Socially-Aware Navigation

Guiding Behavior with Pepper

Ronald Cumbal

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

Socially-Aware Navigation
Guiding Behavior with Pepper

Master of Science Thesis

For the degree of Master of Science in Mechanical Engineering - Control Engineering at Delft University of Technology

Ronald Cumbal

October 2, 2017

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Abstract

Socially-aware robot navigation strives to find efficient methods to autonomously navigate known environments while incorporating social metrics derived from human behavior. Several methods built for navigation on dynamic and uncertain environments have been adapted to resemble human navigation, but fail to appropriately reflect the social characteristics of human decision making.

A reliable solution for this problem is found in model-based methods working as local or global planners. Model-based research on human aware navigation focuses on two specific alternatives. The first option corresponds to models that apply social-psychology and cognitive sciences to create human-like behavior, where the *Social Force Model* is the predominant approach. The second alternative is related to models that use machine learning to copy human-like characteristics into mathematical models.

The former approach has proven to be efficient in human-aware navigation, but further studies are required to analyze the expansion of this method to more complex human-interactive navigation. In this thesis project, we look to test the *Modified Extended Social Force Model* (MESFM) to implement a guiding behavior on a humanoid robot. The MESFM incorporates a new force linked to the guided person to maintain a natural distance between both subjects, while generating smooth navigational maneuvers.

In addition, whether caused by sensor failure or occlusion incidents, the event of losing track of the guided person has a good probability of occurring. This scenario is studied by extending the high level control of the robotic system with a *searching* mode.

The architecture proposed to control the guiding behavior and the *searching* mode exploits the design of the humanoid robot Pepper, from SoftBank Robotics, to incorporate human-like gestures and increase the interaction between robot and human.

We aim to apply this behavior as an Office-Guide Robot and test this solution to understand the reaction of people involved in the guiding task though subjective and objective metrics. Our system is developed on top of the open-software Robot Operative System (ROS) with features extracted from Pepper's Naoqi framework.

Table of Contents

List of Figures

List of Tables

Preface

The idea of working in human-focused robotics has always been a keen objective in my professional career. As many robotics enthusiasts, I share the idea that robots will play a key role in human life in the nearby future. Starting this project was, therefore, an easy choice to make; particularly considering the necessity to correctly understand the concepts behind *human-robot interaction* and *motion planning*. This specific contribution between different fields of research was by-far the most inspiring characteristic of the project. Achieving a socially-aware planner that would be applied in a state-of-the-art humanoid robot felt as the appropriate challenge to test my engineering skills with a brief introduction to scientific research.

Acknowledgements

I would like to express my gratitude to my supervisors Dr.ir. K.V. Hindriks and Dr.ir. J. Alonso-Mora for their guidance and assistance during the development of this project. Their excellent understanding of their respective fields made this endeavor easier to manage.

I would also like to show gratitude to Dr. Joachim de Greeff for his always friendly aid in Softbank-Robotics Systems and Pepper-related application development.

It is also important to recognize the support of Ruud de Jong who provided me with the facilities and technical assistance through various stages of the implementation of this project.

Delft, University of Technology
October 2, 2017

Ronald Cumbal

Por las inalcanzables ganas de luchar, todo mi esfuerzo y aliento se lo dedico a mi abuelo Juan. Hasta pronto Cholito.

I am completely certain that all accomplishments I've reached in life have been a result of the strength my family provides. This work, without a doubt, is dedicated to my Mom, Dad, Sister and Aunt. My gratitude and love for you sees no end.

And for their awkward, happy and incomprehensible support, my all time friends Damian, Daniel, Javier, and my newly found buddies Sherin, Gokul and Linda, this thesis work is dedicated to you as well. *I get by with little help from my friends ...*

Chapter 1

Introduction

1-1 Context

The problem of developing socially compliant robots is based on the idea that many human behaviors are generated automatically by human beings. These sort of practices are normally defined inside *social conventions*. In these scenarios, information is rarely exhibited between participants, yet decisions are easily produced on simple assumptions. The scarce amount of detectable information in these circumstances yields the greater difficulty in human robot interaction. Furthermore, among joint-tasks, i.e. actions that a robot performs with the cooperation of a person, the problem of achieving human-like characteristics augments the complexity of the behavior.

