D_{mah} and the second-least D_{mah} among all targets is compared with a threshold to determine if there is ambiguity. If the ratio is higher than a threshold, then the robot can ask the person to resolve the ambiguity.

If the target objects are represented as a point cloud, we then compute the horizontal and vertical angles for every point in the point cloud and find the minimum mahalanobis distance among all. The distance to an object is then represented by this minimum value. Usage of the point cloud instead of the centroid for determining the intended object has several advantages. First, it yields better estimations due to the coverage of the full point cloud. Second, it takes into account the size of the object. For example, if a person is pointing to a chair or door, it is very unlikely that he/she will target the exact center. If the point cloud is used, then we can easily tell that the object is targeted.

4.4.3 Data Collection

To evaluate the accuracy of pointing gestures, we created a test environment with 7 targets placed on planar surfaces in view of a Kinect sensor (Figure 19). Depth data was collected from six users, who pointed at each of the seven targets with their right arm while standing at 2 meters away from the sensor. Targets 1 through 4 were on tables positioned around the user, while targets 5 through 7 were located on a wall to the user's right. Our use case is on a mobile robot platform capable of positioning itself relative to the user. For this reason, we can assume that the user is always centered in the image, as the robot can easily rotate to face the user and can position itself at a desired distance from the user.

4.4.3.1 Ground Truth Target Positions

We make use of a plane extraction technique from a point cloud to have an accurate ground truth measurement. First, the two table and wall planes are extracted from the point cloud data using the planar segmentation technique described in [131]. We then find the pixel coordinates of corners on targets in RGB images, using Harris corner detection [41], which produces calculated corners in image coordinates with sub-pixel accuracy. The pixel coordinate corresponding to each target defines a ray in

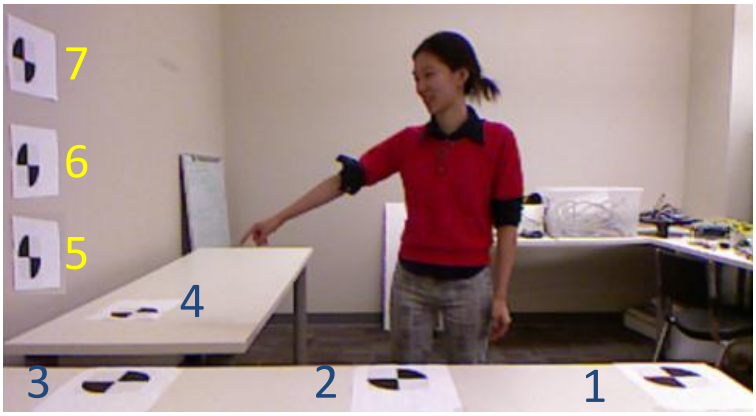


Figure 19: Our study involved 6 users that pointed to 7 targets while being recorded using 30 frames per target.

3D space relative to the Kinect's RGB camera frame. These rays are then intersected with the planes detected from the depth data, yielding the 3D coordinates of the targets.

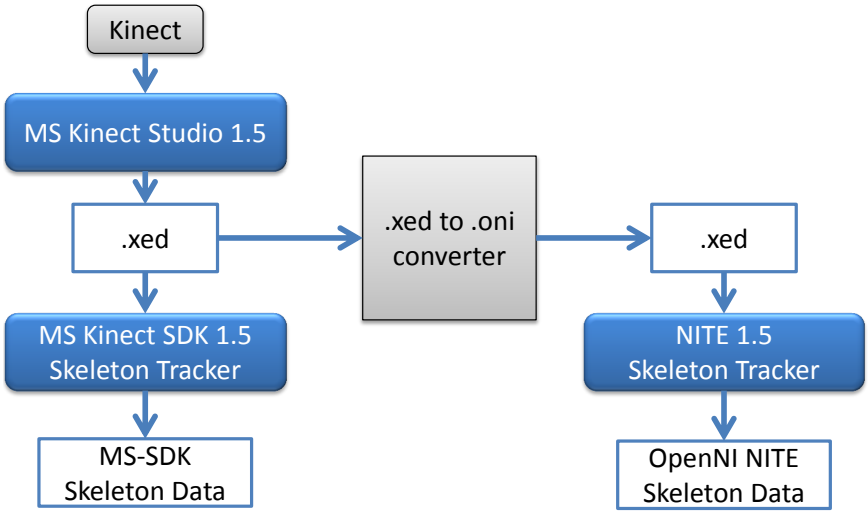


Figure 20: Data capturing pipeline for error analysis of pointing gestures.

In order to be able to do a fair comparison between MS-SDK and OpenNI skeleton trackers, we used the same dataset for both. MS-SDK and OpenNI use different device drivers, therefore can not be directly used on the live depth stream at the same time. Because of this, we extract the skeleton data offline in multiple stages. The pipeline for the data capture procedure is illustrated in Figure 20. We first save the depth

streams as .xed files using Microsoft Kinect Studio program. The acquired .xed file is converted to .oni in a OpenNI recorder application by streaming the depth stream to through Kinect Studio. The .oni is then played back in skeleton tracking application in OpenNI, which writes the OpenNI skeleton data to a .txt file. MS-SDK skeleton data is obtained by playing back the original .xed in the skeleton tracking application.

To factor out the effectiveness of our pointing gesture recognition method described in Section 4.4.2.1, we manually started the time when each pointing gesture began for data collection. Starting from the onset of the pointing gesture as annotated by the experimenter, the following 30 sensor frames were used as the pointing gesture. This corresponds to a recording of 1 second in the Kinect sensor stream. For each frame, we extracted skeleton data using both the MS-SDK and the OpenNI.

4.4.4 Evaluation

4.4.4.1 Error Analysis

The four rays corresponding to the four different pointing approaches described in Section 4.4.2.1 were used for our error analysis. As described in Section 4.4.3.1, the ground truth target positions are available. We computed two types of errors for each gesture and target:

- Euclidean error of ray intersections with target planes (Figure 21)
- Angular error in spherical coordinates (Tables 5, 6 and Figure 22)

We elaborate our error analysis subsequent sections.

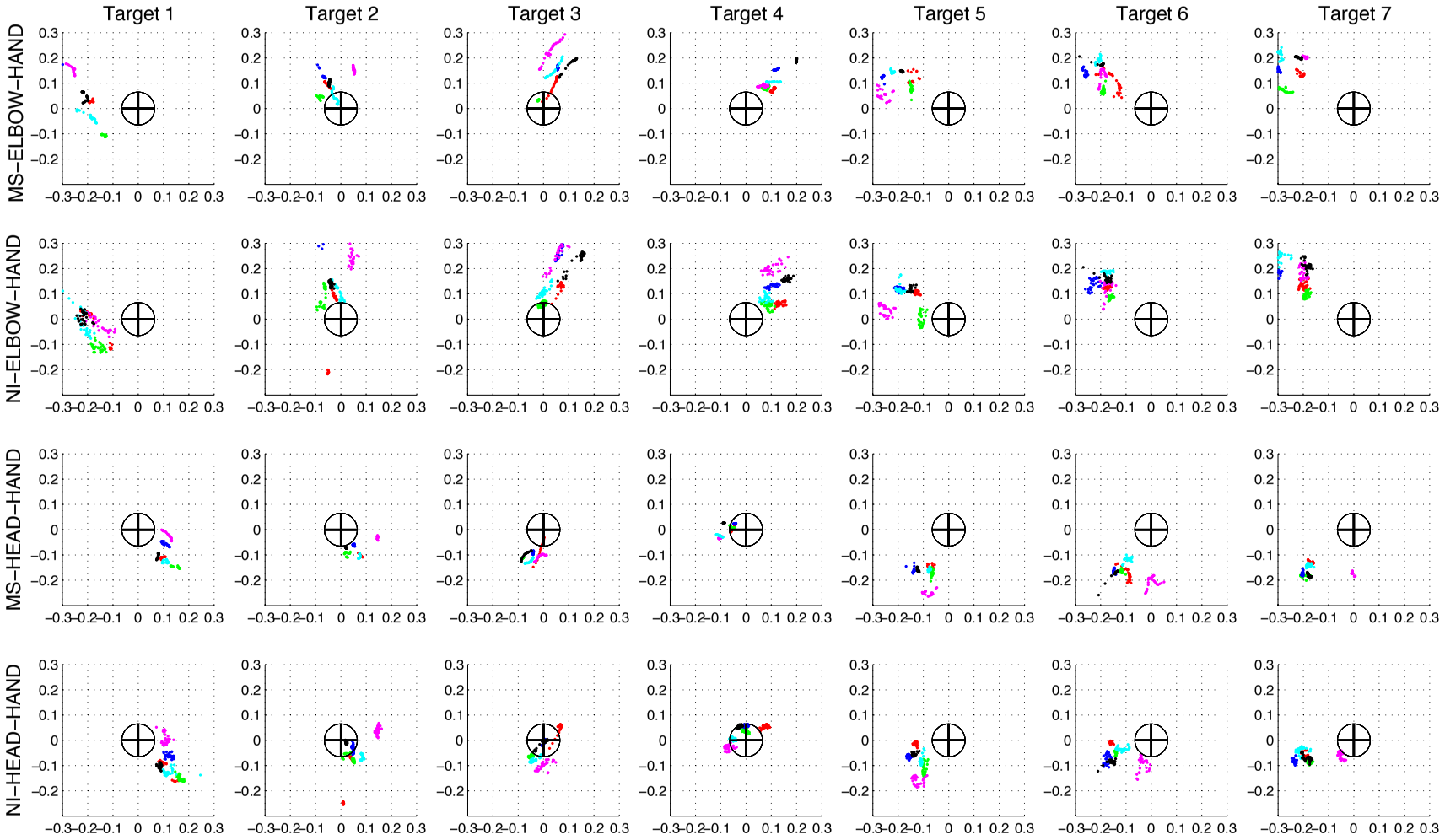


Figure 21: (Best viewed in color). Euclidean distance error in cartesian coordinates for each gesture method and target. The pointing ray intersection points with the target plane are shown here for each target (T1-T7) as the columns. Each subject's points are shown in separate colors. There are 30 points from each subject, corresponding to the 30 frames recorded for the pointing gesture at each target. Axes are shown in centimeters. The circle drawn in the center of each plot has the same diameter (13 cm) as the physical target objects used.

Table 5: μ and σ angular errors in degrees for each of Targets 1-4 (Figure 19), for each pointing method. The aggregate μ and σ is also shown.

	Target 1				Target 2				Target 3				Target 4			
	θ		ψ		θ		ψ		θ		ψ		θ		ψ	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
MS-Elbow-Hand	-15.7	2.9	5.5	6.1	-4.3	6.6	7.8	3.1	-3.7	2.4	7.4	3.1	-3.6	1.9	11.5	2.9
NI-Elbow-Hand	-16.4	2.9	4.3	7.0	-3.8	6.6	11.3	10.9	-4.8	2.6	9.7	3.4	-4.0	4.7	12.4	2.5
MS-Head-Hand	7.7	2.6	-12.0	5.3	10.8	6.4	-9.1	3.2	2.2	2.0	-8.3	3.2	-4.0	1.7	-4.3	3.2
NI-Head-Hand	8.5	2.6	-11.7	6.1	10.2	6.7	-5.7	8.0	2.0	2.3	-2.9	4.7	-3.2	2.5	1.45	4.8

Table 6: μ and σ of angular error in degrees for each of Targets 5-7 (Figure 19), for each pointing method. The aggregate μ and σ is also shown.

	Target 5				Target 6				Target 7				ALL TARGETS			
	θ		ψ		θ		ψ		θ		ψ		θ		ψ	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
MS-Elbow-Hand	-14.5	4.1	11.6	3.7	-16.6	3.3	9.9	3.7	-20.8	3.7	5.7	3.4	-11.3	7.7	8.5	4.5
NI-Elbow-Hand	-12.9	4.2	9.7	5.1	-16.2	2.6	11.7	3.7	-20.2	4.5	8.1	-3.0	-11.2	7.6	9.6	6.3
MS-Head-Hand	-9.4	1.9	-11.6	3.0	-7.1	4.7	-13.8	1.8	-8.0	5.5	-15.6	-2.9	-1.1	8.5	-10.7	4.8
NI-Head-Hand	-11.5	1.5	-4.7	4.8	-11.2	4.8	-4.9	2.9	-12.3	5.2	-8.9	-2.5	-2.4	9.6	-5.3	6.4

4.4.4.2 Euclidean error

Given a ray $\vec{v}$ in the sensors frame from one of the pointing gesture approaches, and a ground truth target point p_{target} lying on a target planar surface, the ray-plane intersection between $\vec{v}$ and plane was computed for each ray, resulting in a 3D point lying on the plane. Figure 21 shows the 2D projections for all 30 measurements from each subject (shown in different colors) and each target. For ease of display, the 3D intersection coordinates with the target planes are displayed in a 2D coordinate system attached to the target plane, with the ground truth target point as the origin.

As can be seen in Figure 21, the pointing gesture intersection points were fairly consistent across all users, but varied per target location. The elbow-hand method produced similar results for MS-SDK and OpenNI. The same is true for the head-hand method. It is also noteworthy that the data for each target tends to be quite clustered for all methods, and typically not centered on the target location.

4.4.4.3 Angular Error

We computed the mean and standard deviations of the angular errors in the spherical coordinate system for each pointing gesture method and target. Section 4.4.2.2 describes in detail how the angles (θ, ψ) are found. The mean and standard deviation values are given in Tables 5 and 6. The aggregate errors are also displayed as a box plot in Figure 22.

Several observations can be made from these results. The data from the elbow-hand pointing method reports that users typically point about 11 degrees to the left of the intended target direction, and about 9 above the target direction. Similarly, the data from the head-hand pointing method reports that users typically point about 2 degrees to the left of the intended pointing direction, but with a higher standard deviation than the elbow-hand method. On average, the vertical angle ψ was about

5 degrees below the intended direction for the OpenNI tracker and 10 degrees below for the MS-SDK tracker, with a higher standard deviation than the elbow-hand methods. The disparity between the two skeleton trackers for this pointing method is because they report different points for the head position, with the MS-SDK head position typically being reported higher than the OpenNI head position. The overall performance of the OpenNI and MS-SDK skeleton trackers, however, is fairly similar, with the MS-SDK having slightly less variation for our dataset.

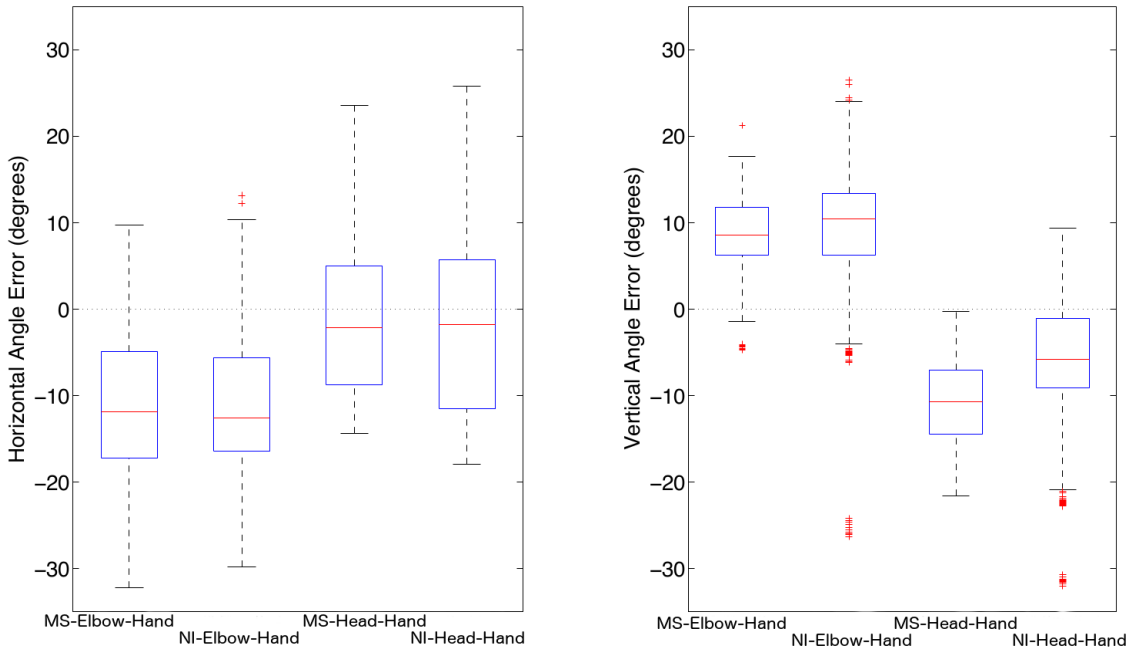


Figure 22: Box plots of the errors in spherical coordinates θ and ψ for each pointing method.

As can be seen in the aggregate box plot in Figure 22, the horizontal angle θ has a higher variation than the vertical angle ψ. Examining the errors for individual target locations shows that this error changes significantly with the target location. As future work, it would be interesting to collect data for a higher density of target locations to attempt to parameterize any angular correction that might be applied.

4.4.4.4 Object Separation Analysis

Using the error analysis and pointing gesture model presented in previous sections, we conducted an experiment to determine how our approach distinguished two potentially ambiguous pointing target objects. We use the results of the angular error analysis results and not the euclidean error analysis for the remainder of the paper because of our angular representation of pointing gestures.

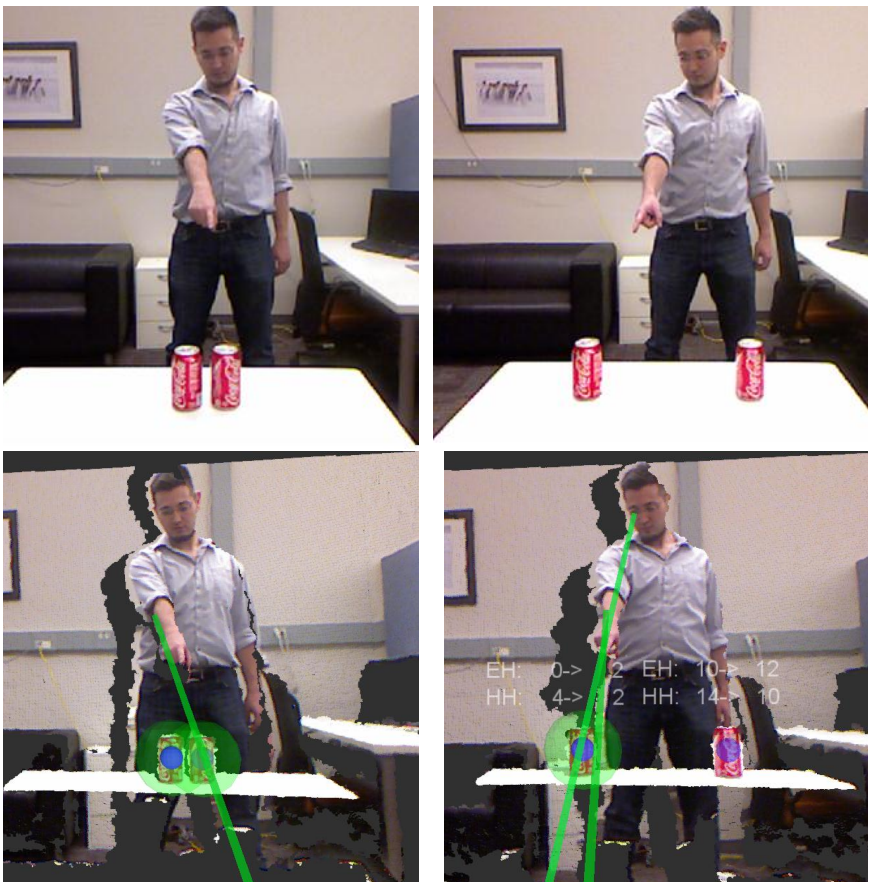
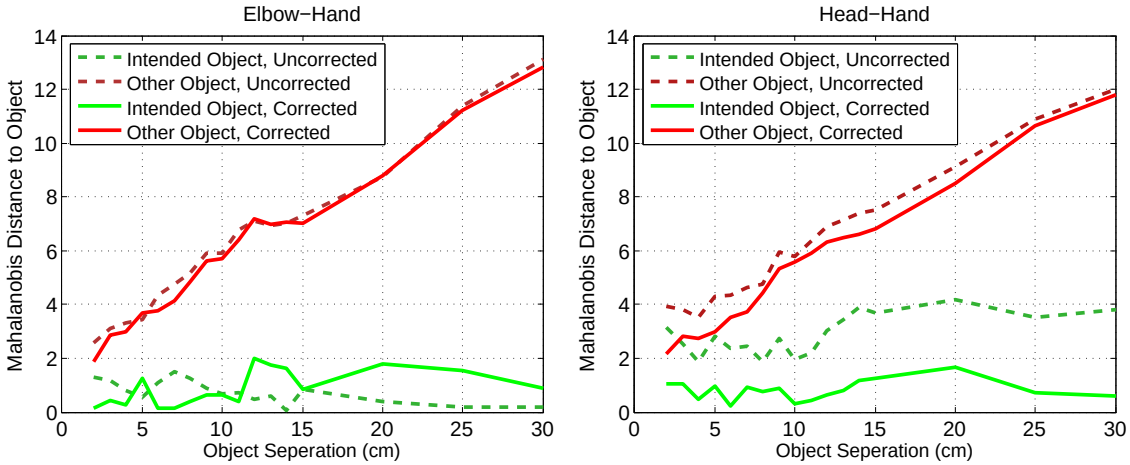


Figure 23: Example scenarios from the object separation test is shown. Our experiments covered separations between 2cm (left images) and 30cm (right images). The object is comfortably distinguished for the 30cm case, whereas the intended target is ambiguous when the targets are 2cm apart. Second row shows the point cloud from Kinect's view. Green lines show the Elbow-Hand and Head-Hand directions and green circles show the objects that are within the threshold $D_{mah} < 2$.

The setup consisted of a table between the robot and the person and two coke cans on the table (Figure 23) where the separation between objects was varied. To detect

the table plane and segment out the objects on top of it, we used the segmentation approach presented in [131]. The object centroid positions, along with their point clouds were calculated in real-time using our segmentation algorithm. The separation between objects were varied with 1 cm increments from $2 - 15cm$ and with $5cm$ increments between $15 - 30cm$. We could not conduct the experiment below 2 cm separation because of the limitations of our perception system. We used the OpenNI skeleton tracker because rest of our system is based in Linux, and we already found that performance difference with MS-SDK for pointing angle errors is insignificant. The experiment was conducted with one user, who was not in the training dataset. For each separation, the user performed 5 pointing gestures to the object on the right and 5 to the object on the left. The person pointed to one of the objects and the Mahalanobis distance D_{mah} to the intended object and the other object is calculated using the approach in Section 4.4.2.3. We used the mean and standard deviation values of Target 2 (Figure 19) for this experiment because the objects were located between the robot and the person.



(a) (b)

Figure 24: Resulting Mahalanabis distances of pointing targets from the Object Separation Test is shown for a) Elbow-Hand and b) Head-Hand pointing methods. Distance to the intended object is shown in green and the distance to the other object is shown in red. Solid lines show distances after correction is applied.

4.4.4.5 Results and Discussion

The results of the object separation experiment is given for Elbow-Hand (Figure 24(a)) and Head-Hand (Figure 24(b)) methods. The graphs plot object separation versus the Mahalanobis distance for the pointed object and the other object for corrected and uncorrected pointing direction. There are several observations we make by looking at these results.

First, the Mahalanobis distance D_{mah} for the intended object was always lower than the other object. The corrected D_{mah} for both Elbow-Hand and Head-Hand methods for the intended object was always below 2, therefore selecting the threshold $D_{thres} = 2$ is a reasonable choice. We notice that some distances for the unintended object at $2cm$ separation is also below $D_{mah} < 2$. Therefore, when the objects are $2cm$ apart, then the pointing target becomes ambiguous for this setup. For separations of $3cm$ or more, D_{mah} of the unintended object is always over the threshold, therefore there is no ambiguity. Second, correction significantly improved Head-Hand method accuracy at all separations, slightly improved Elbow-Hand method accuracy between $2 - 12cm$ but slightly worsened after $12cm$. We attribute this to the fact that the angles we receive is heavily user-dependent and can have a significant variance across methods as showed in Figure 22. Third, the Mahalanobis distance stayed generally constant for the intended object, which was expected. It linearly increased with separation distance for the other object. Fourth, patterns for both methods are fairly similar to each other, other than Head-Hand uncorrected distances being higher than Elbow-Hand. This reinforced our conclusion that both skeleton trackers had similar performance for the estimation of pointing targets. Our evaluation showed that in a scenario where the separation between two objects were varied, our system was able to identify that there is ambiguity for 2 cm separation and comfortably distinguished the intended object for 3 cm or more separation.

CHAPTER V

PEOPLE-AWARE NAVIGATION

Autonomous navigation is one of the most fundamental capabilities for a mobile robot. There are many algorithms that achieve point-to-point autonomous navigation thanks to the advances in mapping, localization and motion planning research. Many of these algorithms are optimized to find the least-cost path, or the shortest path. However, when there are humans in the environment, such algorithms suddenly become inefficient or insufficient. For example, while it is acceptable for a robot to get very close to a wall, doing so to a human is socially unacceptable and unsafe. Similarly, sudden appearance of a robot can surprise humans and cause discomfort. There are many other social scenarios where the shortest path may not be optimal. In addition to sub-optimality, these approaches may be incomplete in the sense that they can not find a solution even though there is a feasible one. This is because shortest-path navigation algorithms treat every object in the environment as an obstacle. This assumption does not hold when intelligent agents are present in the environment. Therefore navigation should differentiate humans and obstacles to enable intelligent robot behaviors.

Another aspect to spatial interaction between humans and robots is the dynamics of the robot motion. For example, people may feel uncomfortable and unsafe when they are in close proximity to high-speed agents or objects. Therefore, for a robot in a human environment, while it may be acceptable to speed up in some regions, its speed should be limited in places where there is a significant possibility of encountering a human.

In this chapter, we first provide a background on the most commonly used autonomous navigation methods in Section 5.1 and provide a literature survey on navigation among people in Section 5.2. In Section 5.3, we review the mapping and localization techniques we use. In Section 5.4, we present how the goal points for navigation are determined. We then present our people-aware navigation method in Section 5.5. Lastly, we present speed maps, that sets speed limits for mobile robots in an environment in Section 5.6.

5.1 Standard Approach in Autonomous Navigation

In general, there are two prerequisites that enables autonomous point-to-point navigation:

1. The map of the environment, usually in the form of a discrete grid, that represents static objects in the environment

2. A method to localize the robot in the map using sensory information as it moves in the environment

Robot navigation involves finding a collision-free path from a start pose (x_0, y_0, θ_0) to a goal pose (x_g, y_g, θ_g), where the coordinates are defined with respect to a world frame, i.e. map. In real-time operation, (x_0, y_0, θ_0) is the robot's current pose as the robot tries to reach to the goal pose from where it currently is. θ_g is optional as the goal of the robot could be to reach the goal position regardless of its orientation. The goal position is provided from an external process, and we will touch upon how the goal positions are calculated in Section 5.4.

A common approach to path planning is to divide the path planning into two parts: *global* and *local*. Global planning aims to find a path from the start position to the goal position. The global path is a set of consecutive positions that connect the start postion to the goal position on the gridmap. A global path is usually found

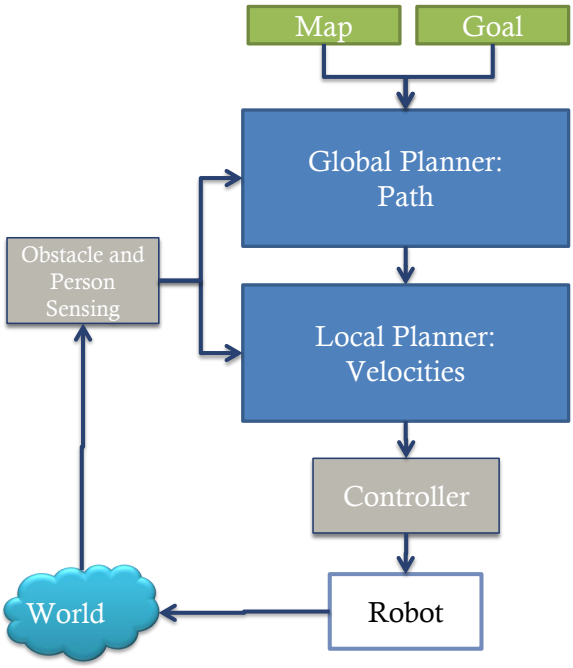


Figure 25: High-level system overview of mobile robot navigation

with a search algorithm executed on a graph or tree of nodes, where a node represents a robot state. The search heuristics is dependent on specific global planners. Usually, short and collision-free shorter paths are favored. The local planner is responsible for the execution of the global path by calculating a trajectory and sending velocity commands to motor controllers. As the robot acts in the environment, its sensors sense the new state of the robot and people, and the new iteration begins. The system overview is shown in Figure 25.

A popular method to to implement the global and local planners is by using a *costmap*. A *costmap* has the same representation as the map. Different from static maps, costmaps cells can have non-zero values, which represent a cost. Many different formations of cost structures can be designed and costmap designs alter the behavior of the robot.

Note that this approach assumes the robot is able to execute any path provided to it. Holonomic robots can move in any direction, however non-holonomic robots has constraints in their movements. For example, car-like robots can not move sideways.

Two common approaches to solve this problem are: to implement trajectory planners that can handle imperfect control or to embed the robot's dynamics into planners.

5.2 Related Work

In this section, we review the literature on robot navigation in human environments including socially acceptable navigation, learning behaviors from humans and cooperative navigation.

Socially Acceptable Path Planning

Socially acceptable robot navigation include point-to-point navigation [119], approaching people [108] and evacuating buildings [90]. Some works used the personal space concept in cost-based general path planners [119, 58]. Sisbot [119] models the personal spaces as a ellipse-shaped Gaussian cost functions, and takes into account the safety and vision fields of humans. Kirby [58] presents a path planner that takes into account social conventions such as tending to one side of the hallways. A potential field based trajectory planner for dynamic human environments is presented by Svenstrup [123]. Rios-Martinez [102] presents a RRT-based planner that considers not just safety but also the comfort of humans. In simulation, if interaction within a group of people is detected, the robot can either join the group or try not to disturb the interaction. This approach is implemented on a wheelchair robot [136]. Althaus [1] presents a robot that can join a group of people and adjust to the formation reactively. The scenario where a robot encounters a human in a hallway is studied by Pacchierotti [94]. They experimentally found parameters such as the distance between the human and the robot when the robot begins to deviate from its path and lateral distance that robot should be placed when it is passing the human. Lu [73] show that using gaze cues makes robot-human hallway passing more efficient.

Learning Navigation from Human Behavior

Robots that move like humans are usually favored, as observed by Sasaki [107].

One way to simulate human navigation behavior is to use costmaps that capture social conventions [109, 74]. Contrary to the imitation approach in Bennewitz's approach [8], the robot tries to avoid predicted paths, with the goal to minimize the risk of interference. Kuderer [66] presents a tele-operated robot that computes the policy of a navigation behavior by learning from observations of pedestrians. Pellegrini [97] trains a dynamic social behavior, that account for social interactions, also using pedestrian data.

Human Cooperation in Robot Navigation

Work by Dragan [23] claim that the robot motions should be predictable so the human observers can judge the motive of the robot and predict its future behaviors. An observational study by Lichtenthaler [71] claims that three features can increase the predictability of robot navigation: straight lines, stereotypical motions and usage of additional gestures. Kruse [64] observes that when paths of two humans are crossed at a right angle, they adapt their velocity rather than the path. This behavior is implemented on a robot, resulting in more predictable motions. Robots can exploit human cooperation in certain scenarios. In populated environments, one way to move with the crowd is to follow individuals that move towards the robot's goal [121, 85]. Trautman [130] mentions the 'freezing problem', where traditional path planners fail to produce a feasible solution in crowded human environments. Muller [85] briefly mentions a 'shooing away' behavior, where the robot accelerates towards a human, hoping that he/she will get out of the way. Kruse [65] introduces an optimistic planner, which assumes that people will cooperate with robot movements. Their approach relies on assigning a non-infinite cost if the robot enters to a human's personal space, however the plan fails if humans doesn't move as expected because of the lack a local planner.

5.3 Mapping and Localization

Our navigation approach uses a 2-layer map:

1. Metric layer: Used for localization and obstacle avoidance

2. Semantic layer: Used for tasking and includes features such as planar landmarks, doors, navigation waypoints and objects

In a new environment, we first run the ROS *gmapping* Simultaneous Localization and Mapping (SLAM) package to generate the metric map. This SLAM approach is based on the Rao-Blackwellized particle filter approach [38]. The front Hokuyo laser scanner is used to generate the map and localize the robot. The semantic layer is generated and edited by the interactive Tour Scenario (Chapter 4).

After the map is generated and saved, the robot continiously localizes itself in the map. We used the ROS *amcl* package for localization, which implements the KLD-samping Monte Carlo particle localization approach [28]. This approach provides local localization: it only can estimate the position of the incrementally from recent location information. Therefore, when the robot is started, this method of localization needs a global position estimate. Our system accepts two methods of initialization:

1. Manual Initialization: This is the standard method of initialization in ROS. A robotics expert provides a crude initial position estimate using the rviz interface that runs on the robot's computer. This method, in its current form, requires a robotics expert to specify the initial pose of the robot.

2. Automatic Initialization using QR codes: We use a specially designed and located QR code to globally localize the robot. The robot automatically infers its initial location upon detection of the QR code, without external help from humans. System designers can place the QR code tags to strategic places in the environment, such as entrances and frequently visited areas.

QR codes in general contain links, text and various other data. In our application, the QR pattern includes text in XML format, which includes the link to the map and speed map files, and the position of the QR pattern in the map. We used the *visp_auto_tracker* ROS package to extract the pose of the QR code as well as the content embedded in the pattern. A QR code pattern that includes localization information is shown in Figure 26. The data embedded to the QR code in the figure is in the form:

Here is a sample information that a robot can read from a QR pattern:

```
<root>
  <map>http://link1/</map>
  <speedmap>http://link2/</speedmap>
  <x>101.26</x>
  <y>98.76</y>
  <z>1.45</z>
  <q1>0.0</q1>
  <q2>0.0</q2>
  <q3>0.0</q3>
  <q4>1.0</q4>
</root>
```

The position provided in the pattern is the position and orientation (in quaternions) of the QR tag in the map frame. Using this information, a robot can acquire the knowledge of the environment automatically and locate itself in the map. When the position of the QR pattern is acquired upon its detection, here is how the pose of the robot ${}^{map}T_{robot}$ is calculated:

$${}^{map}T_{robot} = ({}^{map}T_{QR}) \cdot ({}^{QR}T_{robot})$$

$${}^{QR}T_{robot} = ({}^{QR}T_{cam}) \cdot ({}^{cam}T_{robot}) = ({}^{cam}T_{QR})^{-1} \cdot ({}^{robot}T_{cam})^{-1}$$

Therefore:

Figure 26: A robot can acquire map information and localize itself against the map upon detection of a specially designed QR code

$${}^{map}T_{robot} = ({}^{map}T_{QR}) \cdot ({}^{cam}T_{QR})^{-1} \cdot ({}^{robot}T_{cam})^{-1}$$

where:

${}^{robot}T_{cam}$ is the transformation from the robot base frame to the camera frame on the robot and is known.

${}^{cam}T_{QR}$ is the pose of the QR code in the camera frame and is available upon detection of the QR code.

${}^{map}T_{QR}$ is the transformation from the robot base frame to the camera frame on the robot and is read from the data embedded to the QR code.

${}^{map}T_{robot}$ is the pose of the robot in the map and is unknown.

After the initial pose is provided, the local localization method *amcl* takes over.

5.4 Goal Points for Navigation

As presented in Section 4, our interactive system allows a user to annotate landmarks. After completing the *Tour Scenario*, the robot is able to use labeled entities as navigation goals. A user can enter a navigation destination to the robot in three ways: via labeled waypoints, planar landmarks or objects.

5.4.1 Labeled Waypoints

If a waypoint is labeled and saved, the robot attaches that label to the raw pose. Therefore, if the robot is instructed to navigate to a labeled waypoint, then the goal is readily the pose of that waypoint.

5.4.2 Labeled Planar Landmarks

If the label is attached to planar landmark, or a set of planar landmarks, we use the following methodology depending on the number of landmarks attached to the corresponding label:

5.4.2.1 Only a single plane has the corresponding label:

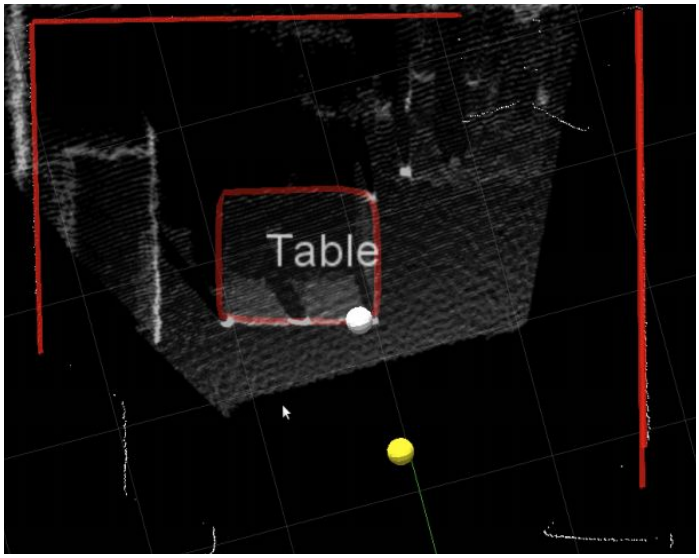


Figure 27: Top down point cloud view of a room. A planar landmark with label *Table* has previously been annotated by a user. The convex hull for the planar landmark is shown in red lines. When asked to navigate to *Table*, the robot calculates a goal position, which is shown as the yellow point.

We assume that the robot should navigate towards the closest edge of the plane, so we first select the closest vertex on the landmark's convex hull to the robot's current position, and project it to the floor plane. We find the line between the closest vertex on the convex hull and the robot's current pose, and the goal position would be on

this line, a meter away from the vertex. This results in the robot navigating to near the desired planar surface, and facing it. This method is suitable for both horizontal planes such as tables, or vertical planes such as doors. An example for calculating a goal pose for a uniquely labeled planar landmark is shown in Figure 27.

5.4.2.2 Multiple planes are attached to the same label

When there are multiple planes associated with the same label, then we interpret this landmark as a region or space, such as a room or corridor. In this case, we project the points of all planes with this label to the ground plane, and compute the convex hull. The goal position is chosen as the centroid of the convex hull. Rooms and spatial regions could be useful fort tasks such as object search. An example for calculating a goal for a two landmarks that has the same label is shown in Figure 28.

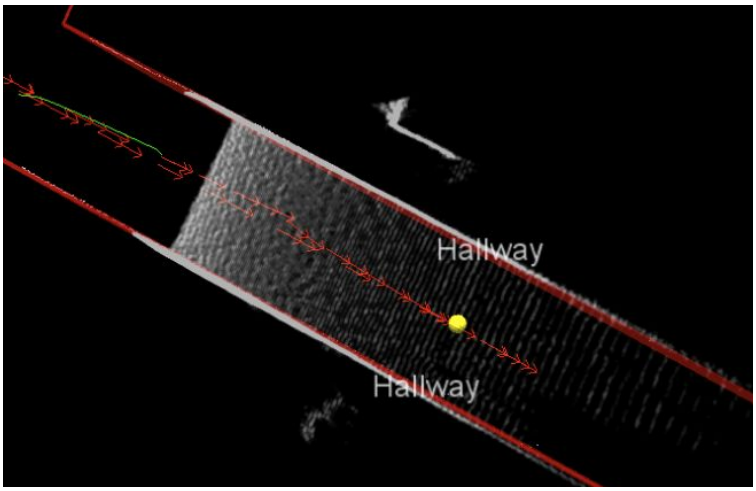


Figure 28: Top down point cloud view of a hallway. The user has previously annotated two planar landmarks with the same label, *Hallway*. When asked to navigate to *Hallway*, the robot chooses a goal position in the middle of the planar landmarks, shown as the yellow point.

5.4.3 Labeled Objects

As discussed in Section 4.2.3, plane detection is executed before tabletop object detection. When the robot is asked to navigate to a labeled object, the planar surface

that the object lies on is given as the goal landmark. The robot calculates the goal position as described in the single labeled landmark case in the previous section.

5.5 People Aware Navigation

People-aware navigation algorithms aim to generate human-friendly paths that consider the safety and comfort of people. As reviewed in Section 5.2, a common assumption for point-to-point people aware navigation is that humans are independent agents and that robot's motions have no effect on people's motions. However, humans navigate by constantly anticipating other people's reactions. Similarly, mere presence of a robot in motion is likely to influence how nearby humans would move. Robots can potentially use this implicit cooperation between moving embodied agents. For example, consider a robot that is outside a room and given a goal pose in the room. There is a person standing at the door and blocking the path. Such an example is illustrated in Figure 29. Standard path planners fail to produce a solution to this problem. The role of physical embodiment in human-robot interaction is significant [138], however it is commonly ignored in robot navigation. A people-aware planner should anticipate that the human may give way to the robot if it expresses its desire to enter the room, either implicitly or explicitly. By using anticipation, a robot can reduce time of travel and behave more human-like.

In this section, we propose a people-aware navigation planner that considers reactions of humans to robot motion. Our planner first finds the least-cost path with a cost definition considering the path length, and safety and disturbance of people. The costmap definition is discussed in Section 5.5.1. Then the path is refined by simulating people's reaction to robot's motion using Social Forces Model [43]. The path refinement step will be described in detail in Section 5.5.1.1. In dynamic simulation, robots and humans repulse each other, and additional forces helps the robot to stay away from obstacles and conserve formation in groups. Paths are re-planned

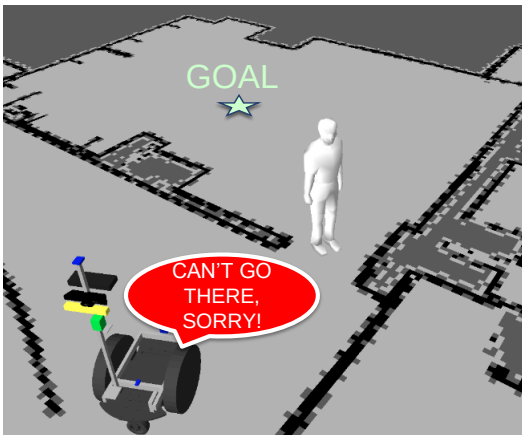


Figure 29: Standard path planners fail to produce a solution to the 'room problem'. Our people-aware planner anticipates that the human can give way to the robot if it approaches towards its goal.

whenever the world state changes or humans do not move as anticipated. In Section 5.5.2, we discuss the local planner. We then discuss the implementation of the system in Section 5.5.3, demonstrate two example scenarios in simulation in Section 5.5.3.1 and two on the real system in Section 5.5.3.2.

5.5.1 Global Planner

The global planner takes the start and goal positions and a 2D grid map as input and aims to find a set of waypoints that connects the start and goal cells. The output path has the minimum cost with regards to a cost function with 3 parameters: path length, human safety and group disturbance. We use A* search with Euclidean distance heuristic on a 8-connected grid map to find the minimum cost path. The configuration space obstacles are found by inflating the map obstacles for as much as the radius of the robot with the circular robot assumption.