The field of *socially-aware navigation* studies these concepts in order to achieve socially acceptable behaviors while a robot navigates in a human populated areas. The objective from this point of view seems quite clear: the presence of a robot should not generate awkward reactions from nearby pedestrians. Navigating autonomously in a human-like manner presents complicated challenges already, but what would happen if we include people inside the control loop? A task like guiding a person toward different locations renders the necessity to generate intuitive actions from the robot that reflect the interactive approach of the activity. In the end, the person should find these actions to be *natural*, *safe* and *comforting*, to mention just a few of these expectations. Research on guiding behaviors has brought some options to achieve this objective, but most alternatives aim to solve particularities of the general problem, leaving the application of a general solution still open for further research.

1-2 Motivation

The project described in this report tries to solve this issue by extending solutions presented in previous works and apply them to the humanoid robot *Pepper* in a guiding task. Fortunately, Pepper was specifically designed for human-robot interaction, which provides versatility in the features that can be incorporated on the complexity of a guiding behavior. The challenge, nonetheless, remains on the implementation of a method that accurately combines the qualities of the interactive design with the efficiency of a motion planning algorithm. The former problem is bound to be complicated as the interactive design is optimal in stationary interactions, and hardware specifications could limit the resources required to implement a motion planner that performs optimally. This last remark is related to relative low cost of Pepper in comparison to other mobile robots. This characteristic, nevertheless, contributes to the idea that state-of-the-art algorithms can be presented to a wider range of public, if robotic platforms are made more accessible to consumers. Consequently no hardware modification will be introduced to Pepper to comply with this idea.

1-3 Literature Review

Initial attempts to conceive interactive guiding robots were proposed by Burgard *et al.* [2], Thrun *et al.*, [3] and Nourbakhsh *et al.* [4] in museum guiding experiments. These projects focused on the efficiency of navigation, but were not deeply involved in the social aspect of navigation. Recent works have aim to study different qualities of a guiding behavior. These works are briefly described next.

People-Aware Navigation for Goal-Oriented Behavior Involving a Human Partner Feil-seifer *et al.* [5] adapts a learning based model to slow down or stop when the human partner is not following. This approach uses a Gaussian Mixture Model (GMM) based on a 4-dimensional feature vector composed by normalized time spent on the task, normalized distance between the robot to the goal, normalized distance of the partner to the goal and the distance between the robot and partner. The objective of this research was limited to investigate the distance between the guided person and the robot, leaving other characteristics of social navigation out of the scope. This constraint limits the application of this method as more information is required to achieve collision avoidance and active interaction between subjects.

Design of an Office-Guide Robot for Social Interaction Studies Pacchierotti *et al.* [6] worked on a guiding robot for an office hallway. Their approach consisted in a multi-module path planner that focused specifically on accompanying a guest, passing-people maneuver and a park-and-wait behavior as a last resort when a safe trajectory is unfeasible. A collision avoidance module (Nearness Diagram) is integrated to guarantee safe navigation with dynamic obstacles. The passing behavior is simply determined to move as far to the right as is possible for the robot.

Once again, the objective of this research was set to study Human Robot Interaction without optimizing the motion planning algorithm and including the input from the guiding subject in the trajectory generation. Moreover, the park-and-wait maneuver deviates from the expected behavior of a human being.

An Adaptive and Proactive Human-Aware Robot Guide In [7], the authors propose a method that constantly monitors its user to adapt to his/her behaviors and be ready to proactively help its partner. A Situation Assessment component gathers spatial information to make decisions, while a planning framework is implemented based on hierarchical MOMPDs (Mixed Observability Markov Decision Processes) and a Supervision System is used to control all modules. Proactive change of speed is achieved with the Speed Adaptation MOMDP module, which will try to influence its speed on the guided person. In order to suspend the behavior, the authors use Activity Areas and time of deviation to define how long the robot should wait for the person. Finally, a option to re-engage the user in the guidance is achieved by giving interesting information to the user.

This paper presents a novel concept to adapt the behavior of the guided person to the robot in a proactive manner, but the greater disadvantage in the implementation of this approach is the constant monitoring of the partner, a characteristic that is quite limited within the sensor capacity of Pepper; specially considering the *back-front* formation while guiding.

Cooperative Social Robots to Accompany Groups of People Garrell *et al.* [8] elaborated the problem of collaboratively controlling a group of robots that guide a group of people. The main contribution of this work is the *Prediction and Anticipation Model* used to re-engage people when they move away from the guided path. Their planner optimizes a cost function computed as the sum of different *work* definitions. The concept of *work* is derived from the forces exerted on the robot, similar to the approach of a Social Force Model. This work studies a scenario outside of the scope of this project, but data retrieved from these experiments are of interest in this project. In fact, some spatial definitions (distance between robot and guide person) can be obtained from their results.