Path length cost: Each action a of the robot (moving to one of the 8 adjacent cells) has a non-negative action cost $Cost_a(x_i, y_i, a)$. If the destination cell is occupied by a configuration space obstacle, then the action cost is infinite. Otherwise, it is the

euclidean distance in meters. The action cost is thus defined as:

$$Cost_a(x_i, y_i, a) = \begin{cases} u & \text{if } a = \text{N, E, S, W} \\ u\sqrt{2} & \text{if } a = \text{NW, NE, SW, SE} \\ \infty & \text{if } Cell(x_{i+1}, y_{i+1}) \text{ in obstacle} \end{cases} \tag{11}$$

where N,NW,.. are the grid cell expansion directions and u is the grid cell size. The resulting path length cost of a path P is then the sum of all action costs:

$$Cost_{path}(P) = \sum_{a \in P} Cost_a(x_i, y_i, a) \tag{12}$$

Safety cost: The notion of safety is the absolute need of any human-robot interaction scenario. This cost is a human centered 2D Gaussian form of cost distribution and aims to keep a distance between the robot and the humans in the environment. While some approaches use un-isotropic cost functions to account for human orientation, we use a isotropic Gaussian for its simplicity. Each cell coordinate around a human contains a cost inversely proportional to the distance. Since the safety loses its importance when the robot is sufficiently far away from the human, safety cost becomes zero after a threshold distance. If there are multiple people in an environment, the safety cost of a cell takes its value from the closest human.

$$Cost_{safety}(x, y) = \begin{cases} r \max_{h \in H}(\mathcal{N}(\mu_h, \sum)) & \text{if } d < dmax_{safety} \\ 0 & \text{if } d \geq dmax_{safety} \end{cases} \tag{13}$$

where d is the distance to the closest human, H is the set of all humans, $\mu_h = (|x - h.x|, |y - h.y|)$ and $\sum = 0.5I_2$ is a fixed covariance matrix. The multiplication by the grid cell size r compensates for the grid map resolution. Otherwise, for example, if a very fine map was used, safety cost would dominate the path length and disturbance costs, which are independent of the map resolution.

Disturbance cost: This cost is aimed to represent the cases where the robot potentially disturbs the interaction of a group of humans. For example, if two people are facing each other and talking, then the robot should not cross between them.

We model this with a disturbance cost that is introduced if the robot's path crosses between two people. We do not detect if there actually is conversation between the people, but estimate the disturbance cost using body poses of agents. This cost increases if body orientations of two people are facing each other and is inversely proportional on the distance between the two humans.

For each step taken in the grid, we check if the line segment from the current position to the projected position intersects a line segment between all pairs of humans. To illustrate, let's assume the robot crosses the line between human A and human B in Figure 30(a).

The disturbance cost is calculated as:

$$\begin{aligned} Cost_{dist}(x, y, a) &= \max(0, f(d).(\vec{AA'}.\vec{AB} + \vec{BB'}.\vec{BA})) \\ f(d) &= \frac{1}{d} - \frac{1}{dmax_{disturbance}} \end{aligned} \tag{14}$$

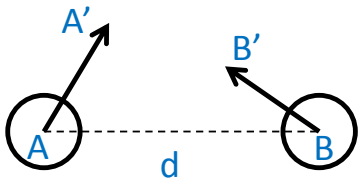
where all the vectors are normalized and $dmax_{disturbance}$ is the maximum distance between the humans that returns a disturbance cost. Figure 30(b) illustrates several examples of disturbance costs with $dmax_{disturbance} = 3$ meters.

Total Cost: The total cost of a path P is computed with a weighted average of path length, safety and disturbance costs. We use A* search to find the least-cost path.

$$Cost_{Total}(P) = w_p.Cost_{path} + w_s.Cost_{safety} + w_d.Cost_{dist} \tag{15}$$

5.5.1.1 Path Refinement using Social Forces

In this section, we describe the path refinement process that is applied to the global path. The initial geometric path generated by the global planner is not smooth, therefore robot motion might not be easy to interpret for human observers. The path refinement processes the global plan and simulates the parts of the path where group of humans are closeby. We use Social Forces Model (SFM) [43] to simulate the motions of humans and the robot. Interaction between people are modeled as



(a)

Humans Formation	Cost (d=1m)	Cost (d=2m)
	0	0
	0.66	0.16
	0.66	0.16
	0.94	0.23
	1.33	0.33

(b)

Figure 30: Disturbance costs in different human-human configurations and distances is shown. A path that crosses the dashed lines incurs the disturbance cost calculated on the right side of the table shown in b)

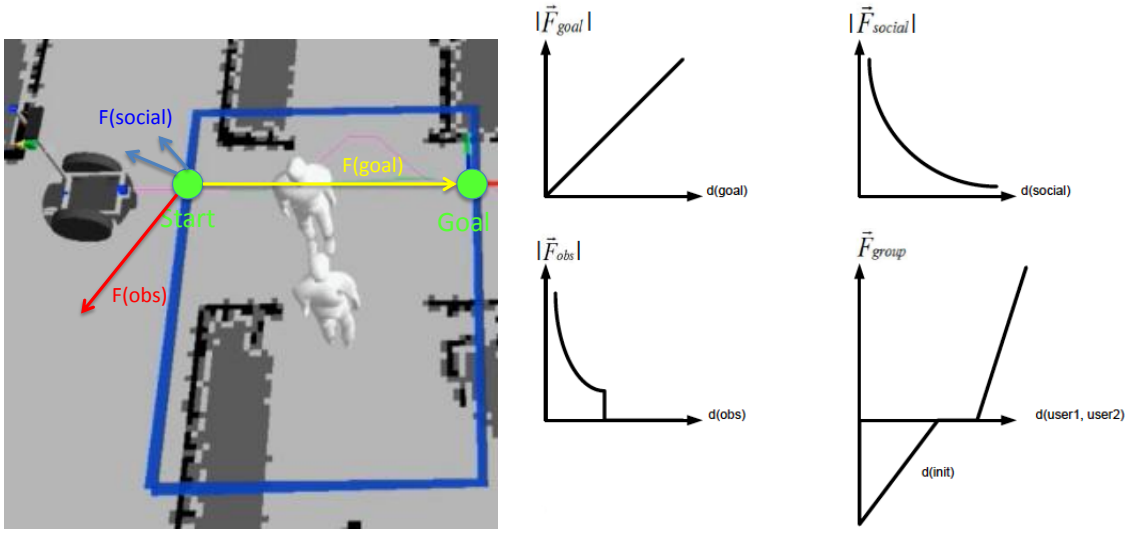
attractive and repulsive forces in SFM, similar to the Potential Field Method [55] for robot navigation. The forces are recomputed iteratively and the resulting simulated paths replaces the corresponding path sections in the global plan.

First, groups of people are found by clustering with respect to their positions. Simple euclidean distance thresholding is used for clustering people. In our current implementation, a group region is defined as a rectangle, although other shapes can also be used. The path refinement process receives the global plan and finds out where it enters and exits each group region. Goal of the dynamic iterative simulation is to find a sub-plan between two points: where the robot enters and departs the group region. Forces apply to all agents, including the robot and humans. We define 4 forces acting on the agents:

- F_{goal} : attraction towards a sub-goal
- F_{social} : repulsion from other agents

- F_{obs} : repulsion from the nearest obstacle
- F_{group} : attraction or repulsion towards group members

The directions of forces acting on the robot at the first iteration of force simulation are illustrated in Figure 31(a). The force magnitudes with respect to distances are plotted in Figure 31(b).



(a) (b)

Figure 31: a) Social forces acting on the robot, F_{goal},F_{social} and F_{obs}, are shown at the first iteration of the dynamic planner. Note that $F_{group} = 0$ as the robot does not belong to a group in this example. b) Social force magnitudes as a function of the distance between the two agents

Starting from the first group region that intersects the static plan, the following procedure is applied within every group region: At every iteration, first the force vectors acting on the robot are calculated. Then the planner takes a step in the direction of the net force vector for a fixed step size. Then each of the humans in the group takes a step towards the resultant force that is acting on them. The planner continues the iterations until a solution is found or there is a timeout. If a solution is found, the calculated sub-plan replaces the static plan in this group region. Potential fields are known to stuck to local minima [62], and the planner might go into infinite

loop. If a solution cannot be found, we stop the planner after a number of iterations and the static plan is taken as it is in the corresponding group region.

5.5.2 Local Planner

The local planner is responsible for finding the trajectory that the robot is capable of executing. It accepts a geometric global path as input and computes the linear and angular velocity necessary to follow the dynamic path. We adopt a local planner inspired by Dynamic Window Approach (DWA) by Fox [29]. In the original DWA approach, only circular trajectories are considered, defined by pairs of linear and angular velocities (v, w). They use an objective function, consisting of target heading, clearance from obstacles and velocity of the robot is maximized by sampling admissible velocities. Our approach also samples admissible velocities, but the optimization criteria we use consists only of the Euclidean distance to a sub-goal point chosen on the path that is ahead of the robot. The velocity that results in the closest proximity to the sub-goal is chosen and sent to robot controllers. At every control iteration, the sub-goal is chosen as the first point ahead of the robot that is further than a distance threshold. We found that a threshold of 0.25 meters was sufficient to choose the sub-goal. After the local planner calculates the output velocities, they are applied to the robot and the iterative process continues until the the robot reaches the goal. Since the goal is a singular point, it is impossible for the robot to be exactly at the goal. Therefore, a tolerance around the goal point, defined as a circle around the goal is defined.

Given the robot's current pose and an applied velocity, the DWA approach requires to have a motion model for the robot. The motion model projects what the robot pose would be, if a velocity pair is applied to it for a time period. The actuation model we used for our implementation is a non-holonomic mechanism with two wheels. While one can use the general motion equations derived in [29], we used linear approximated

motion equations for our robot, as shown in Equation 16. Given a robot pose $q^t = (x^t, y^t, \theta^t)$ at time t and an input velocity (v, w), the projected robot pose at time $t + \Delta t$ is:

$$q^{t+\Delta t} = f_{motion}(q^t, v, w, \Delta t) = \begin{Bmatrix} x^t - \frac{v}{w}sin(\theta^t) + \frac{v}{w}sin(\theta^t + w\Delta t) \\ y^t + \frac{v}{w}cos(\theta^t) - \frac{v}{w}cos(\theta^t + w\Delta t) \\ \theta^t + w\Delta t \end{Bmatrix} \quad (16)$$

5.5.3 Results

In this section, we provide qualitative results both in simulation and on the real robot.

5.5.3.1 Simulation

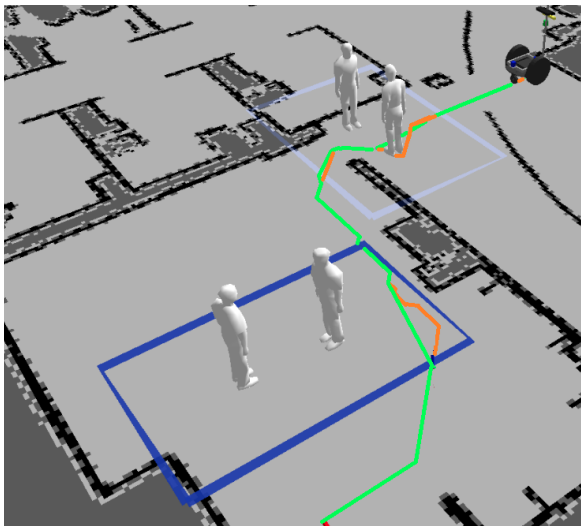


Figure 32: A solution is shown to the "Room Problem". The robot is outside a room and the goal pose is inside the room. Traditional planners can not solve the problem because two people are blocking the doorway. Our planner generates a tentative path, with the initial global plan shown in green and the dynamic refinements are shown in orange.

Room Problem: In this scenario, the robot is outside the room and a point inside the room is given as the goal (Figure 32). There are two people standing at the doorway and there are two more standing people inside. Traditional planners

can not find a solution to this scenario because the path is blocked. The static plan and dynamic plans are shown in green and orange, respectively. This path is planned for the current time but makes assumptions about future positions of humans. Note that the dynamic planner modifies only the parts of plan inside group regions (blue rectangles). In the first group region near the doorway, the static plan involves going between the humans. Dynamic simulation suggests that people will get closer to each other if the robot drives towards the side. In the second group region, since two humans are oriented to each other, going between them would add a high disturbance cost, therefore the static plan avoids going between them. Safety costs encourages staying far from the humans, but not too far because a longer path would increase the path length cost. The robot is further led to stay closer to the room boundaries in the dynamic planner due to the repulsive forces from both humans.

Office Environment: Goal of the robot is to navigate to a goal position in an office environment where there are 4 people (Figure 33). In this scenario, we show how the planned path is changes with the poses of humans even though the start and goal position of the robot doesn't change. There are 3 main ways the robot can navigate to its goal: left, center or right corridor.

In the first configuration in Figure 33, two people are grouped together as they are looking at each other. The robot decides to take the center corridor, first slightly disturbing the speaking duo, then switches sides in the corridor and reaches its goal. In the figure, the dynamic path (pink line) is overlaid on the static path (green line).

In the second configuration in Figure 33, The third person at the center corridor joins the conversation. Now we have 2 human groups, shown as rectangles, in the scene. Since passing through a group of 3 people would incur a high disturbance cost in addition to the safety cost, the robot decides to take a longer route (left corridor). Since this path does not intersect any group regions, no dynamic simulation is executed.

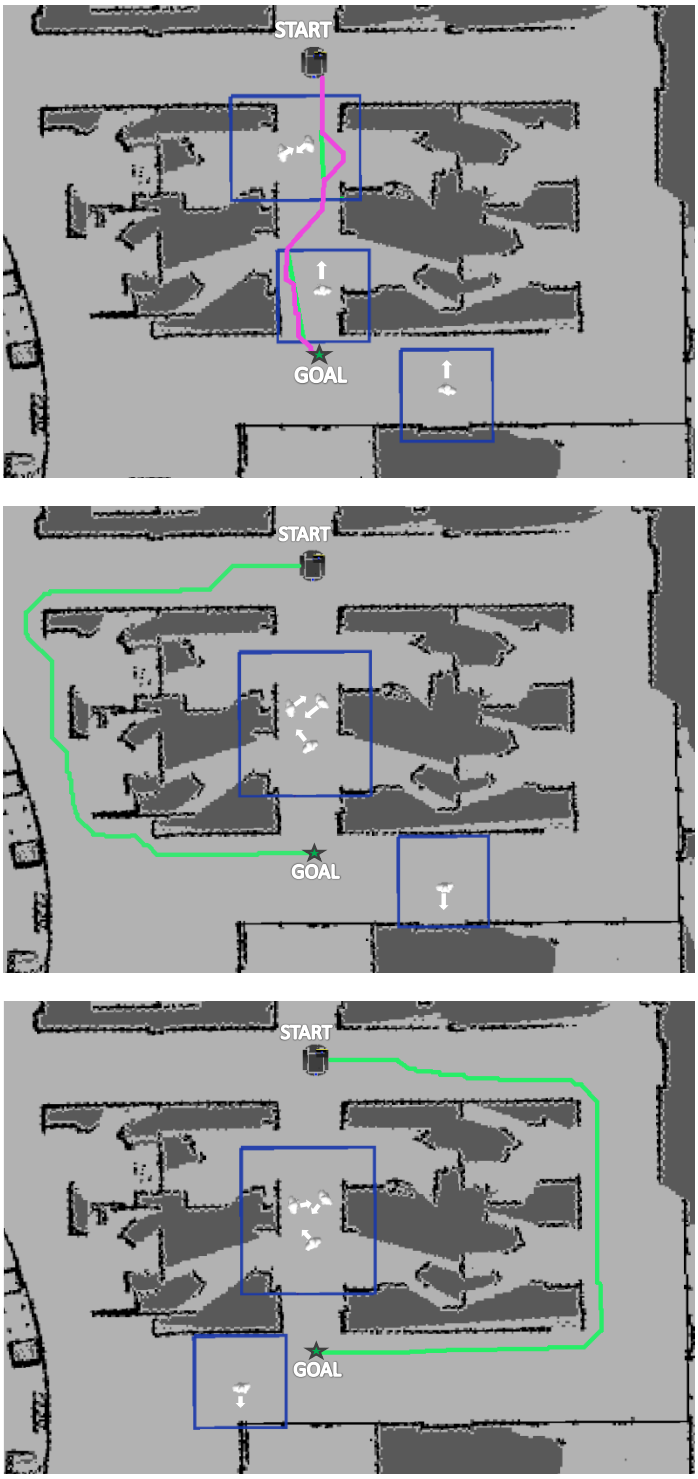


Figure 33: An example scenario for people aware navigation is shown. Each image depicts a different configuration of people in the environment. Our algorithm calculates a path for every situation. a) The robot takes shortest route, traveling in the vicinity of a group of two and another individual. b) third individual joins the group. Robot takes a longer path that doesn't have humans on path. c) fourth person changes his position, leading the robot to take the longest route.

In the third configuration in Figure 33, the group of three hasn't moved, but the fourth person person has changed its position. In this case, if the left corridor is taken again, an additional safety cost would be incurred. Therefore the robot decides to take the longest route (right corridor). Again, since the robot travels far from humans, no dynamic simulation is executed.

5.5.3.2 Real Robot

We used a the mobile robot platform Segway RMP-200 for experiments on the real system. We demonstrate our anticipatory navigation planner on the real system in two environments: hallway and kitchen. Each scenario is run twice under different human poses in order to demonstrate how the planner responds.

Hallway passing: In this scenario (Figure 34), robot's goal is to navigate to the end of the hallway. In the first run, humans move as the robot anticipates. In the second run, humans do not move as anticipated, and the robot adjusts its path. Each step is described in the caption of the figure. In both cases, the initial plan is to cut in between the two people. This is because the safety cost for getting close to one of the humans was more dominant than the disturbance cost.

Narrow corridor: In this scenario (Figure 35), robot's goal is to drive towards the exit door. There are 3 people nearby the robot. The robot can either take the shorter route that is the direct path, or take a longer path that is to the left of the table. Each important step is described in the caption of the figure. The first run shows that the robot may plan hoping to influence the human. The second run shows that the robot may take a longer route if the disturbance and safety costs are going to be large.

5.6 Speed Limits for Safe Navigation

In this chapter so far, we studied navigation planning that aims to generate human-friendly paths. The most important criteria for social navigation is the safety of

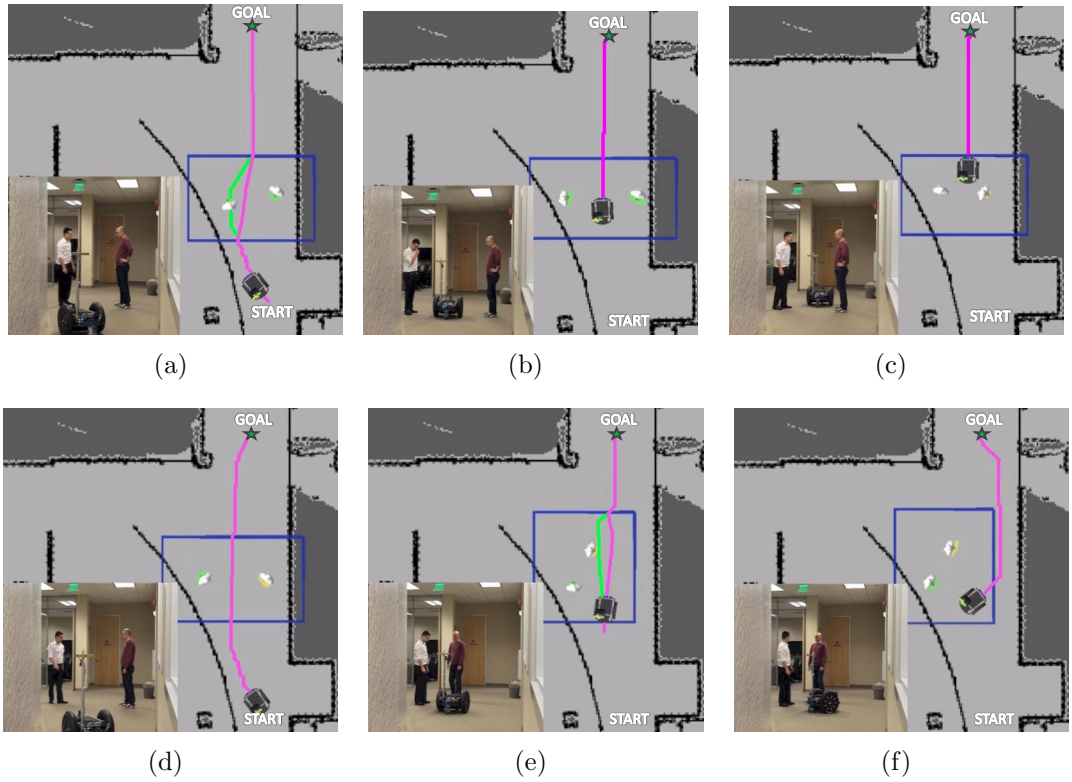


Figure 34: The Hallway scenario for people aware point-to-point navigation. Each row displays the steps of a different run. The static plan (green line) and dynamic plan refinement (pink line) are shown. First run: a) Navigation starts. The dynamic planner anticipates that people will give way to the robot when it starts to move towards them. b) Humans notice the robot, and give way by increasing the separation between them. c) The robot continues towards its goal and humans regroup. Second run: d) both the static and dynamic plan involves going in between humans again e) human on the right gets closer to the other person. Since a human made significant movement, dynamic planner re-plans. Plan no longer involves going in between. f) static planner periodic re-plan triggers, leading to robot to stick to the wall to the right.

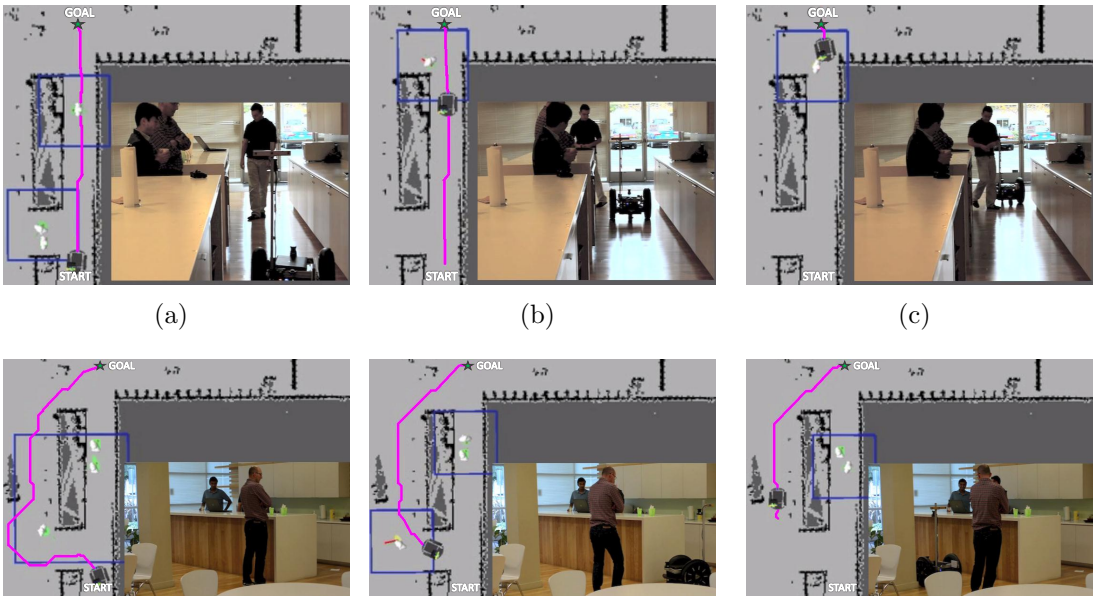


Figure 35: The Kitchen scenario for people aware point-to-point navigation. Each row displays the steps of a different run. In the first run, there are two people blocking the path to the left and one person at the narrow corridor. a) robot decides to take the shorter route, because it would disturb one person instead of two. There is not enough space to pass, and dynamic planner assumes the person would get out of the bottleneck to give way. b) human behaves as robot anticipated and gets out of the narrow passage. robot slows down because it enters the human region. c) person gets back to his original position, robot reaches the goal. In the second run: d) there are two people at the narrow corridor and one person on the left. The robot decides to take the longer route and pass the third person from left. The safety cost from the two others would be too high if the robot took the direct route. e) the person steps back as he recognizes the robot. since the person has moved, the dynamic planner re-plans and decides to pass from right. f) after the robot passes the person, it proceeds to its goal.

people. Previously, we introduced **Safety Cost** in Section 5.5.1 as well as repulsion forces in Section 5.5.1.1 to bolster safety. However, a seemingly safe path generated by our approach can still lead to an accident because it is may not be possible for the robot to track every human in the environment all the time. For example, when the robot is turning a corner, there is a risk that the robot suddenly encounters a person. Since the robot have finite deceleration, a collision may be unavoidable. Moreover, the person tracker may fail and may not be able to sense people. The top speed a robot would be allowed to navigate should dependent on the context. For instance, a robot should move slowly and carefully in a hospital room. On the other hand, the robot may navigate faster in a long office corridor. The speed limits should be provided by experts or the building owners, who would govern how fast the robots should navigate in a particular environment.

In this section, we introduce the concept of speed maps that sets the speed limits for mobile robots in an environment. We claim that usage of such speed maps make the robots safer and potentially more efficient. The speed map designed for the second floor of the College of Computing building at Georgia Tech is shown in Figure 36.

The free spaces in this map are divided into three zones:

1. Green zone: The robot is allowed to travel at relatively faster speeds.

2. Yellow zone: Human interaction is possible and the top speed is limited.

3. Red zone: Human encounter is probable and the top speed is minimal.

Speed maps can be designed in many ways depending on the context and task. This speed map shown in Figure 36 is designed by hand using the following rules: Spaces corresponding to rooms and cubicles are covered as Red Zones. Blind corners are covered with a Red Zone close to corner and Yellow zone enclosing the Red Zone. Corridors are covered as Green Zones and the rest is covered as Yellow Zones.

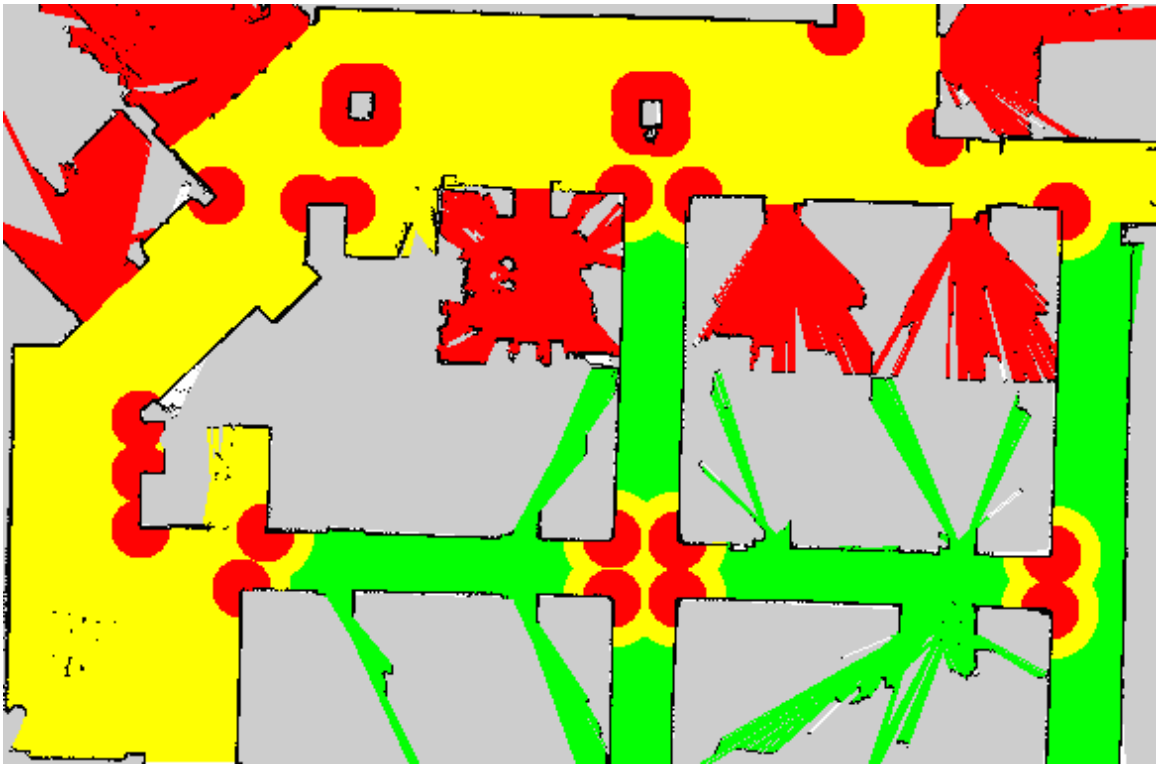


Figure 36: A speed map designed for the IRIM Lab at Georgia Tech is shown. The robot has to be relatively slow in red zones, can have moderate speed in yellow zones and is allowed to move relatively faster in green zones.

The speed map depicted in Figure 36 is designed by hand, however the speed map generation can be automatized using automatic room categorization and additional processing. Room segmentation has been proposed in an interactive fashion by Diosi [22], as well as automatically, especially for creating topological maps [84].

5.6.1 Results

In this section, we evaluate the effect of applying speed limits in the scenario shown in Figure 37(a). The robot is in an office environment and it has to turn a corner to reach the its goal. There is a person standing right around the corner and the person is not visible to the robot until it gets fairly close to the person. We had two conditions of speed limits:

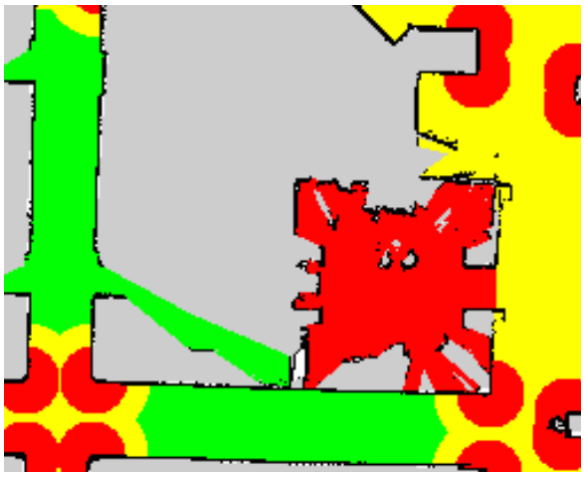
- Condition 1: Speed map is not used, the top speed of the robot was set at

$v_{max} = 1.0m/s$

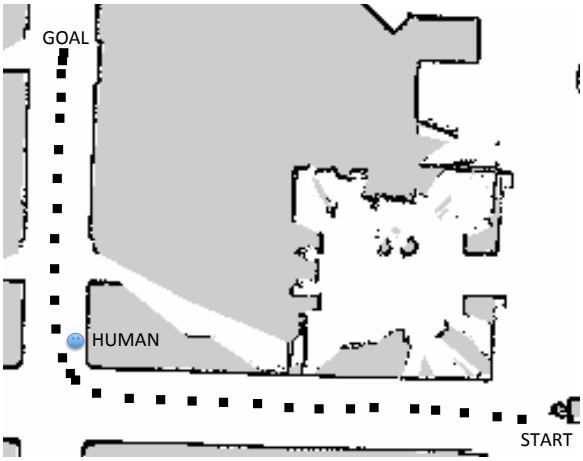
- Condition 2: Speed map in Figure 37(a) is used. The top speed varied in each zone as: $v_{max}(green) = 1.5m/s$, $v_{max}(yellow) = 0.5m/s$, $v_{max}(red) = 0.15m/s$

In both conditions, standard ROS Navigation is used, which found the lowest cost path on a costmap consisting of costs from nearby obstacles. The robot detected the person using our multimodal person detection and tracking method described in Chapter 2. We analyze the distance/velocity of the robot as it encountered the human and measure the time to reach to the goal as evaluation metrics.

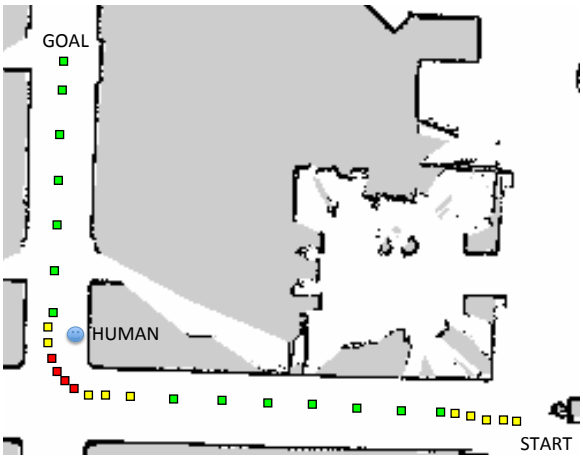
The trajectory generated in Condition 1 is shown in Figure 37(b). Note that the robot got dangerously close to the human while it turned the corner and it was traveling with about $0.3m/s$ when it detected the person. The trajectory generated in Condition 2 is shown in Figure 37(c). With this approach, the robot slowed down as it approached the corner, therefore allowing earlier detection of the human. The robot gave more space to the human in this case and the encounter speed was $0.15m/s$, half of the encounter speed in Condition 1. Considering the speed of the robot and distance to the human at the time of encounter, we can say that the robot behavior in Condition 2 was safer than the robot behavior in Condition 1. The second metric we measured was the time to reach the goal. The robot was more efficient with the proposed approach, reaching the goal in $28s$ in Condition 2 compared to $29.1s$ in Condition 1.



(a)



(b)



(c)

Figure 37: The section of the map that corresponds to a turn is shown in the figures. The trajectories resulting from the experiment comparing our speed maps approach with a fixed top speed are displayed. The robot has a fixed goal location. Right around the corner, there is a bystander human, who is not visible to the robot until the robot makes the turn. Points annotate robot position measured at fixed time intervals. a) Speed map of a corridor intersection at the second floor of College of Computing at Georgia Tech. b) Robot's top speed is fixed at $1.0m/s$. Note that the distance between robot positions are mostly constant. The robot gets very close to the bystander because it is moving relatively fast when it turned the corner. c) The robot is allowed to move with $1.5m/s$ in green, $0.5m/s$ in yellow and $0.15m/s$ in red zones. Colors of the sampled points on the path show the associated speed zone. It can be observed from the trajectories that the robot motion handled the corner turn safer in in c) than in b)

CHAPTER VI

PERSON GUIDANCE

The application of guiding a person to a location with a robot has many uses, such as giving tours in museums, showing a location to visitor and helping the visually impaired navigate. We think the guidance behavior is one of the most fundamental capabilities a socially interactive robot should have. A straightforward approach to guide a person would be the the stop-and-wait method: robot plans a path and executes it normally as long as the guided person is nearby, and stops then the person is outside a defined radius. However, this method of guidance may lead to sudden stops and would not be socially acceptable. A guide robot should consider the distance to the human and incorporate this information in its control strategy.

In this chapter, after reviewing the literature on guide robots in Section 6.1, in Section 6.2 we present our guidance method, which adjusts the speed of the robot as a function of the distance to the person. Then we present an application of person guidance in Section 6.3, specifically tailored for blind users.

6.1 Related Work

The earliest works in guide robots focused on long-term deployment of tour guide robots in public places such as museums. Burgard [12] presents the robot Rhino, that was deployed to a museum for 47 hours. The Minerva robot was an improved model over Rhino [125], and was deployed to a museum with an order of magnitude larger floor space. This robot was in operation for two weeks, and was able to have short-term interaction with people via head motion and facial expressions. Siegwart [117] presents a robot that was deployed in an exhibition for 6 months. Nourbakhsh [89] presents a project where four guide robots were deployed to museums for a period

of five years. The authors remark that it is indeed possible to deploy guide robots public places without supervision. All these tour-guide robots had various degrees of autonomy and generally was received with enthusiasm. However, in all of these works it was apparent that there is a need for research in the area of Human-Robot Interaction.

Pacchierotti [93] demonstrates an office guide robot, but the main focus is on passing people in corridors. Clodic [20] presents robot deployed in another museum. It was reported that a continuous interaction all along the guiding mission is fundamental to keep visitor's interest. Martin [75] studies the scenario of guiding a visitor in an office environment and focuses on robust person tracking. Pandey [95] focuses on the leave-taking of the guided person. The robot predicts the intent of the discontinuation of the task and either breaks the mission or searches for the user depending on the waiting time. Martinez-Garcia [77] focuses on the scenario of guiding a group of people with multiple robots at the same time. Garrell [32] works on a similar problem, where the task of the two robots is to control group of people and guide them. Another relevant scenario is the evacuation scenario, in which there is a danger and robots guide people to the safe a location [56, 103].

6.2 Guide Robot

In this section, we describe how the guiding behavior is achieved. The robot's higher level actions are governed by a finite state machine, as shown in Figure 38.

When the robot is first started, it is in the *NON-LOCALIZED* state. When the initial pose is given, either manually or by detection of the QR code (Section 5.3), the robot switches to *WAITING* state. This is the state where the robot actively looks input from users. When a person is closeby, robot approaches towards the person so the on-board tablet GUI is facing the potential user. If the person inputs a guide request, the robot displays destination and asks for confirmation. If the user confirms

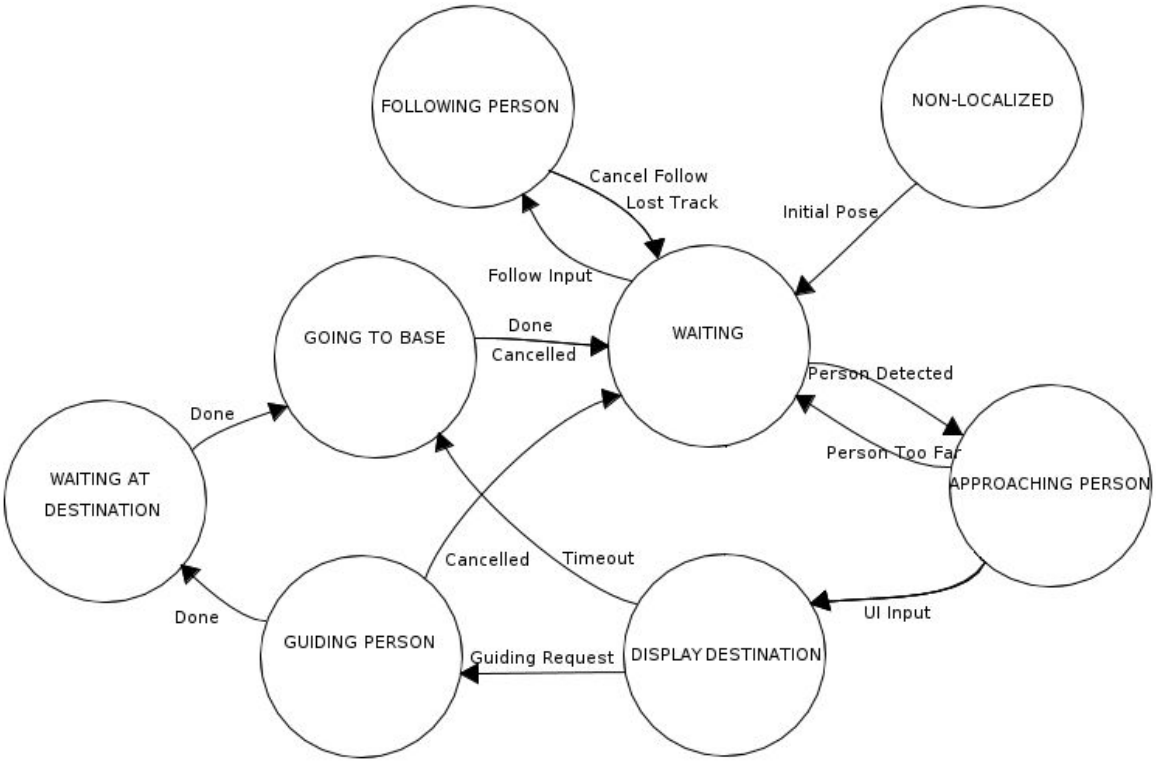


Figure 38: Finite State Machine for the Guide Robot

the goal, then the robot goes in to *GUIDING PERSON* state. When guiding is completed, robot waits for a while at the guidance destination and goes back to the base. At any time, a user can cancel an operation or enter a new command.

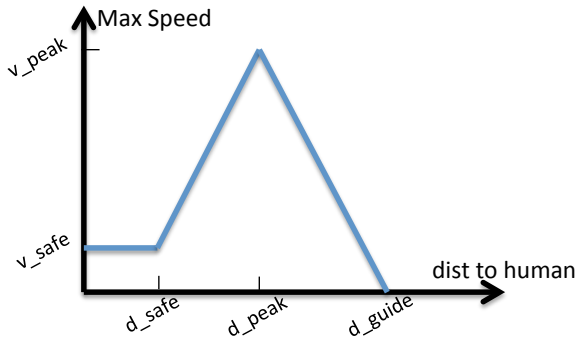


Figure 39: Speed profile of a person guiding robot as a function of the distance to the user.

After a person guidance request is received from a higher level process, the robot first plans a path using a navigation planner such as ROS Navigation or our navigation

planner presented in Chapter 5. The robot continues on its path while constantly monitoring the distance between itself and the guided person, and adjusts its speed accordingly. A variable speed profile so that the robot can keep up with the person and the motions of the robot is smoother. We define a speed function that is a function of the distance between the robot and the user. This speed profile is shown in Figure 39. The robot moves at a low speed v_{safe} if the human is dangerously close. The speed is peaked at distance d_{peak} and the robot stops if the distance is larger than d_{guide}, which may indicate that the human is not interested in being guided. Note that v_{peak} is capped by the speed limits in the environment, provided by the speed map approach presented in Section 5.6.

6.2.1 Pilot Study

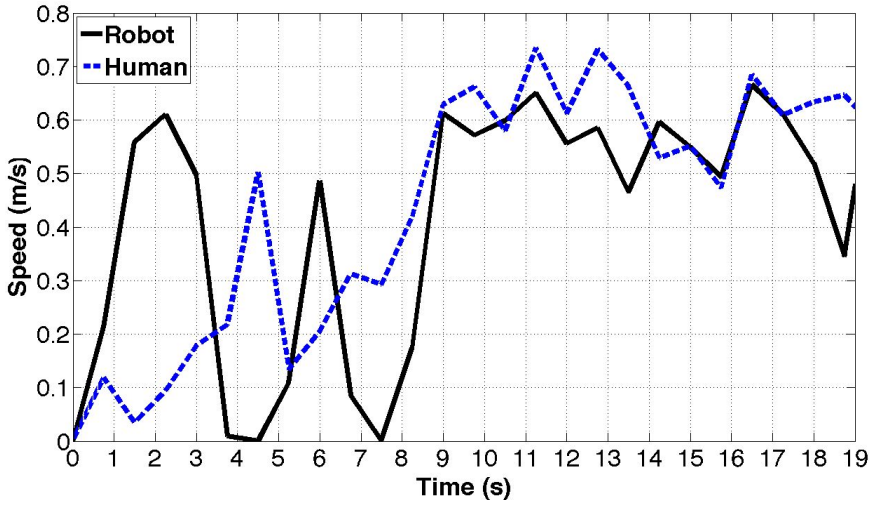
We evaluate the proposed guidance method in a comparison study. The task of the robot is to guide a person in an office environment. Figure 37(a) shows the part of the map and the goal of the robot. There are not any other people in the environment other than the guided person and ROS Navigation is used for point-to-point navigation planning. The robot actively guides a user to this goal position, in the following two conditions:

1. Stop-and-wait: The robot stops as long as the distance to the user is higher than a threshold

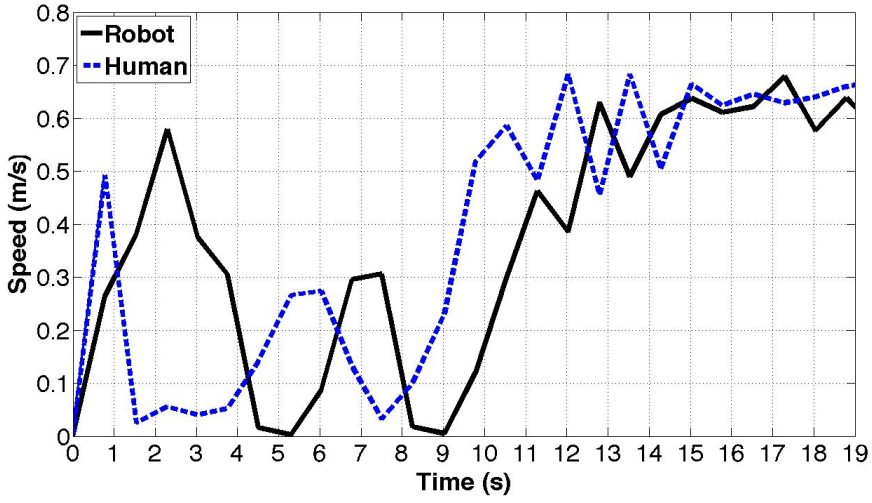
2. Proposed method: The maximum speed of the robot is altered as a function of the distance to the user. The velocity profile shown in Figure 40 is used with $d_{guide} = 1.7$m, $d_{peak} = 0.9$m, $d_{safe} = 0.1$m, $v_{safe} = 0.1$m/s, $v_{peak} = 1.0$m/s.