Robot social-aware navigation framework to accompany people walking side by side Ferrer *et al.* [9] worked on a robot walking side-by-side to a human with the implementation of an *Extended-SFM* (ESFM). This EXFM incorporates a new *Robot-Person Force* to simulate the attraction to a partner. The method includes direct feedback from the users to indicate when a behavior is perceived as social or not. This grading is used to optimize the parameters of the model.

A framework for Adaptive Motion Control of Autonomous Guide robot In [10] Zhang *et al.* present a planner based on an Artificial Potential Field that includes the guided subjects and *sub-goal* location inside an office hallway. The guided person -or group- is represented as an attractive or repulsive potential field. This potential field will pull or push the robot when the *social distance* between them is higher or lower than a fixed value. In this manner, the authors achieve velocity adaptation to the guided person. This same consideration will trigger the change of behavior from *Guiding* to *Following* in the case that a person deviates from the original path.

The last two alternatives from Ferrer [9] and Zhang [10] provide an optimal background to develop the planning algorithm that adapts to the technical characteristics of Pepper. The modification of both studies into a single *Modified Extended Social Force Model* will render the best results in motion planning with Pepper. Moreover, the proposed high-level control in this project drifts apart from the constant pursue-of-Partner that both methods consider, and instead, seeks to inquire its Partner for continuation of the guiding task after re-connection with the person has been established.

1-4 Research Question

This information provides a concrete background to pose the objective of this thesis project: *how do we generate a socially compliant guiding behavior with the humanoid robot Pepper?*

In order to answer this question we propose to divide this problem in sections that work together to achieve a robust behavior. The following set of problems will be addressed in sequential order:

- Motion Planning: How do we integrate the guided person as part of the control loop that generates the trajectories?
- Adaptability to Pepper: What factors in the software and hardware characteristics of Pepper will influence the implementation of this behavior?
- Human-Robot Interaction: How can we relate Pepper's interactive design to engage the guided person in the navigating task?
- Decision Making: What are the expected scenarios in the guiding task and how can we react to unexpected situations?

1-5 Approach

The initial concern will be addressed from the perspective of path planning algorithms. The goal of this section is to include the partner as a continuous input for the generation of trajectories. State-of-the-art methods, as mentioned in section 1-3, provide a good starting point, but specific modifications need to be introduced in these methods. A portion of these changes are necessary from the perspective of Pepper's software or hardware limitations. Nevertheless, Pepper's interactive capabilities will be exploited to engage the partner in an intuitive and attractive behavior. Finally, the control structure that manages the behavior will be design to anticipate specific circumstances in the guiding behavior, e.g. a sudden deviation from the guiding task.

1-6 Contribution

This project aims to augment the interactive characteristic of guiding a person by introducing three specific ideas:

- We modified the Extended Social Force Model (MESFM) to provide cooperation and adaptation between the robot and the guided person,
- We increase interaction by adding body motions and verbal expressions within the behavior.
- We test a *search* reaction to the deviation of the Partner from the guiding task.

The *MESFM* differs from the work of Ferrer *et al.* [9] in the form that the *Robot-Partner Force* is generated. We believe that this force should work in a dual manner (repulsive/attractive) to hint a higher level of interaction between the participants. Ferrer *et al.* [9], on the other hand, use this force to create a *following pattern* as the person walks accompanied by the robot. Furthermore, Pepper's design is optimal to create human-like gestures that increase the interaction between the robot and the guided person. This approach differs from the work of Zhang *et al.* [10] where few remarks and criteria on human-interaction is applied to their Potential Field method.

A final effort is introduced to study the reaction of the robot when the guided person deviates from the guiding trajectory. Under these circumstances we propose to use previous known locations to search for the lost person in a way that imitates a natural human reaction. Upon finding the Partner, we inquire the person if he/she wishes to terminate the guiding assistance. This approach was not studied by Ferrer; Zhang [10], on the other hand, triggers a *following mode* in this situation. From experience on generating a *following mode* with Pepper, people find this reaction inadequate as the robot shows an intrusive characteristic by constantly following them.

As a final remark, the implementation of a socially-aware navigational algorithm on Pepper, without modifications on hardware or software, presents an easy solution to increase the presence of robots to the general public, as Pepper is comparatively easier to acquire in comparison to other *top-of-the-line* interactive robots.