In the experiment, when guiding was enabled, the human first waited until the robot stopped at d_{guide}. Then he took a step and waited for a second time, and then started following the robot. As the evaluation metric, we measure the instantaneous speeds of the robot and the human.

6.2.2 Results



(a)



(b)

Figure 40: Comparison of robot and human speeds are shown for two person guidance methods. a) Stop-and-wait guidance with ROS Navigation b) Proposed guidance method. Accelerations were less steeper in b).

The comparison of robot speeds is given in Figure 40(a) for Condition 1 and Figure 40(b) for Condition 2. Between $t = 0$ and $t = 9s$, the accelerations were higher for the stop-and-wait condition. Robots that exhibit high accelerations will likely be perceived as unsafe, therefore the proposed method exhibits more socially acceptable behavior. Moreover, after the person started closely following the robot ($t > 9s$), our

approach is better at mimicking the speed of the human.

6.3 Application To Blind Users

In this section, we present a person guidance application that is specifically tailored for guiding blind users. Our approach consists of planning a path for the user and applying vibrations via a haptic belt to keep the user on the path.

6.3.1 Tactile Belt

In the previous section, we assumed that the guided person could observe where the robot is. With a blind user, this assumption does not hold. Therefore we need a mechanism to give directions to the user. Readily available options for assistive interfaces are limited to Braille or devices that presents content with speech synthesis. These ways of presenting information have difficulty dealing with representing spatial information. We also think visually impaired individuals would prefer a non-spoken interface because they mostly rely on their sense of hearing in daily life. We therefore use a tactile belt for navigation guidance, because it can represent directions and rotations, be worn discreetly and does not occupy the hearing sense.

The belt has 8 pancake vibration motors, linearly spaced around the waist, and the motors can be controlled asynchronously via an Arduino board. We used two distinct vibration patterns to control the person's movements:

1. Directional Movement: When the guided person should move in a direction

2. Rotational Movement: When the guided person should turn around self

The vibration patterns that induce directional and rotational movements in the human can be specified in many ways. We evaluated four patterns in each category with a user study. The details of this user study as well as a survey about the usability of the tactile belt can be found in Appendix B. The user study showed that the

directional motion pattern with the highest recognition rate and least reaction time was the **TWO TAPS** pattern, which is illustrated in Figure 41. For the rotational motion pattern, we used the continuous rotation with a single motor, illustrated in Figure 42.

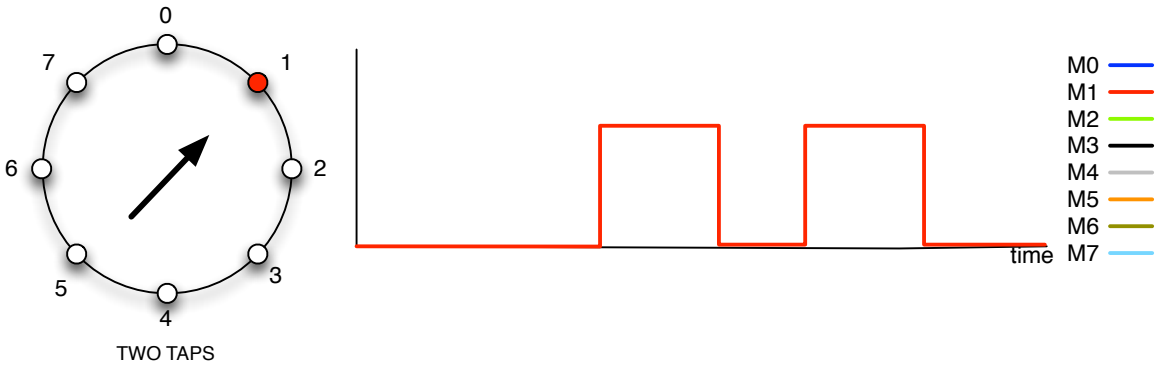


Figure 41: The vibration pattern applied by the Tactile Belt to induce directional movements. A motor is fired for a duration of $250ms$, inactivated for $250ms$ and fired again for $250ms$.

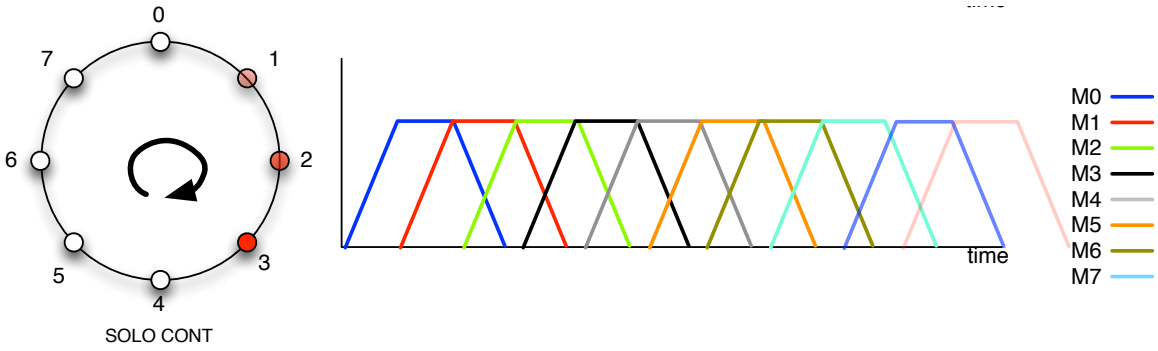


Figure 42: The vibration pattern applied by the Tactile Belt to induce rotational movements. The consequent vibrations motors are fired consecutively, starting from left for CW and right for CCW rotation.

A special stop signal is applied to the belt when the user reaches the destination. Stop signal is implemented similar to **TWO TAPS** pattern except all the motors are activated instead of one.

6.3.2 Planning the Path of the User

We model the person as a non-holonomic robot and plan a path for him/her. Vibration signals are applied to influence the motion of the user and keep him/her on the path. Coupled with our person tracking module, the 'robot' stays localized in the map and with respect to the path. The path is re-planned every second to deal with possible deviations. Next section is concerned with how the linear and angular velocities extracted from the planner are converted to the vibration patterns.

6.3.3 Velocity to Vibration Mapping

Given a desired velocity that the 'robot' should execute, we first determine if a directional or rotational vibration pattern should be applied by the belt. If the linear velocity is dominant, then the human should walk towards that direction. If the angular velocity is dominant, the human should rotate around self. If both the linear and angular velocity is close to zero, the human should not move. To calculate which motion is appropriate, the 'robot' is simulated using Equation 16. If the distance the 'robot' took is larger than a threshold, then a directional vibration pattern is used. If it is less than the threshold, a rotational pattern is used. If both of the velocities are small enough, then no vibration is applied. When the human reaches the goal, the stop signal is applied with the belt.

6.3.4 Demonstration

We demonstrated that our system can successfully guide a blindfolded person to a goal location in a room. The experimenter manually provided several goal poses using the GUI. Note that since the system re-plans frequently, therefore the planner is able to accommodate dynamic obstacles and compensate unpredictable motions of the person. The demonstration steps as well as their explanations are shown in Figure 43.

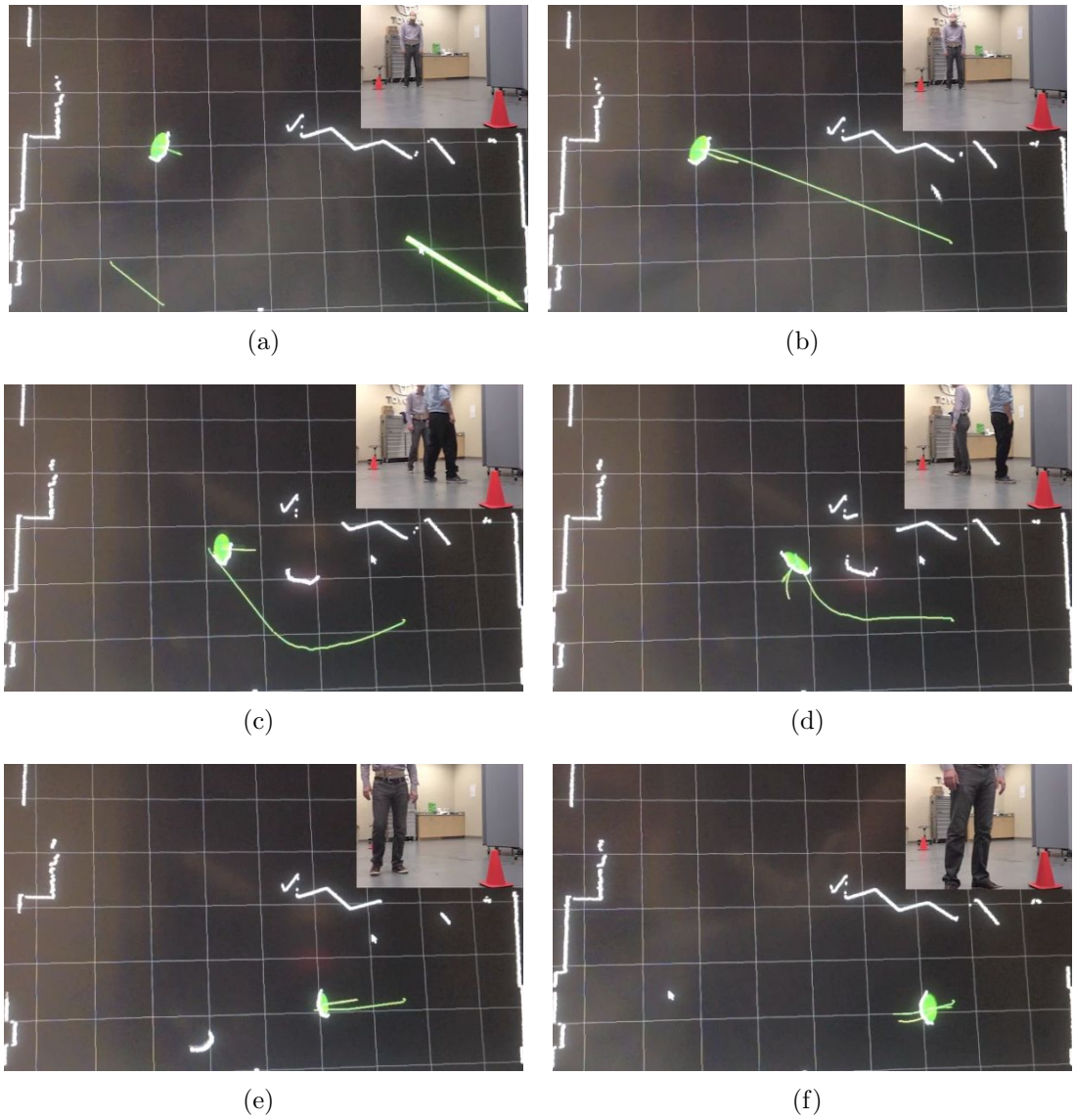


Figure 43: Autonomous guiding of a blindfolded person using the tactile belt. a) The guidance starts. The user is blindfolded and is standing at the left of the screen. The human detection system detects him and places an ellipse marker with an arrow depicting his orientation. The operator gives a goal point by clicking on the screen. The goal point is the right traffic cone, and given by the big arrow. b) The system autonomously generates a path for the user. As seen in the picture the path is collision free. At this stage the belt begin to vibrate towards the front of the user. c) An unexpected obstacle (another person) appears and stops in front of the user. The system detects the other person as an obstacle, and reevaluates the path. A new path going around the obstacle is immediately calculated and sent to the user by the belt. d) The user receives a rotation vibration modality, and begins to turn towards the new path. And follows this path from now on. e) The obstacle leaves. The path is then reevaluated and changed. The user receives forward directional belt signal, and advances towards the goal. f) The person reaches to the vicinity of the goal and stop signal is applied.

CHAPTER VII

SITUATION AWARE PERSON FOLLOWING

Simply put, *Situation Awareness (SA)* is knowing what is going on around you. Endsley [26] defines three steps for SA: perception, comprehension and projection. Perception is detecting the situation by perceiving cues, comprehension is combining and interpreting information and projection is forecasting future events. In this section, we discuss SA for person following behavior for mobile robots. In the Chapter 3, we presented the basic following behavior, where the robot follows the person strictly from behind, while maintaining a fixed distance. The related works on person following discussed in Section 3.1 mostly use the same principle: the robot uses the person to calculate a target position and blindly follows the human irrespective of the situation. Although this method is sufficient for some scenarios, it can easily lead to socially awkward situations. For example, consider the case that the followed person stops just outside a door. In this case, the robot would occupy the doorway, blocking other people's passage, however thinks it is doing its task well because it maintains a fixed distance to the user. If the robot knows what the user intends to do, it can anticipate those actions and suitably adjust its behavior. Person following can be used in different contexts, such as for carrying luggage in airports or groceries in a supermarket. We showed in Chapter 4 that semantic maps that include landmarks and waypoints could be used to communicate goals between the robot and the user. The stored semantic information can also be used to facilitate robot navigation. We specifically focus on three scenarios of situation aware person following:

1. Joining a group of people

2. Labeling landmarks during the Tour Scenario

3. Passing Doors

The robot behaviors for each of these scenarios are modeled and executed via triggered events, inspired by Cakmak's framework [14]. Handling of an event during following is implemented as a sequence of four phases:

1. Signal: Robot detects an event using perceptual cues
2. Approach: Robot moves to a position better suited to the task
3. Execution: Robot and/or Human execute the task
4. Release: Robot detects the end of event and continues with the basic following behavior

With this methodology, robot uses the three steps of SA: Perception for detecting the start and ending of an event, Comprehension for interpreting where it should move to and what the task is, and Projection to estimate the future goals of the person. In this chapter, we study three scenarios that could be encountered while the robot is following the person. In Section 7.1, we first show how the robot moves when the followed person stops and talks with someone else. In Section 7.2, we study the following behavior for labeling landmarks during the Tour Scenario. In Section 7.3, we look at how the robot should handle door passages.

7.1 Joining a Group

While the robot is following a person, one of the scenarios that could be encountered is when the followed person interacts with some other person. If the basic following method is used, the robot would stay behind the person and that could lead to an awkward formation where the robot is left out of the group (see Figure 44 for an example). Our proposed solution to this problem is for the robot to join the group

Figure 44: The problem that occurs when the followed person stops and interacts with another person is shown. The robot is left out of the group when it does not detect or react to this social situation.

of people using engagement rules. Joining a group has been addressed previously [1, 112], however not in the person following context.

Our solution to this problem is to choose a position for the robot to stay during the conversation. Our choice of the position is guided by Kendon's F-Formations [53]. Kendon studied how people assemble and what formations they assume while they are interacting. In this representation the space is divided in three regions as can be seen in Figure: 45. *o-space* is the empty space surrounded by people, *p-space* contains the participants and *r-space* is the outer area beyond the *o-space*. Kendon studied commonly encountered formations such as L-formation, circular, face-to-face and side-by-side formations.

In our approach, when the robot detects the existence of a group formation, it samples poses on the *p-space*, scores them and chooses a collision-free goal pose. When the interaction ends and the group formation is broken, the robot continues the basic following behavior. The execution phases for this behavior as well as the behavior transitions are given in Table 7.

We follow the Signal/Approach/Task/Release procedure for the design of this behavior. The Signaling phase is triggered whenever the user is close to another

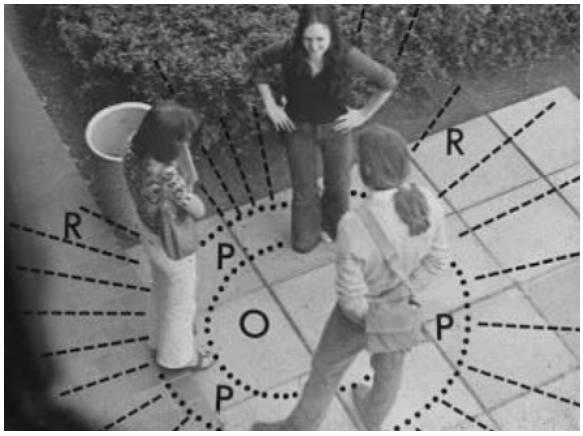


Figure 45: Definition of space around people according to Keldon's F-Formation representation of group formations

Table 7: Conditions to trigger phases when the user is involved with the Joining a Group Event during following.

Signal	$dist(user, groupcenter) < threshold$ $speed(user) \sim 0$ person roughly facing group
Approach	Optimal Goal: in the *p-space* in the circular formation
Execution	The group interacts with each other, including the robot
Release	$dist(user, groupcenter) > threshold$

person or a group of people. The user must have close to zero speed to enable signaling for this behavior, because the user may walk past the group. After the robot detects the signal, we sample positions around the group to locate a "suitable" goal pose for the robot. A pose that is collision free but that gives the robot highest chance of interaction is favored. We use a utility function that scores candidate goal points. Intuitively, a pose that is close to both people and could see both is considered a suitable goal pose. We sample points 360° around the *p-space*, for a fixed sampling resolution.

Every sampled position p has a score of:

$$Score(p) = 1.0 - Cost_{visibility}(p) - Cost_{obstacle}(p)$$

(a) (b)

(c) (d)

Figure 46: Demonstration of the robot joining a group when the followed person interacts with a group of people. The robot is initially following the user throughout the environment and keeping a fixed distance of $1.2m$ to the user. a) Signal phase: The user has stopped and is in the cloxe proximity to to another person. b) Approach phase: The robot calculates and navigates to a goal position, so it can potentially interact with people in the group. c) Execution phase: The interaction happens. c) Release phase: user moves away from the group and basic following behavior continues.

Where we define the costs as:

$$\begin{aligned} Cost_{visibility}(p) &= dist(user, person)/(dist(p, person) - dist(p, user)) \\ Cost_{obstacle}(p) &= max(local\_cost(p), global\_cost(p)) \end{aligned} \tag{17}$$

The local and global costs are fetched from the ROS costmaps, normalized to [0.0-1.0] interval. The sample with the highest nonnegative score is chosen as the goal position. The orientation of the robot is chosen as looking toward the center of all

the people in the group. After the goal pose is determined, the robot is commanded to navigate there. During the Execution phase, interaction that potentially involves the robot occurs. Whenever the user leaves the group, then the Release phase is executed. The robot continues tracking the user with the basic following method.

This behavior is implemented on a real robot and snapshots from this demonstration is shown in Figure 46.

7.2 Following For Labeling

One of the problems we found out during the Tour Scenario experiments was that as the robot is following the user, it does not have any information about the task. This leads to awkward situations when the user wants to label a landmark or object, because the robot can not perceive the pointing gesture or the object/landmark of interest at the same time when it is following from behind all the time. This situation is illustrated in Figure 47.

The robot behavior can be more intelligent in those cases if the robot can predict ahead of time when the user is going to label landmark. The robot follows the phase transitions defined in Table 8 for this scenario.

Table 8: Conditions to trigger phases when the user is involved with the Landmark Labeling Event during following.