1-7 Outline

The outline of the report follows a progressive development of the project with an initial *Chapter 2* focused on the characteristics of the humanoid robot Pepper and an explanation of the open-software ROS used to implement the system architecture. *Chapter 3* explains the aspects of Human-Robot Interaction that are fundamental for the development of this project and presents the path planning algorithm used in this behavior. *Chapter 4* describes the process of implementing the navigation planner, control and interactive modules for the guiding behavior. *Chapter 5* describes the user focused experiments used to evaluate our solution. Finally, *Chapter 6* shows the analysis of the final results as a conclusion of this project and description of future work.

Chapter 2

Experimental Platform

2-1 Humanoid Robot Pepper

Pepper is a human-shaped robot designed by Softbank Robotics. Originally created under the French company *Aldebaran Robotics*, widely known for their highly interactive robot *NAO*, it was acquired by *Softbank Mobile Group* in 2013, as the starting stone to build the *Pepper project* [11]. The concept behind Pepper shows the desired to manufacture a friendly robot companion that is accessible for a wider range of customers. It's greatest features generate the ability to understand human emotions and react in a predefined manner to those emotions.

Although most features on Pepper are pre-programmed and demonstrate a far-than-less conscious thinking process, it's structural design and software capabilities offer a great predisposition for human interaction. This characteristic renders a perfect advantage to work on social navigation. From personal experience while developing this report, people are easily attracted to the robot's friendly design.

On the technical side, the robot incorporates a range of proximity and vision sensors that enable the development of tracking, localization and navigation algorithms. The wheel drive system is holonomic, allowing a broader range of movements that fit adequately in human related scenarios. Further information on Pepper's software framework and hardware capabilities related to this project are included in Appendix A.

2-1-1 Dimensions

The principal dimensions (left) and body parts (right) are displayed in Figure 2-1. Pepper weights approximately 29 kg, with most of this weight distributed in the lower part of the body. Notably, this design specification optimizes the movement of the upper body, allowing Pepper a bigger range of motions in it's arms and torso.

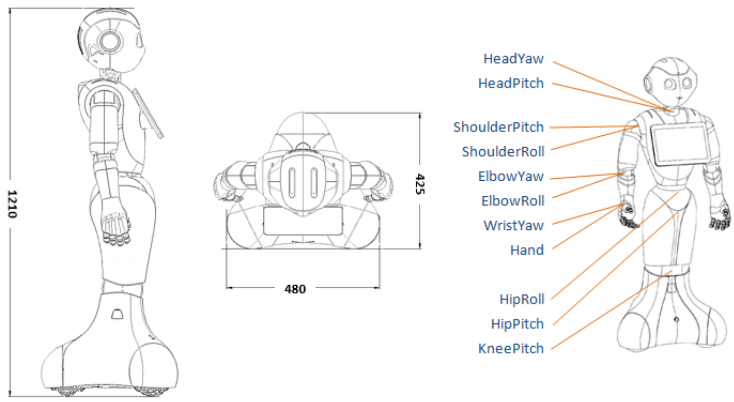


Figure 2-1: Physical dimensions and joint location per body parts of Pepper [1].

2-1-2 Sensors

This section introduces a brief description of Pepper's sensor capacity. We only describe those sensors that will be used or mentioned in this report. Refer to Figure 2-1 to identify the name of each joint in Pepper. These joints will define the relative position of the sensors mentioned in this section.

2-1-2-1 Lasers

Pepper has 3 set of lasers built-in bellow the `KneePitch` as described in Figure 2-1. The official documentation of Pepper [1] separates the control of these lasers as different sensors, but inside the ROS Wrapper [12] these 3 lasers are joined into one single sensor `node` simply defined as *Laser*. Each laser is composed of *15 beams*, but the fused Laser in the ROS Wrapper adds 16 virtual beams set to zero for the blind parts of the detection field. Consequently, the joined field of detection is set too 240°, of which only 180° are truly detection zones, as presented in Figure 2-2. The maximum distance of detection is defined up to *10 meters* with a maximum height of 10 cm for the objects that the lasers can detect.

2-1-2-2 Sonar

Pepper has 2 sonars built in the front and back of the `KneePitch`. It provides detection of objects between 0.3 and 5 meters. The field of detection is depicted in Figure 2-2. The output of the sonar is the distance in meters of the closes object in the proximity of Pepper. No other information can be provided by this sensor.

2-1-2-3 Stereo Camera

Two identical cameras are located in the top and bottom place of Pepper's head (forehead and mouth). The field of view that both cameras produce is shown in Figure 2-3. The relevant characteristics of this sensor are summarized in Table A-1 (Appendix).