Signal	$dist(user, convexhull(landmark)) < threshold$ $speed(user) \sim 0$ person roughly facing landmark
Approach	Optimal Goal: Close to both the landmark and person, facing in between
Execution	User points and labels landmark
Release	$dist(user, convexhull(landmark)) > threshold$

Our approach relies on predicting when a labeling is going to happen, and position the robot base so it has a better chance to perceive both the pointing gesture and the object/landmark of interest. Our methodology for this scenario is similar to the

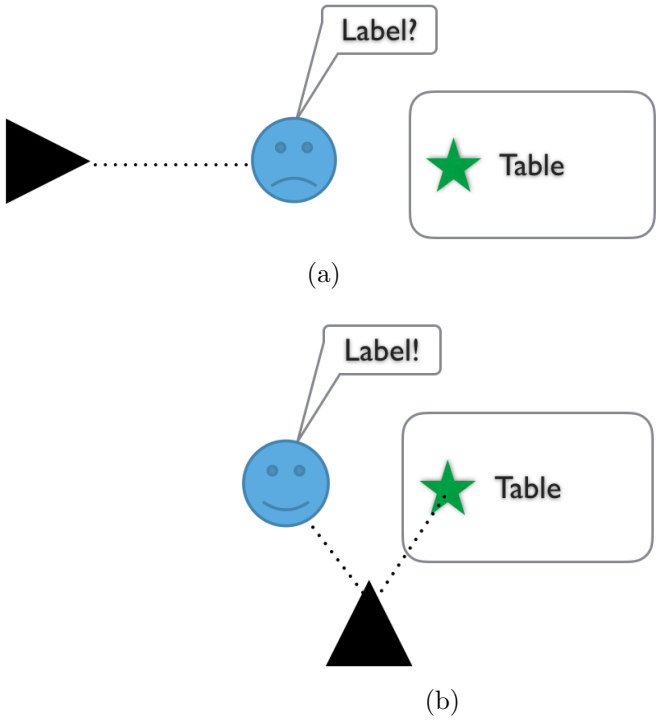
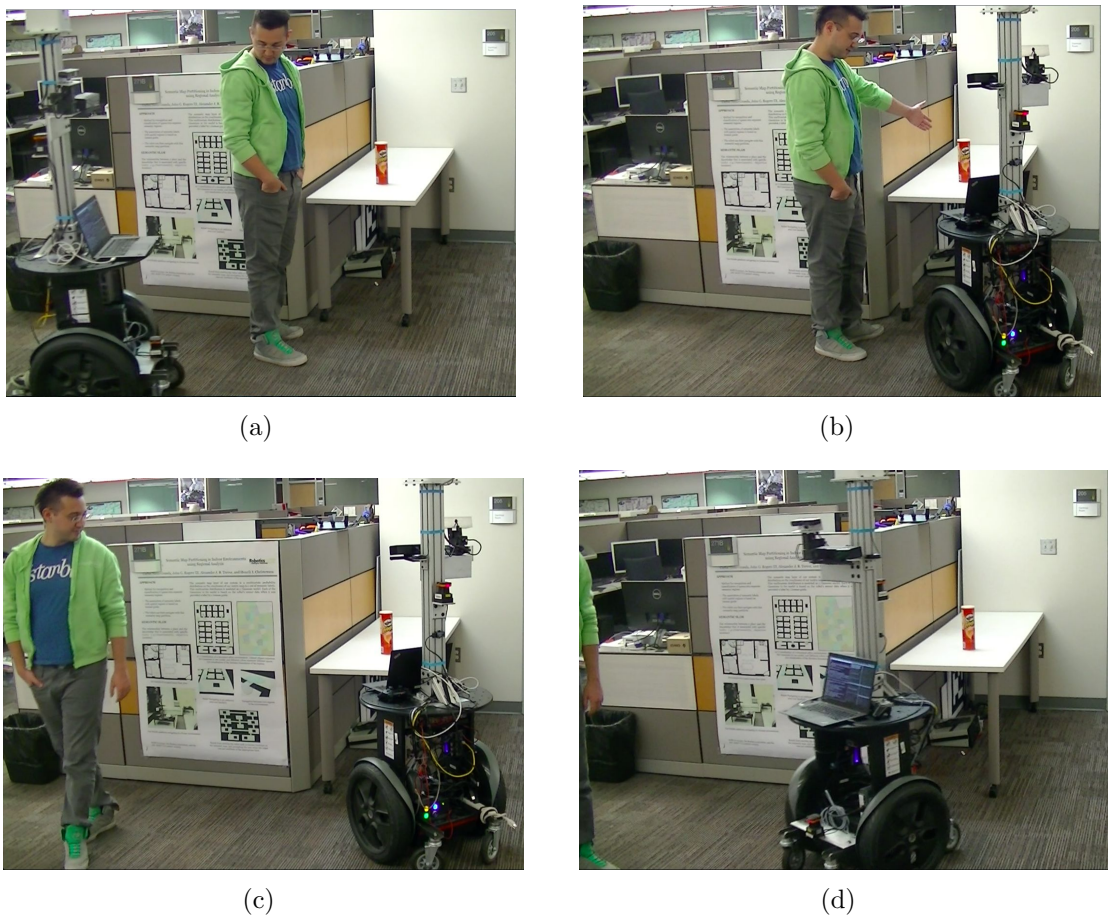


Figure 47: a) A common problem we encountered during the Tour Scenario. The user wants to label an object on the table, however the robot does not understand this intention and stays behind at a fixed distance to the user. b) Our solution is to move to a location that has the visibility of both the user and the object.

joining a group behavior as presented in Section 7.1. The difference is that, instead of sampling points around the group of people, for the landmark labeling behavior we sample points around a circle that includes the user and the centroid of the convex hull of the landmark.

To go over the phases, this behavior is triggered when the user has stopped nearby an unlabeled landmark or object and facing it. A goal position that is close to both the landmark and person, facing in between is found and the robot navigates there. Then the user can execute the labeling task via pointing gestures. After the task is completed, the robot waits until the user to leaves the vicinity of the landmark. When that happens, the robot continues following the user. If, during any of the phases, the person tracking fails, it informs the user so following can be restarted. The phases and conditions for this behavior are summarized in Table 8. The situation awareness



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

Figure 48: Demonstration of situation awareness for the Tour scenario. The robot is following the user throughout the environment and keeping a fixed distance of $1.2m$ to the user. a) Signal phase: The user has stopped and is in the cloxe proximity to the convex hull of the table. b) Approach phase: The robot calculates and navigates to a goal position, so it can perceive the pointing gesture and target. Execution phase: The user points out to the object on the table. c) Release phase: user moves away from the table d) Basic following behavior continues.

for labeling landmarks is implemented on the Segway robot for the Tour Scenario. Snapshots from a demonstration for this behavior is shown in Figure 48.

7.3 Door Passing

When the user approaches a door during following, the situation can easily become problematic if the robot continues with the basic person following behavior. For example, if the user intends to close an open door or open a closed door, the robot

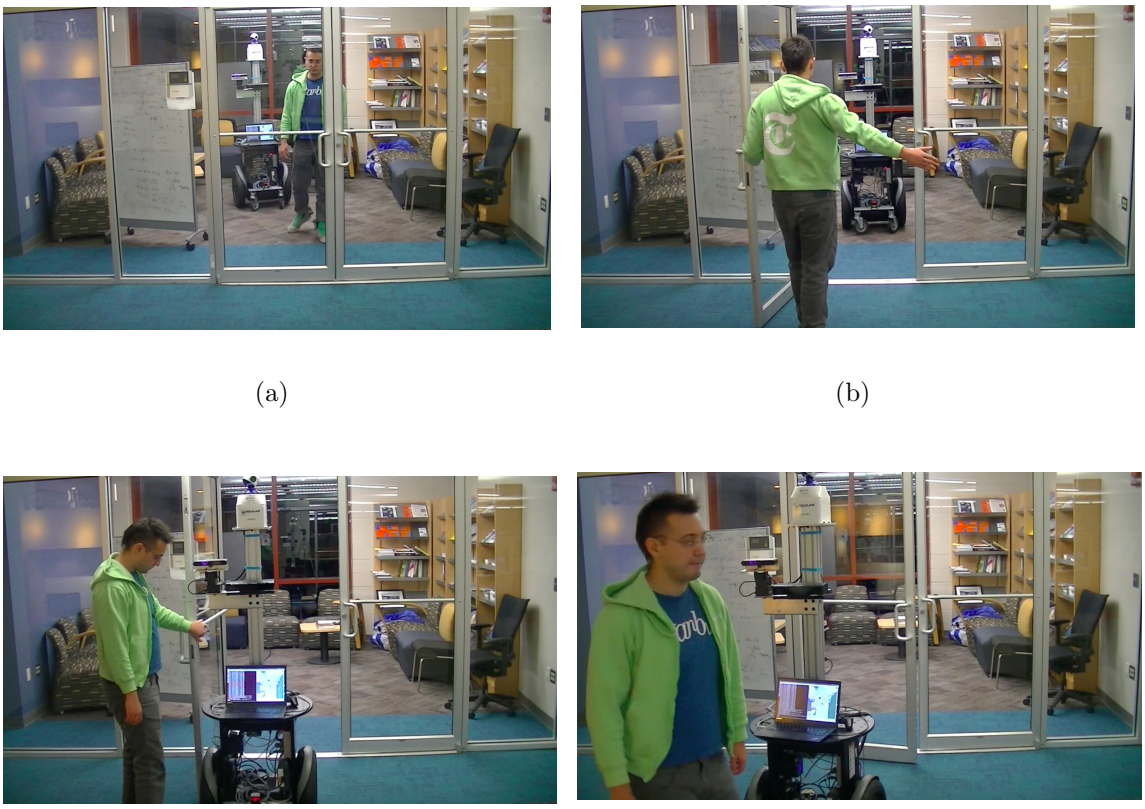
might end up blocking the movement of the door. Moreover, a deadlock situation occurs when the user wants to go through a door with spring-loaded hinges. In that case, the user would need to hold to door to keep it open, and because the distance between the robot and the user is less than the following threshold, the robot would stay still and won't pass the door. A robot SA should be aware of this possibility and take appropriate action.

Table 9: Conditions to trigger phases when the user is passing through a door during following.

Signal	$dist(user, convexhull(door)) < threshold$ $speed(user) \sim 0$ User performs pointing gesture towards the passage
Approach	Optimal Goal: A position on the other side of the door that doesn't block the doorway
Execution	Robot and user meet at the same side of the door
Release	$dist(user, convexhull(door)) > threshold$

Opening, closing or passing through a door, and detecting these actions require a sophisticated recognition and system. However the robot can assume that any of those actions are possible when the user is approacing the door. In our approach, the robot can can continuously monitor the user's proximity to the doors using the semantic map, if the user labeled door landmarks beforehand. Our approach can handle doors with spring-loaded hinges, even though it does not have a model of the door except the convex hull of its plane.

The phases and conditions for door passing situation are summarized in Table 9. The robot takes action when the user is nearby a door and performs a pointing gesture towards it, to signal that the robot should pass from the door (Signal Phase). If the action is not signaled, the robot continues with basic following during the door passage. After the detection of a pointing gesture, a goal position is calculated (Approach Phase). The goal positions are sampled on the other side of the door, that



(a) (b)

(c) (d)

Figure 49: Demonstration of situation awareness for door passing during person following. It is assumed that the user previously added the door as a labeled landmark to the semantic map via the Tour Scenario. This is a swing door with spring loaded hinges, so it would close if not kept open actively. a) The robot is following the user by keeping a fixed distance to the user. b) Signal phase: The user has stopped, is in close proximity to the door and performed a pointing gesture toward the other room. c) Approach Phase: The robot passes the door while the user is holding the door d) Release Phase: User has more than a threshold distance to the door, and robot continues with the basic following.

is guaranteed not the block the opening/closing of the door. A collision-free position with the least obstacle cost sample is chosen as the goal point. Note that while the robot is moving, it does not aim to keep fixed distance to the user anymore. After the robot reaches the goal, it waits for the person to pass the door (Execution Phase). After the user moved away from the door, the basic following behavior takes over. The situation awareness for this scenario is implemented on the Segway robot. The

snapshots from the behavior can be seen in Figure 49.

CHAPTER VIII

CONCLUSION

Most of the robots that are in operation today are on factory floors in separation from people for safety. However, robots that work for and work with humans have a great potential. Navigation is one of the most fundamental capabilities for a mobile robot. Mobile robots occupy the same space with us, therefore should respect spatial rules of engagement. Humans are good at adjusting their spatial relationships with each other in social situations, therefore it is reasonable to use human-human interaction studies to design navigation behaviors. State-of-the art robots use only a metric map for navigation, however utilizing semantic maps that includes objects and landmarks can enable new navigation behaviors. This thesis addressed several challenges: interactive semantic labeling, social navigation behavior design and situation awareness for navigation.

To conclude, we re-state the thesis statement: Non-expert users can effortlessly interact with and control a mobile robot through the use of semantic maps and spatial rules of engagement. Having an app interface (Section 4.3), use of QR codes for automatic initialization (Section 5.3), use of natural gestures and autonomous behavior for telepresence robots (Section 3.3) shows our focus on effortless non-expert interactions. The benefits of semantic maps were demonstrated in accepting goals from users (Section 5.4), adjusting the speed of the robot (Section 5.6) and for situations such as door passing (Section 7.3) and landmark labeling (Section 7.2). Spatial rules of engagement we used include personal spaces and group interactions (Section 5.5).

The rest of this chapter elaborates on the key contributions of this thesis, discusses the results and observations, outlines future work for the next steps for each topic,

and concludes with final remarks.

8.1 Interactive Map Labeling

At the core of this thesis lies the idea of the Tour Scenario, where the user takes the robot on a tour and teaches objects and places. Tour Scenario provides the opportunity for a mobile robot to acquire spatial representations from human users and later utilize them for better usability and effective navigation. This scenario could be used the first time a robot is brought home, for example if it is purchased brand new. That would also give an opportunity to users to interact with the robot and get familiar with their new companion. Tour Scenario could also be used after the first use, for example whenever the layout of the house changes, or when a new object or furniture is purchased. Therefore one can think that the Tour Scenario could be used for long-term interactions.

In our work, Chapter 4 presented our semantic map representation as well as the steps of the Tour Scenario. The main contribution of the chapter is the use of an interactive labeling process to add a multitude of features to the map: objects, waypoints and planar surfaces. Use of the labeled semantic features as navigation goal enables communicating goals in natural language. Another contribution is the use of natural pointing gestures for interactive labeling and a model of determining the likelihood of which object the user intends to point at.

We think that future commercial domestic robots will have the "familiarization task" in some form or another, because it is important for the robot to learn the objects and locations its user cares about. We believe that the Tour Scenario presented in this thesis presents a feasible solution for this task. Semantic maps creates a common ground between users and robots, and will enable new set of service robotic tasks. Use of annotated semantic maps can also be studied in other contexts, such as to improve SLAM [132], visual object search [104], and direction giving [61].

8.2 Social Navigation Behavior Design

This thesis touches upon several considerations and use cases for navigation behavior design for social robots. These include people-aware navigation, person following and guidance, and situation-aware navigation.

8.2.1 People-Aware Navigation

Mobile robots will be entwined in our daily lives in the future. These robots will operate in environments designed for humans, therefore people would expect them to move in a socially acceptable manner. Possibly the most common navigation task would be point-to-point navigation: the robot navigates to a goal position from its current position, without explicit communication with bystanders. The social part makes the social navigation algorithms interesting and different than classical motion planning methods. Research in this field suggested using safety and comfort as criteria for safe and comfortable navigation among people. This requires multi-disciplinary effort in both robotics and psychology research. The research in this area is concentrated in two camps: 1) Social costmap design on geometric path planning. 2) Reactive and short-term behavior design to move in the vicinity of people. This thesis contributes to both: with costmap design for people-aware navigation and reactive behaviors for person following and guidance.

Our work on point-to-point people aware navigation was presented Chapter 5. We demonstrated that by using anticipation, the robot exhibits human-like navigation behavior, can reach its destination in less time and can find solutions to problems that are insolvable by standard planners. We believe that our method will increase the predictability of robots motions.

Currently, social navigation techniques is not viewed as an essential capability for mobile robots. In the future, robots will be a part of our daily lives and social navigation planning will be a necessity. Human-human spatial interaction studies are

shown to be helpful in designing interactive navigation behaviors. Currently, it is not very straightforward to measure how socially aware the navigation planners are, mainly because a planner can exhibit many different behaviors in different contexts. We think a set of standard evaluation methods will need to be developed for this area of robotics research. As future work, we would like to evaluate and compare our planner with other approaches.

8.2.2 Person Following and Guidance

An effective person following behavior is an essential to the Tour Scenario. It enables the robot to keep up with the guide-person during the guided exploration of the space and move around the environment. Person guidance is one of the applications that could be used after the environment representation is acquired. It can be used in scenarios where the robot conveys information to the user, (i.e. museum tours), or to take a person to a location in an unfamiliar environment (i.e. guiding people in airports).

Our work on person following and person guidance were presented in Chapters 3 and 6, respectively. We presented a basic method for person guidance, and presented a complete tour-guide robot system. We also showed that our guidance approach results in more gracious motions compared to standard ROS Navigation. A specific application, targeted on wayfining for blind users is studied in depth. We showed that it is technically feasible to guide a blindfolded person in indoors using vibrations applied by a haptic belt.

We also presented a basic following behavior that keeps a certain distance from the user and studied use of autonomous person following for telepresence robots. Current commercial telepresence robots do no have autonomous features and our user study showed that people favor the use of autonomous navigation features.

Interactive navigation behaviors such as following and guidance is useful for several tasks. In the literature, such behaviors are limited to specific scenarios, such as hallway encounters [94] and joining a group [1]. We presented basic behaviors that are aimed for general human environments. We further delved into specialized applications for particular user communities. Our approach relies on developing different planners for each behavior, however in the future it may be interesting to develop a unifying planning framework that can handle different types of interactive navigation scenarios.

8.2.3 Situation Awareness for Navigation

In this thesis, one of contributions is enabling of situation awareness for navigation. We introduced *speed maps*, that specifies robot top speeds in an environment. The speed limits were determined by partitioning the map into corridors, rooms, and corners. Depending on the environment, speed limits can vary. For example, the robot should move slower in a hospital room but can travel faster in an empty corridor. We showed that speed maps not only can reduce the impact of a potential collision, but it can also reduce travel time. Robots yielding to speed maps also can potentially be can perceived safer. In the future, we envision a comprehensive set of rules for robot navigation, essentially acting as traffic rules for mobile robots.

We also demonstrated situation awareness for person following. Our approach relies on designing specialized navigation behaviors upon detection of an event (i.e. user passes a the door or labels a landmark). We split an event into phases in an attempt to standardize the detection of events and execution of actions. Not many works in this area addressed door passing for navigation [144]. Our approach allows handling spring-loaded doors graciously and facilitates the Tour Scenario.

Situation awareness demonstrated in this work only scratches the surface of the possibilities. A planner can potentially consider many other different factors, such

as the identity of users, time of the day, cultures and task instructions. Another interesting possibility is to use machine learning to learn the preference of its users.

8.3 Discussion

Despite the encouraging demonstrations and results achieved throughout this thesis, there are several limitations of the presented research. These limitations include short-term challenges in perception and planning as well as long-term challenges such as evaluation and integrated system design.

Among the short-term challenges is the perception of people. Even though our body of work on people detection and tracking was sufficient to carry on experiments, it is not of production quality. Our current system can detect only standing people and has a usable range of about 3-4 meters, which can be significant limitations for some applications. A flexible yet general-purpose person tracking method should be able to accommodate different sensor configurations, robustly track multiple people in the presence of heavy occlusion and recognize people without necessarily seeing their faces. The common practice in robotics research is to develop a person tracking method from scratch for a specific purpose and sensor suite or to use a third party tracker as a black-box. We think that as a community we should set common standards and build on previous efforts.

Another significant limitation of the work in this thesis is the lack of thorough validation and evaluation of the designed navigation behaviors. These behaviors should be assessed both from a technical point of view and from a user experience perspective. The technical assessment is easier to do, as simulations and quantitative measures could be used for evaluating navigation behaviors. Apart from building an accurate theoretical model, the real performance of the robot behavior needs to be taken into account especially for a complex mobile robot system moving in dynamic

environments and situations. User studies can be designed to quantify the user experience that can not be measured from technical assessment. Even though user studies gives an initial opinion about a design, they do not capture the user experience of a long-term interaction with a product. Therefore we ultimately need to have the robots in the field to have an realistic opinion about the behavior design. One way of improving the robot behaviors could be a fast design iteration using feedback from early adopters of a the mobile robot product.

We designed navigation behaviors such that there is only an implicit interaction between the robot and humans during the navigation. The robot signaled its intention only through motion and did not explicitly communicate with people. However, additional modalities such as speech and gaze could be utilized to complement the motions of the robot.

Algorithms and behaviors presented in this thesis used some fixed parameters. Most of the time, these parameters were determined empirically and with trial-and-error experiments. However, this is not the best methodology to tune the parameters. One can use data science tools such as machine learning to tune the parameters of the system. Machine learning could also be used for a wider range of behaviors, such as to learn socially acceptable paths or behaviors or to adapt to specific user preferences.

In this thesis, we studied a number of navigation behaviors including point-to-point navigation, person following, guidance. We delved more into the person following by studying at specific scenarios that could be encountered person following: passing doors, joining a group and landmark labeling. The choice of which scenarios are worthy to address was guided by our initial tests. For example, as we were testing the Tour Scenario, the guide person had to scramble to adjust his own location so that the robot could see both the landmark and the user. Although we think the scenarios we studied are essential for person following, it is likely that user studies or real use could reveal that different following behaviors are needed. Furthermore,

the tour guide robot and situation aware following was implemented using a state machine, which requires a pre-defined event to happen to change the behavior. Although state machines are easy to use because of their simplicity in representation, more sophisticated architectures could be used going forward such as subsumption, hierarchical task planning or petri nets.

The hardest part of the research that went into this thesis was the system integration and implementation. A great deal of work went to get the robot behaviors right and to make all the different modules work at the same time. Robots fail and fail often. Robots has to operate with reasonable robustness under dynamic conditions in real-world operation. Robotics research should be complemented with solid engineering to create great robot products.

8.4 Final Remarks

The reasoning methods described in this thesis are vital for robots to navigate autonomously among people. If, in the future, robots co-exist with us, assist people in daily tasks, help the elderly and carry our bags for us, they will need to navigate safely without making nearby people uncomfortable. The feasibility and reliability of those applications will determine the business value of mobile domestic service robots. In this thesis, we showed proof of concept for enabling behaviors and that there is potential for commercialization for both navigation in general environments and for specialized applications such as telepresence robotics and aiding the blind. We expect that future work inspired by the concepts presented in this thesis will allow robots to exhibit intelligent navigation behaviors.

APPENDIX A

TELEPRESENCE STUDY SURVEY

1) **I was able to understand what the experimenter was saying**

[ ] Strongly agree
[ ] Agree
[ ] Somewhat agree
[ ] Neither agree or disagree
[ ] Somewhat disagree
[ ] Disagree
[ ] Strongly disagree

2) **The user interface was easy to operate**

[ ] Strongly agree
[ ] Agree
[ ] Somewhat agree
[ ] Neither agree or disagree
[ ] Somewhat disagree
[ ] Disagree
[ ] Strongly disagree

3) **The motions of the robot were natural to have a conversation**

[ ] Strongly agree
[ ] Agree
[ ] Somewhat agree
[ ] Neither agree or disagree
[ ] Somewhat disagree
[ ] Disagree
[ ] Strongly disagree

4) **I was worried about the robot colliding with someone/something**

[ ] Strongly agree
[ ] Agree
[ ] Somewhat agree
[ ] Neither agree or disagree
[ ] Somewhat disagree
[ ] Disagree
[ ] Strongly disagree

5) **I was able to pay attention to the story the experimenter is telling**

[ ] Strongly agree
[ ] Agree
[ ] Somewhat agree
[ ] Neither agree or disagree
[ ] Somewhat disagree
[ ] Disagree
[ ] Strongly disagree

6) **Evaluate the speed of the robot**

[ ] Too fast
[ ] Fast
[ ] Somewhat fast
[ ] Neither fast or slow
[ ] Somewhat slow
[ ] Slow
[ ] Too slow

7) **It was fun to walk with a person using this specific method**

[ ] Strongly agree
[ ] Agree
[ ] Somewhat agree
[ ] Neither agree or disagree
[ ] Somewhat disagree
[ ] Disagree
[ ] Strongly disagree

[ ] Manual	**[ ] Autonomous**

Which method would you prefer? (After both methods)

Why?

APPENDIX B

VIBRATION PATTERN ANALYSIS FOR HAPTIC BELTS

In this Chapter, we provide the user study for vibration patterns to be used on the Tactile Belt described in Chapter 6.

B.1 Tactile Belt

The belt is made of elastic material so that the vibration motors make contact with the human body and the angle between motors is fixed. This property allows us to represent the 8 cardinal directions consistently for every user.

Upon receiving a vibration pattern, the controller on the belt applies corresponding sequences of voltages to the vibration motors. A vibration pattern incorporate the activation frequency, duration and rhythm of vibrations for all motors.

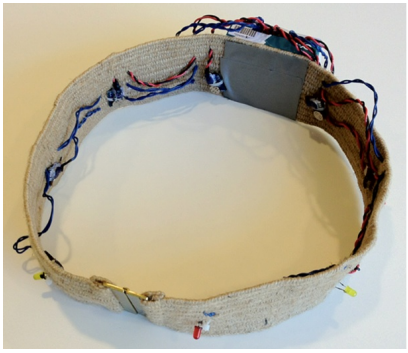


Figure 50: The haptic belt prototype used for guidance

B.1.1 Hardware

8 coin type vibration motors are placed uniformly placed around the belt. Although coin motors provide less strength than cylindrical motors, their size and shape make them ideal for the belt application. The motors have 3V rated voltage, 9000 rpm

rotation, 90 mA rated current and a maximum 50db mechanical noise and they are driven by a single ULN2803A chip with Pulse Width Modulation (PWM) at 20 kHz.

For controlling the belt, we chose an Arduino, which is a relatively cheap open source electronics prototyping platform. The communication between the Controller PC and the belt is achieved by a RS-232 serial port. Arduino Uno has a built-in USB-to-serial converter and the electronics are powered by this USB port.

B.1.2 Software

Our system makes use of the Robot Operating System (ROS) software infrastructure for interprocess communication and message passing. Controller PC sends messages to Arduino on belt, which consists of a bit array for each of the 8 motors. A bit in the array indicates if the motor is going to vibrate or not, during the corresponding time slot of 1/16 seconds. Upon receiving the bit arrays command, Arduino starts executing the bits one by one. Since all the motors are actuated in the main controller loop, the vibrations are synchronous.

B.2 Vibration Patterns

We define two main classes of vibration patterns depending on the type of intended human motion: directional and rotational. Directional patterns induce a motion towards a direction. Rotational patterns induce a rotation around self, which is intended to control the orientation of the human.

- **ONE TAP:** a motor is active for 250ms
- **TWO TAPS:** a motor is active 250ms, inactive for 250ms and active for 250ms again
- **CONT:** a motor is active until a new pattern is applied
- **WAVE:** a feeling that starts from the opposite end of desired direction and ends in desired direction

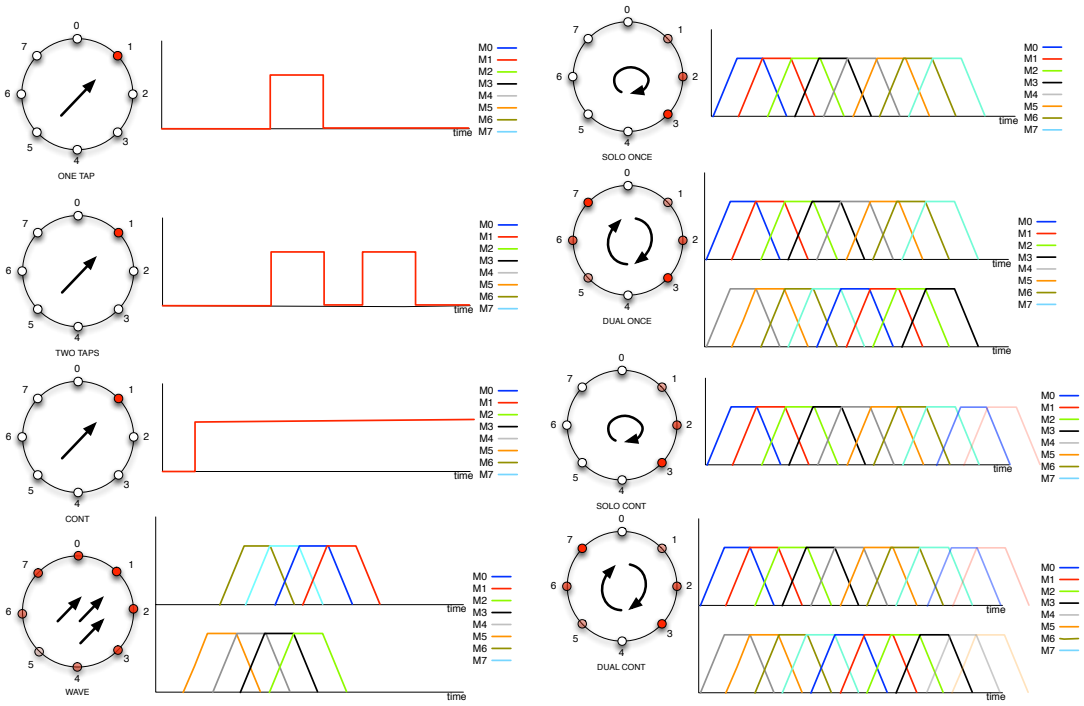


Figure 51: Evaluated vibration patterns. a) Directional b) Rotational

Figure 51(a) illustrates all 4 directional patterns for towards northeast cardinal direction (Vibration Motor 1).

- **SOLO ONCE:** activates all 8 motors consecutively, starting from left motor for clockwise, right for counter-clockwise.
- **SOLO CONT:** repeats **SOLO ONCE** pattern
- **DUAL ONCE:** circle motion is executed for one full circle with two opposing motors instead of one
- **DUAL CONT:** repeats **DUAL ONCE** pattern

Figure 51(b) illustrates all 4 rotational patterns for a clockwise rotation motion.

B.3 Procedure

The subject first went through a training for 5 minutes where experimenter applied directional and rotational patterns and told the correct direction/rotation. The subject is instructed to walk randomly in a confined area so that the patterns are tested while the human is in motion. The subject was asked to press a button on a Xbox controller whenever he/she decides on the direction/rotation and then tell the direction (coded 1-8) or rotation (left or right). The experiments consisted of 4 studies:

1. 5 samples from each of the **ONE TAP, TWO TAPS, WAVE** directional patterns are applied in random order.
2. 8 samples from **CONT** directional pattern are applied.
3. 2 samples from each of the 4 rotational patterns are applied.
4. 2 samples from each of the 4 rotational patterns while a random directional motion pattern is applied in between samples.

Upon the completion of the experiment, the experimenter applied all vibration patterns one by one and asked which directional and rotational pattern the he/she would prefer. Then, a post-study survey is conducted.

We hypothesize that subjects will prefer a one-time signal pattern in Experiment 1 to continuous pattern in Experiment 2 because a long lasting vibration may be annoying to the users. Our second hypothesis is that applying intermediate directional patterns will reduce the recognition rate of rotational patterns. This would be tested by comparing Study Experiment 3 and Experiment 4 results.

B.4 Measures

3 metrics were used for evaluation:

- **Recognition error (RE):** Angle difference between applied and perceived direction
- **Recognition accuracy (RA):** Percentage of correct recognition
- **Reaction time (RT):** Time between the start of the pattern to the instant the subject decides on an answer. A timeout occurs if subject can not give an answer in 5 seconds.

The post-survey consisted of 4 usability questions on a 10-point Likert scale. Participants were also asked for their preferred directional and rotational patterns.

B.5 Results

B.5.1 Directional Patterns

Recognition error and RT w.r.t. pattern type: A total of 344 directional pattern samples were sampled in Experiments 1 and 2. Since the directions are discretized, RE from any single test ranges from 0 to 180° with 45° increments. Results with respect to pattern type is in Table 10.

Table 10: Average recognition error and reaction times of directional patterns

	ONE TAP	TWO TAPS	CONT	WAVE
Directional Error	12.4°	10.6°	8.4°	23.1°
Reaction Time (s)	1.32	1.13	1.26	1.92

With a mean RE of 8.4°, **CONT** pattern was the most accurate directional pattern, followed by **TWO TAPS** pattern with 10.6°. Subjects recognized **TWO TAPS** the fastest by an average of 1.13 seconds. **WAVE** pattern performed significantly worse than others on both measures.

Recognition accuracy w.r.t. applied direction: The confusion matrix for RA of the applied directions is given in Table 52.

Our results show that the recognition accuracy is highly dependent on the applied direction. The subjects recognized the front direction with highest RA (%97) whereas

Applied Direction \ Perceived Direction	1	2	3	4	5	6	7	8
1	0.97	0.03	0	0	0	0	0	0
2	0	0.59	0.38	0	0.03	0	0	0
3	0	0.1	0.55	0.3	0.05	0	0	0
4	0	0.04	0.04	0.76	0.16	0	0	0
5	0	0.02	0	0.09	0.81	0.02	0	0
6	0	0.02	0	0.02	0.04	0.84	0.08	0
7	0.02	0	0	0	0	0.26	0.65	0.06
8	0.03	0	0	0	0	0.03	0.14	0.80

Figure 52: Confusion matrix of recognition accuracy of directional vibration patterns. Our results show that the recognition accuracy is highly dependent on the applied direction. The subjects recognied the front direction (Direction 1) with the highest accuracy (%97) whereas Direction 3 had the least accuracy (%55)

Direction 3 (right) had the least RA (%55).

We expected the directional RA to be symmetrical around the belt, meaning that the accuracy of right and left directions should be equally sensitive to haptic feedback. However, there is at least %10 accuracy difference for all 3 such pairs. This could be due to imperfect alignment of motors in the belt prototype.

B.5.2 Rotational Patterns

Table 11: Average recognition accuracy and reaction times of rotational patterns

	SOLO ONCE	DUAL ONCE	SOLO CONT	DUAL CONT
Recog. Accuracy	%100	%92	%100	%98
Reaction Time (s)	1.32	1.84	1.16	1.68

A total of 256 rotational pattern samples were tested Experiments 1 and 2. Results are in Figure 11.

Subjects recognized **SOLO ONCE** and **SOLO CONT** patterns with %100 accuracy, whereas had **DUAL ONCE** had %92 RA. **SOLO CONT** had the least RT by 1.16 seconds. It is interesting to note that this reaction time is almost identical to the directional pattern with the least RT (1.13s).

B.5.3 Usability of Tactile Belt

Subjects thought that it was fairly easy to move while wearing the belt (M=9.2). Most subjects thought that the belt was comfortable (M=8.3) and it fit their waist well (M=8.4). The vibration motors were not found very silent (M=6.8). Actual questions in the post-study survey results are shown in Figure 53.

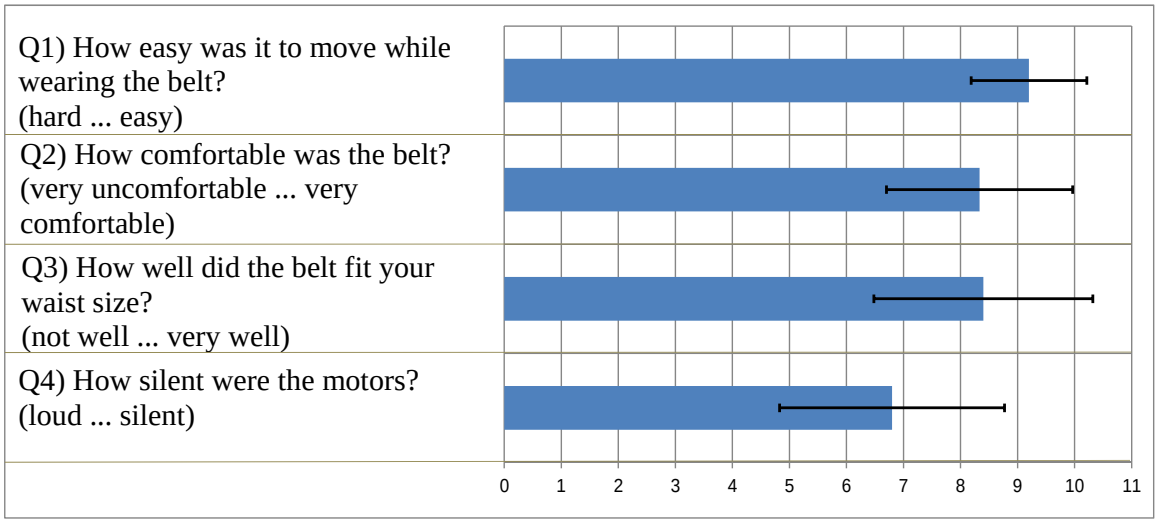


Figure 53: Results of post-study survey.

B.5.4 Discussion

Among directional patterns, **TWO TAPS** had the least RT and was found the most intuitive by our usability study. When asked which directional pattern they would prefer, 10 out of 15 subjects chose **TWO TAPS**, 3 subjects chose **CONT** and 2 subjects selected **ONE TAP** and no one chose **WAVE**. Although **CONT** have less recognition error, most subjects preferred **TWO TAPS** probably because people didn't feel comfortable when the vibration lasted for a long time. This supports our first hypothesis that continuous patterns would not be found preferable.

7 out of 15 subjects preferred **SOLO ONCE**, 7 subjects preferred **SOLO CONT** and 1 subject preferred **DUAL CONT** in the post-study survey. Our results show that subjects rarely made mistakes in recognizing rotations and using a single motor

for rotation patterns is preferable over using two motors. Therefore an application may use either of the **SOLO** patterns.

Our second hypothesis did not hold, as we found that application of other pattern between rotational patterns did not deteriorate the recognition rate.

REFERENCES

[1] Althaus, P., Ishiguro, H., Kanda, T., Miyashita, T., and Christensen, H. I., "Navigation for human-robot interaction tasks," in *Robotics and Automation, 2004 IEEE International Conference on*, vol. 2, pp. 1894–1900, IEEE, 2004.

[2] Aly, A. and Tapus, A., "An integrated model of speech to arm gestures mapping in human-robot interaction," in *Information Control Problems in Manufacturing*, vol. 14, pp. 817–822, 2012.

[3] Arras, K. O., Mozos, O. M., and Burgard, W., "Using boosted features for the detection of people in 2d range data," in *Robotics and Automation, 2007 IEEE International Conference on*, pp. 3402–3407, IEEE, 2007.

[4] Bar-Shalom, Y. and Li, X.-R., *Multitarget-multisensor tracking: principles and techniques*, vol. 19. YBS Storrs, Conn., 1995.

[5] Baumberg, A. and Hogg, D., "Learning deformable models for tracking the human body," in *Motion-Based Recognition*, pp. 39–60, Springer, 1997.

[6] Bellotto, N. and Hu, H., "Multisensor-based human detection and tracking for mobile service robots," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 39, no. 1, pp. 167–181, 2009.

[7] Bennewitz, M., Axenbeck, T., Behnke, S., and Burgard, W., "Robust recognition of complex gestures for natural human-robot interaction," in *Proc. of the Workshop on Interactive Robot Learning at Robotics: Science and Systems Conference (RSS)*, 2008.

[8] Bennewitz, M., Burgard, W., Cielniak, G., and Thrun, S., "Learning motion patterns of people for compliant robot motion," *The International Journal of Robotics Research*, vol. 24, no. 1, pp. 31–48, 2005.

[9] Bernardin, K. and Stiefelhagen, R., "Audio-visual multi-person tracking and identification for smart environments," in *Proceedings of the 15th international conference on Multimedia*, pp. 661–670, ACM, 2007.

[10] Blodow, N., Marton, Z.-C., Pangercic, D., Rühr, T., Tenorth, M., and Beetz, M., "Inferring generalized pick-and-place tasks from pointing gestures," in *IEEE International Conference on Robotics and Automation (ICRA), Workshop on Semantic Perception, Mapping and Exploration*, 2011.

[11] Brooks, A. G. and Breazeal, C., "Working with robots and objects: Revisiting deictic reference for achieving spatial common ground," in *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pp. 297–304, ACM, 2006.

[12] Burgard, W., Cremers, A. B., Fox, D., Hähnel, D., Lakemeyer, G., Schulz, D., Steiner, W., and Thrun, S., "The interactive museum tour-guide robot," in *AAAI/IAAI*, pp. 11–18, 1998.

[13] Buys, K., Cagniart, C., Baksheev, A., De Laet, T., De Schutter, J., and Pantofaru, C., "An adaptable system for rgb-d based human body detection and pose estimation," *Journal of Visual Communication and Image Representation*, 2013.

[14] Cakmak, M., Srinivasa, S. S., Lee, M. K., Kiesler, S., and Forlizzi, J., "Using spatial and temporal contrast for fluent robot-human hand-overs," in *Proceedings of the 6th international conference on Human-robot interaction*, pp. 489–496, ACM, 2011.

[15] Carballo, A., Ohya, A., and Yuta, S., "Fusion of double layered multiple laser range finders for people detection from a mobile robot," in *Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on*, pp. 677–682, IEEE, 2008.

[16] Cheng, K. and Takatsuka, M., "Hand pointing accuracy for vision-based interactive systems," in *Human-Computer Interaction–INTERACT 2009*, pp. 13–16, Springer, 2009.

[17] Chung, T. H., Hollinger, G. A., and Isler, V., "Search and pursuit-evasion in mobile robotics," *Autonomous robots*, vol. 31, no. 4, pp. 299–316, 2011.

[18] Cipolla, R. and Hollinghurst, N. J., "Human-robot interface by pointing with uncalibrated stereo vision," *Image and Vision Computing*, vol. 14, no. 3, pp. 171–178, 1996.

[19] Clark, H. and Brennan, S., "Grounding in communication," *Perspectives on socially shared cognition*, vol. 13, no. 1991, pp. 127–149, 1991.

[20] Clodic, A., Fleury, S., Alami, R., Chatila, R., Bailly, G., Brethes, L., Cottret, M., Danes, P., Dollat, X., Elisei, F., and others, "Rackham: An interactive robot-guide," in *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, pp. 502–509, IEEE, 2006.

[21] Cosgun, A., Florencio, D. A., and Christensen, H. I., "Autonomous person following for telepresence robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4335–4342, IEEE, 2013.

[22] Diosi, A., Taylor, G., and Kleeman, L., "Interactive slam using laser and advanced sonar," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 1103–1108, IEEE, 2005.

[23] Dragan, A. D., Lee, K. C., and Srinivasa, S. S., "Legibility and predictability of robot motion," in *Human-Robot Interaction (HRI), 2013 8th ACM/IEEE International Conference on*, pp. 301–308, IEEE, 2013.

[24] Droeschel, D., Stuckler, J., and Behnke, S., "Learning to interpret pointing gestures with a time-of-flight camera," in *Human-Robot Interaction (HRI), 2011 6th ACM/IEEE International Conference on*, pp. 481–488, IEEE, 2011.

[25] Droeschel, D., Stuckler, J., Holz, D., and Behnke, S., "Towards joint attention for a domestic service robot-person awareness and gesture recognition using time-of-flight cameras," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1205–1210, IEEE, 2011.

[26] Endsley, M. R. and Garland, D. J., *Situation awareness analysis and measurement*. CRC Press, 2000.

[27] Fitzgibbon, A., Pilu, M., and Fisher, R. B., "Direct least square fitting of ellipses," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 21, no. 5, pp. 476–480, 1999.

[28] Fox, D., Burgard, W., Dellaert, F., and Thrun, S., "Monte carlo localization: Efficient position estimation for mobile robots," *AAAI/IAAI*, vol. 1999, pp. 343–349, 1999.

[29] Fox, D., Burgard, W., and Thrun, S., "The dynamic window approach to collision avoidance," *Robotics & Automation Magazine, IEEE*, vol. 4, no. 1, pp. 23–33, 1997.

[30] Ganapathi, V., Plagemann, C., Koller, D., and Thrun, S., "Real time motion capture using a single time-of-flight camera," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 755–762, IEEE, 2010.

[31] Garcia-Salicetti, S., Beumier, C., Chollet, G., Dorizzi, B., les Jardins, J. L., Lunter, J., Ni, Y., and Petrovska-Delacrétaz, D., "Biomet: a multimodal person authentication database including face, voice, fingerprint, hand and signature modalities," in *Audio-and Video-Based Biometric Person Authentication*, pp. 845–853, Springer, 2003.

[32] Garrell, A. and Sanfeliu, A., "Local optimization of cooperative robot movements for guiding and regrouping people in a guiding mission," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 3294–3299, IEEE, 2010.

[33] Germa, T., Lerasle, F., Ouadah, N., and Cadenat, V., "Vision and rfid data fusion for tracking people in crowds by a mobile robot," *Computer Vision and Image Understanding*, vol. 114, no. 6, pp. 641–651, 2010.

[34] Glas, D. F., Miyashita, T., Ishiguro, H., and Hagita, N., "Laser-based tracking of human position and orientation using parametric shape modeling," *Advanced robotics*, vol. 23, no. 4, pp. 405–428, 2009.

[35] Gockley, R., "Developing spatial skills for social robots.," in *AAAI Spring Symposium: Multidisciplinary Collaboration for Socially Assistive Robotics*, pp. 15–17, 2007.

[36] Gockley, R., Forlizzi, J., and Simmons, R., "Natural person-following behavior for social robots," in *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pp. 17–24, ACM, 2007.