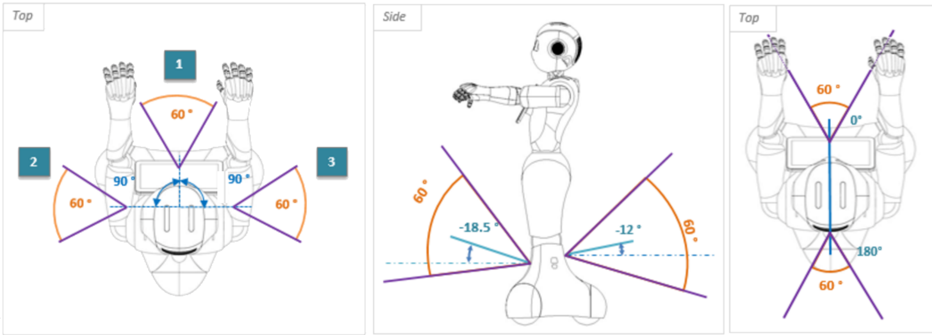


Figure 2-2: Detection zone for Pepper's lasers (left) and sonars (center and right) [1].

2-1-2-4 Depth Camera

One of the main improvements of Pepper's design in comparison to the NAO models -produced by Aldebaran-SoftBank Robotics- is the 3D Sensor located in the eyes of the robot. The depth camera is the commercially available One ASUS Xtion model which allows a field of detection as shown in Figure 2-3. The important characteristics of this sensor are summarized in Table A-2 (Appendix).

2-1-3 Motor Drive

Although Pepper has 20 motors coupled to its degrees of freedom, the scope of this project is mainly concerned with the motors connected to the three wheels at the base of the humanoid robot. The details of these motors are presented in Table A-3 and a diagram of the wheel configuration is displayed in Figure A-3 (Appendix). The wheel drive is *holonomical*, allowing a greater freedom of motion while navigating.

2-1-4 Dynamic Characteristics

A crucial aspect of this project are the velocity and acceleration characteristics of Pepper. The official documentation from Aldebaran provides the values shown in Table 2-1, but self-produced tests with Pepper showed these values to be closer to those presented in Table 2-2. These results determine specific requirements for the motion planning algorithm. An initial concern is drawn from the upper threshold for velocity. This value is well under a normal

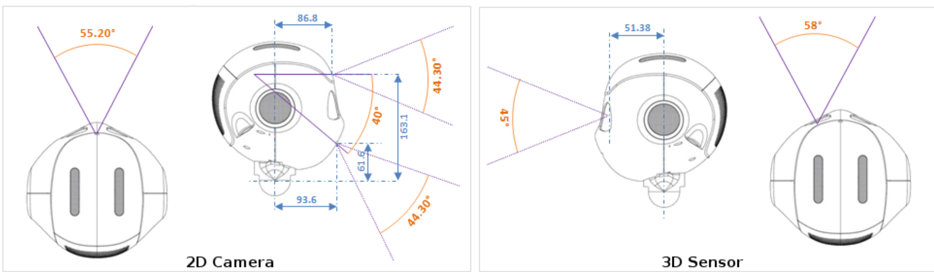


Figure 2-3: Top/Bottom cameras and 3D sensor located in Pepper's head [1].

walking pace of a human being, which might produce some discomfort from people testing this implementation. The limit of acceleration, on the other hand, should not represent any inconvenience as lower accelerations are preferable by human beings.

Drive Parameters	Default	Minimum	Maximum	Settable
Linear Velocity (m/s)	*0.35*	*0.1*	*0.55*	*yes*
Angular Velocity (rad/s)	*1.0*	*0.2*	*2.00*	*yes*
Linear Acceleration (m/s^2)	*0.3*	*0.1*	*0.55*	*yes*
Angular Acceleration (rad/s^2)	*0.75*	*0.1*	*3.00*	*yes*
Linear Jerk (m/s^3)	*1.0*	*0.2*	*5.00*	*yes*
Angular Jerk (rad/s^3)	*2.0*	*0.2*	*50.00*	*yes*

Table 2-1: Drive Parameters from official documentation

Drive Parameters	Default	Minimum	Maximum	Settable
Linear Velocity (m/s)	-	*0.1*	***0.35***	*yes*
Angular Velocity (rad/s)	-	*0.2*	***1.00***	*yes*
Linear Acceleration (m/s^2)	-	*0.1*	***0.21***	*yes*
Angular Acceleration (rad/s^2)	-	*0.1*	***0.60***	*yes*
Linear Jerk (m/s^3)	-	-	-	*yes*
Angular Jerk (rad/s^3)	-	-	-	*yes*

Table 2-2: Drive Parameters from real tests.