[37] Granata, C. and Bidaud, P., "A framework for the design of person following behaviors for social mobile robots," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 4652–4659, IEEE, 2012.

[38] Grisetti, G., Stachniss, C., and Burgard, W., "Improved techniques for grid mapping with rao-blackwellized particle filters," *Robotics, IEEE Transactions on*, vol. 23, no. 1, pp. 34–46, 2007.

[39] Hall, E. T., *The hidden dimension*. Anchor Books, 1966.

[40] Hall, E. T., "The hidden dimension .," 1966.

[41] Harris, C. and Stephens, M., "A combined corner and edge detector," in *Alvey vision conference*, vol. 15, p. 50, Manchester, UK, 1988.

[42] Hato, Y., Satake, S., Kanda, T., Imai, M., and Hagita, N., "Pointing to space: modeling of deictic interaction referring to regions," in *Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction*, pp. 301–308, IEEE Press, 2010.

[43] Helbing, D. and Molnar, P., "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[44] Hicheur, H., Vieilledent, S., Richardson, M., Flash, T., and Berthoz, A., "Velocity and curvature in human locomotion along complex curved paths: a comparison with hand movements," *Experimental brain research*, vol. 162, no. 2, pp. 145–154, 2005.

[45] Hoeller, F., Schulz, D., Moors, M., and Schneider, F. E., "Accompanying persons with a mobile robot using motion prediction and probabilistic roadmaps," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pp. 1260–1265, IEEE, 2007.

[46] Hosoya, E., Sato, H., Kitabata, M., Harada, I., Nojima, H., and Onozawa, A., "Arm-pointer: 3d pointing interface for real-world interaction," in *Computer Vision in Human-Computer Interaction*, pp. 72–82, Springer, 2004.

[47] Hu, K., Canavan, S., and Yin, L., "Hand pointing estimation for human computer interaction based on two orthogonal-views," in *Pattern Recognition (ICPR), 2010 20th International Conference on*, pp. 3760–3763, IEEE, 2010.

[48] Jojic, N., Brumitt, B., Meyers, B., Harris, S., and Huang, T., "Detection and estimation of pointing gestures in dense disparity maps," in *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pp. 468–475, IEEE, 2000.

[49] Kahn, R. E. and Swain, M. J., "Understanding people pointing: The perseus system," in *Computer Vision, 1995. Proceedings., International Symposium on*, pp. 569–574, IEEE, 1995.

[50] Kalman, R. E., "A new approach to linear filtering and prediction problems," *Journal of Fluids Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[51] Kehl, R. and Van Gool, L., "Real-time pointing gesture recognition for an immersive environment," in *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, pp. 577–582, IEEE, 2004.

[52] Kemp, C. C., Anderson, C. D., Nguyen, H., Trevor, A. J., and Xu, Z., "A point-and-click interface for the real world: laser designation of objects for mobile manipulation," in *Human-Robot Interaction (HRI), 2008 3rd ACM/IEEE International Conference on*, pp. 241–248, IEEE, 2008.

[53] Kendon, A., *Conducting interaction: Patterns of behavior in focused encounters*, vol. 7. CUP Archive, 1990.

[54] Khan, Z., Balch, T., and Dellaert, F., "An mcmc-based particle filter for tracking multiple interacting targets," in *Computer Vision-ECCV 2004*, pp. 279–290, Springer, 2004.

[55] Khatib, O., "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[56] Kim, Y.-D., Kim, Y.-G., Lee, S. H., Kang, J. H., and An, J., "Portable fire evacuation guide robot system," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pp. 2789–2794, IEEE, 2009.

[57] Kinnunen, T. and Li, H., "An overview of text-independent speaker recognition: From features to supervectors," *Speech communication*, vol. 52, no. 1, pp. 12–40, 2010.

[58] Kirby, R., Simmons, R., and Forlizzi, J., "Companion: A constraint-optimizing method for person-acceptable navigation," in *Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on*, pp. 607–612, IEEE, 2009.

[59] Kirchner, N., Alempijevic, A., and Virgona, A., "Head-to-shoulder signature for person recognition," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 1226–1231, IEEE, 2012.

[60] Kleinehagenbrock, M., Lang, S., Fritsch, J., Lomker, F., Fink, G. A., and Sagerer, G., "Person tracking with a mobile robot based on multi-modal anchoring," in *Robot and Human Interactive Communication, 2002. Proceedings. 11th IEEE International Workshop on*, pp. 423–429, IEEE, 2002.

[61] Kollar, T., Tellex, S., Roy, D., and Roy, N., "Toward understanding natural language directions," in *Human-Robot Interaction (HRI), 2010 5th ACM/IEEE International Conference on*, pp. 259–266, IEEE, 2010.

[62] Koren, Y. and Borenstein, J., "Potential field methods and their inherent limitations for mobile robot navigation," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1398–1404, IEEE, 1991.

[63] Kowadlo, G., Ye, P., and Zukerman, I., "Influence of gestural salience on the interpretation of spoken requests.," in *INTERSPEECH*, pp. 2034–2037, 2010.

[64] Kruse, T., Basili, P., Glasauer, S., and Kirsch, A., "Legible robot navigation in the proximity of moving humans," in *Advanced Robotics and its Social Impacts (ARSO), 2012 IEEE Workshop on*, pp. 83–88, IEEE, 2012.

[65] Kruse, T., Kirsch, A., Sisbot, E. A., and Alami, R., "Exploiting human cooperation in human-centered robot navigation," in *RO-MAN, 2010 IEEE*, pp. 192–197, IEEE, 2010.

[66] Kuderer, M., Kretzschmar, H., and Burgard, W., "Teaching mobile robots to cooperatively navigate in populated environments," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pp. 3138–3143, IEEE, 2013.

[67] Kuipers, B., "The spatial semantic hierarchy," *Artificial intelligence*, vol. 119, no. 1, pp. 191–233, 2000.

[68] Lee, M. K. and Takayama, L., "Now, i have a body: Uses and social norms for mobile remote presence in the workplace," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 33–42, ACM, 2011.

[69] Leibe, B., Seemann, E., and Schiele, B., "Pedestrian detection in crowded scenes," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 878–885, IEEE, 2005.

[70] Li, Z., Hofemann, N., Fritsch, J., and Sagerer, G., "Hierarchical modeling and recognition of manipulative gesture," in *Proc. of the Workshop on Modeling People and Human Interaction at the IEEE Int. Conf. on Computer Vision*, vol. 77, 2005.

[71] Lichtenthäler, C. and Kirsch, A., "Towards Legible Robot Navigation - How to Increase the Intend Expressiveness of Robot Navigation Behavior," in *International Conference on Social Robotics - Workshop Embodied Communication of Goals and Intentions*, 2013.

[72] Loper, M. M., Koenig, N. P., Chernova, S. H., Jones, C. V., and Jenkins, O. C., "Mobile human-robot teaming with environmental tolerance," in *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pp. 157–164, ACM, 2009.

[73] Lu, D. V. and Smart, W. D., "Towards more efficient navigation for robots and humans," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1707–1713, IEEE, 2013.

[74] Luber, M., Spinello, L., Silva, J., and Arras, K. O., "Socially-aware robot navigation: A learning approach," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 902–907, IEEE, 2012.

[75] Martin, C., Böhme, H.-J., and Gross, H.-M., "Conception and realization of a multi-sensory interactive mobile office guide," in *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, vol. 6, pp. 5368–5373, IEEE, 2004.

[76] Martin, C., Steege, F.-F., and Gross, H.-M., "Estimation of pointing poses for visually instructing mobile robots under real world conditions," *Robotics and Autonomous Systems*, vol. 58, no. 2, pp. 174–185, 2010.

[77] Martínez-García, E., Akihisa, O., Yuta, S., and others, "Crowding and guiding groups of humans by teams of mobile robots," in *Advanced Robotics and its Social Impacts, 2005. IEEE Workshop on*, pp. 91–96, IEEE, 2005.

[78] Matikainen, P., Pillai, P., Mummert, L., Sukthankar, R., and Hebert, M., "Prop-free pointing detection in dynamic cluttered environments," in *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pp. 374–381, IEEE, 2011.

[79] Matuszek, C., Bo, L., Zettlemoyer, L., and Fox, D., "Learning from unscripted deictic gesture and language for human-robot interactions," 2014.

[80] Mitzel, D. and Leibe, B., "Close-range human detection for head-mounted cameras," in *British Machine Vision Conference (BMVC)*, 2012.

[81] Miura, J., Satake, J., Chiba, M., Ishikawa, Y., Kitajima, K., and Masuzawa, H., "Development of a person following robot and its experimental evaluation," in *Proceedings of the 11th International Conference on Intelligent Autonomous Systems, Ottawa, Canada*, pp. 89–98, 2010.

[82] Moeslund, T. B. and Granum, E., "A survey of computer vision-based human motion capture," *Computer Vision and Image Understanding*, vol. 81, no. 3, pp. 231–268, 2001.

[83] Montemerlo, M., Thrun, S., and Whittaker, W., "Conditional particle filters for simultaneous mobile robot localization and people-tracking," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 1, pp. 695–701, IEEE, 2002.

[84] Mozos, O. M., Triebel, R., Jensfelt, P., Rottmann, A., and Burgard, W., "Supervised semantic labeling of places using information extracted from sensor data," *Robotics and Autonomous Systems*, vol. 55, no. 5, pp. 391–402, 2007.

[85] Müller, J., Stachniss, C., Arras, K., and Burgard, W., "Socially inspired motion planning for mobile robots in populated environments," in *Proc. of International Conference on Cognitive Systems*, 2008.

[86] Munsell, B. C., Temlyakov, A., Qu, C., and Wang, S., "Person identification using full-body motion and anthropometric biometrics from kinect videos," in *Computer Vision–ECCV 2012. Workshops and Demonstrations*, pp. 91–100, Springer, 2012.

[87] Murakami, R., Morales Saiki, L. Y., Satake, S., Kanda, T., and Ishiguro, H., "Destination unknown: walking side-by-side without knowing the goal," in *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*, pp. 471–478, ACM, 2014.

[88] Nickel, K. and Stiefelhagen, R., "Pointing gesture recognition based on 3d-tracking of face, hands and head orientation," in *Proceedings of the 5th international conference on Multimodal interfaces*, pp. 140–146, ACM, 2003.

[89] Nourbakhsh, I. R., Kunz, C., and Willeke, T., "The mobot museum robot installations: A five year experiment," in *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 4, pp. 3636–3641, IEEE, 2003.

[90] Ohki, T., Nagatani, K., and Yoshida, K., "Collision avoidance method for mobile robot considering motion and personal spaces of evacuees," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),*, pp. 1819–1824, IEEE, 2010.

[91] Ohya, A. and Munekata, T., "Intelligent escort robot moving together with human-interaction in accompanying behavior," in *Proceedings 2002 FIRA Robot World Congress*, pp. 31–35, 2002.

[92] Ota, M., Ogitsu, T., Hisahara, H., Takemura, H., Ishii, Y., and Mizoguchi, H., "Recovery function for human following robot losing target," in *Industrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE*, pp. 4253–4257, IEEE, 2013.

[93] Pacchierotti, E., Christensen, H., Jensfelt, P., and others, "Design of an office-guide robot for social interaction studies," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pp. 4965–4970, IEEE, 2006.

[94] Pacchierotti, E., Christensen, H. I., and Jensfelt, P., "Human-robot embodied interaction in hallway settings: a pilot user study," in *Robot and Human Interactive Communication, 2005. ROMAN 2005. IEEE International Workshop on*, pp. 164–171, IEEE, 2005.

[95] Pandey, A. K. and Alami, R., "A step towards a sociable robot guide which monitors and adapts to the person's activities," in *Advanced Robotics, 2009. ICAR 2009. International Conference on*, pp. 1–8, IEEE, 2009.

[96] Park, J. J. and Kuipers, B., "Autonomous person pacing and following with model predictive equilibrium point control," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 1060–1067, IEEE, 2013.

[97] Pellegrini, S., Ess, A., Schindler, K., and Van Gool, L., "You'll never walk alone: Modeling social behavior for multi-target tracking," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 261–268, IEEE, 2009.

[98] Phillips, P. J., Flynn, P. J., Scruggs, T., Bowyer, K. W., Chang, J., Hoffman, K., Marques, J., Min, J., and Worek, W., "Overview of the face recognition grand challenge," in *Computer vision and pattern recognition, 2005. CVPR 2005. IEEE computer society conference on*, vol. 1, pp. 947–954, IEEE, 2005.

[99] Prassler, E., Bank, D., and Kluge, B., "Motion coordination between a human and a robotic wheelchair," in *Robot and Human Interactive Communication, 2001. Proceedings. 10th IEEE International Workshop on*, pp. 412–417, IEEE, 2001.

[100] Quintero, C. P., Fomena, R. T., Shademan, A., Wolleb, N., Dick, T., and Jagersand, M., "Sepo: Selecting by pointing as an intuitive human-robot command interface," in *IEEE Int. Conference of Robotics and Automation, Karslruhe, Germany*, 2013.

[101] Raza Abidi, S. S., Williams, M., and Johnston, B., "Human pointing as a robot directive," in *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*, pp. 67–68, IEEE Press, 2013.

[102] Rios-Martinez, J., Spalanzani, A., and Laugier, C., "Understanding human interaction for probabilistic autonomous navigation using risk-rrt approach," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp. 2014–2019, IEEE, 2011.

[103] Robinette, P. and Howard, A. M., "Incorporating a model of human panic behavior for robotic-based emergency evacuation," in *RO-MAN, 2011 IEEE*, pp. 47–52, IEEE, 2011.

[104] Rogers, J. G., "Life-long mapping of objects and places in domestic environments," 2013.

[105] Rubner, Y., Tomasi, C., and Guibas, L. J., "A metric for distributions with applications to image databases," in *Computer Vision, 1998. Sixth International Conference on*, pp. 59–66, IEEE, 1998.

[106] Rusu, R. B. and Cousins, S., "3d is here: Point cloud library (pcl)," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1–4, IEEE, 2011.

[107] Sasaki, T. and Hashimoto, H., "Human observation based mobile robot navigation in intelligent space," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pp. 1044–1049, IEEE, 2006.

[108] Satake, S., Kanda, T., Glas, D. F., Imai, M., Ishiguro, H., and Hagita, N., "How to approach humans?-strategies for social robots to initiate interaction," in *Human-Robot Interaction (HRI), 2009 4th ACM/IEEE International Conference on*, pp. 109–116, IEEE, 2009.

[109] Scandolo, L. and Fraichard, T., "An anthropomorphic navigation scheme for dynamic scenarios," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 809–814, IEEE, 2011.

[110] Schmidt, J., Hofemann, N., Haasch, A., Fritsch, J., and Sagerer, G., "Interacting with a mobile robot: Evaluating gestural object references," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pp. 3804–3809, IEEE, 2008.

[111] Schulz, D., Burgard, W., Fox, D., and Cremers, A. B., "Tracking multiple moving targets with a mobile robot using particle filters and statistical data association," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 2, pp. 1665–1670, IEEE, 2001.

[112] Setti, F., Russell, C., Bassetti, C., and Cristani, M., "F-formation detection: Individuating free-standing conversational groups in images," *PloS one*, vol. 10, no. 5, 2015.

[113] Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A., "Real-time human pose recognition in parts from single depth images," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1297–1304, IEEE, 2011.

[114] Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., Cook, M., and Moore, R., "Real-time human pose recognition in parts from single depth images," *Communications of the ACM*, vol. 56, no. 1, pp. 116–124, 2013.

[115] Sidenbladh, H., Black, M. J., and Fleet, D. J., "Stochastic tracking of 3d human figures using 2d image motion," in *ECCV*, pp. 702–718, Springer, 2000.

[116] Sidenbladh, H., Kragic, D., and Christensen, H. I., "A person following behaviour for a mobile robot," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 1, pp. 670–675, IEEE, 1999.

[117] Siegwart, R., Arras, K. O., Bouabdallah, S., Burnier, D., Froidevaux, G., Greppin, X., Jensen, B., Lorotte, A., Mayor, L., Meisser, M., and others, "Robox at expo. 02: A large-scale installation of personal robots," *Robotics and Autonomous Systems*, vol. 42, no. 3, pp. 203–222, 2003.

[118] Sirovich, L. and Kirby, M., "Low-dimensional procedure for the characterization of human faces," *JOSA A*, vol. 4, no. 3, pp. 519–524, 1987.

[119] Sisbot, E. A., Marin-Urias, L. F., Alami, R., and Simeon, T., "A human aware mobile robot motion planner," *Robotics, IEEE Transactions on*, vol. 23, no. 5, pp. 874–883, 2007.

[120] Spinello, L., Arras, K. O., Triebel, R., and Siegwart, R., "A layered approach to people detection in 3d range data," in *AAAI Conf. on Artif. Intell.(AAAI)*, 2010.

[121] Stein, P., Spalanzani, A., Santos, V., Laugier, C., and others, "Robot navigation taking advantage of moving agents," in *IROS Workshop on Assistance and Service robotics in a human environment*, 2012.

[122] Stein, P. S., Santos, V., Spalanzani, A., Laugier, C., and others, "Navigating in populated environments by following a leader," in *Ro-man 2013-International Symposium on Robot and Human Interactive Communication*, 2013.

[123] Svenstrup, M., Bak, T., and Andersen, H. J., "Trajectory planning for robots in dynamic human environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4293–4298, IEEE, 2010.

[124] Takayama, L., Marder-Eppstein, E., Harris, H., and Beer, J., "Assisted driving of a mobile remote presence system: System design and controlled user evaluation," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2011.

[125] Thrun, S., Bennewitz, M., Burgard, W., Cremers, A. B., Dellaert, F., Fox, D., Hahnel, D., Rosenberg, C., Roy, N., Schulte, J., and others, "Minerva: A second-generation museum tour-guide robot," in *Robotics and automation, 1999. Proceedings. 1999 IEEE international conference on*, vol. 3, IEEE, 1999.

[126] Topp, E., Christensen, H., and others, "Topological modelling for human augmented mapping," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pp. 2257–2263, IEEE, 2006.

[127] Topp, E. A. and Christensen, H. I., "Tracking for following and passing persons," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pp. 2321–2327, IEEE, 2005.

[128] Topp, E. A. and Christensen, H. I., "Detecting region transitions for human-augmented mapping," *Robotics, IEEE Transactions on*, vol. 26, no. 4, pp. 715–720, 2010.

[129] Topp, E. A., *Human-robot interaction and mapping with a service robot: Human augmented mapping*. PhD thesis, Swedish School of Sport and Health Sciences, GIH, 2008.

[130] Trautman, P. and Krause, A., "Unfreezing the robot: Navigation in dense, interacting crowds," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 797–803, IEEE, 2010.

[131] Trevor, A. J. B., Gedikli, S., Rusu, R., and Christensen, H. I., "Efficient organized point cloud segmentation with connected components," in *ICRA Workshop on Semantic Perception Mapping and Exploration*, 2013.

[132] Trevor, A. J., "Semantic mapping for service robots: Building and using maps for mobile manipulators in semi-structured environments," 2015.

[133] Trevor, A. J., Rogers III, J. G., Christensen, H., and others, "Planar surface slam with 3d and 2d sensors," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 3041–3048, IEEE, 2012.

[134] Turk, M. A. and Pentland, A. P., "Face recognition using eigenfaces," in *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, pp. 586–591, IEEE, 1991.

[135] Tuzel, O., Porikli, F., and Meer, P., "Human detection via classification on riemannian manifolds," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pp. 1–8, IEEE, 2007.

[136] Vasquez, D., Stein, P., Rios-Martinez, J., Escobedo, A., Spalanzani, A., Laugier, C., and others, "Human aware navigation for assistive robotics," in *ISER-13th International Symposium on Experimental Robotics-2012*, 2012.

[137] Viola, P. and Jones, M. J., "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[138] Wainer, J., Feil-Seifer, D. J., Shell, D. A., and Mataric, M. J., "The role of physical embodiment in human-robot interaction," in *15th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 117–122, IEEE, 2006.

[139] Wilson, A. D. and Bobick, A. F., "Parametric hidden markov models for gesture recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 21, no. 9, pp. 884–900, 1999.

[140] Xavier, J., Pacheco, M., Castro, D., Ruano, A., and Nunes, U., "Fast line, arc/circle and leg detection from laser scan data in a player driver," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 3930–3935, IEEE, 2005.

[141] Yan, P. and Bowyer, K. W., "Biometric recognition using 3d ear shape," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 8, pp. 1297–1308, 2007.

[142] Yuan, F., Hanheide, M., Sagerer, G., and others, "Spatial context-aware person-following for a domestic robot," 2008.

[143] Zender, H., Jensfelt, P., and Kruijff, G.-J. M., "Human-and situation-aware people following," in *Robot and Human interactive Communication, 2007. RO-MAN 2007. The 16th IEEE International Symposium on*, pp. 1131–1136, IEEE, 2007.

[144] Zender, H., Jensfelt, P., Mozos, Ó. M., Kruijff, G.-J. M., and Burgard, W., "An integrated robotic system for spatial understanding and situated interaction in indoor environments," in *AAAI*, vol. 7, pp. 1584–1589, 2007.

[145] Zhang, C. and Zhang, Z., "A survey of recent advances in face detection," tech. rep., Tech. rep., Microsoft Research, 2010.

[146] Zhao, W., Krishnaswamy, A., Chellappa, R., Swets, D. L., and Weng, J., "Discriminant analysis of principal components for face recognition," in *Face Recognition*, pp. 73–85, Springer, 1998.

[147] Zukerman, I., Kowadlo, G., and Ye, P., "Interpreting pointing gestures and spoken requests: a probabilistic, salience-based approach," in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pp. 1558–1566, Association for Computational Linguistics, 2010.

[148] Zukerman, I., Mani, A., Li, Z., and Jarvi, R., "Speaking and pointing-from simulations to the laborator," *Knowledge and Reasoning in Practical Dialogue Systems*, p. 58, 2011.