2-1-5 NAOqi Framework

"NAOqi is the name of the main software that runs on the robot and controls it. The NAOqi Framework, [on the other hand] is the programming framework used to program NAO [and Pepper] robots" [13].

Through the use of NAOqi, any developer can create a wide range of applications with Pepper. The supported languages for this purpose are Python and C++, but an adaptation for Java has also been developed. These applications can be developed through the modules that NAOqi provides. These modules have default methods built for specific processes. These methods are included in an API (Application Programming Interface) that enables the user to easily handle various commands that Pepper can understand. Those that have been used in the core development of this project are detail in the next section. Further information on this framework and a full list of all API methods can be found in [13].

2-1-5-1 NAOqi Motion

This is the main tool to perform different movements with Pepper, whether the user needs to animate Pepper in a fixed location or move around a defined environment. Two specific APIs are directly linked to navigation control for this robot, these are listed ahead:

ALNavigation This API provides a simple control for safe displacements with Pepper. Three API methods can be used to perform navigation-related commands:

- *Navigate to:* allows Pepper to move between two points set on a 2D space.
- *Move along:* commands Pepper to drive along the points given in a defined trajectory.
- *Find free zone:* Pepper can also find free space in an area and navigate towards this space.

After several tests with these methods, it was clear that their performance were far from optimal. When navigating an area of no more than 4 m^2, Pepper would overshoot the desired destination for more than a meter in several occasions. Collision avoidance performed adequately, but in many test-runs Pepper would opt to stop upon finding an obstacle, rather than trying to recalculate a trajectory around the obstacle. To the displeasure of the developers, there is no manner of modifying the navigation algorithm or tuning the intrinsic parameters. The lack of information about the base-algorithms limits the understanding of the problems that this behavior might be encountering in different situations. These set of inconveniences discourage further use of the *ALNavigation* module in this project.

ALMotion This module allows the user to send motion commands to Pepper. These instructions can be separated into the methods explain ahead:

- *Stiffness control:* allows the user to set a fixed torque in any of Pepper joints.
- *Joint control:* provides control over the angle position of any joint in Pepper. Feedback from the joint position is also provided through this method.
- *Locomotion control* supports displacement control through velocity commands.
- *External-collision avoidance:* allows the user to enable or disable external collision avoidance and modify the distance from Pepper where objects will trigger a collision warning.

2-1-5-2 NAOqi Audio

The set of methods in this module focuses on the various reactions Pepper can display when a sound is detected through the microphone as well as the use of the speakers for specific activities. In this project we are mainly interested in the sound detection-localization and language management derived from this module.

ALSpeechRecognition This module controls the ability and reaction of Pepper when a predefined word or phrase is recognized. The words recognized by Pepper trigger two different events: `WordRecognized` and `WordRecognizedAndGrammar`. In both instances a *confidence* value accompanies the words recognized.

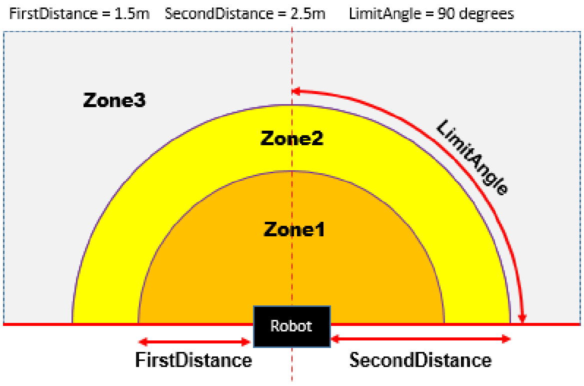


Figure 2-4: Default people detection zones [1].

ALTextToSpeech This module is used to make the robot speak in several supported languages. It sends commands to a text-to-speech engine and authorizes voice customization. The result of the synthesis is sent to the robot's loudspeakers. No complex use of this module was included in this project, only the `ALTextToSpeechProxy::say` method was applied in the guiding behavior.

2-1-5-3 NAOqi People Perception

This API module presents all the available methods that provide information about the people located within the detection zones around Pepper. From all the available modules, those that are used directly in this project are:

ALEngagementZones Allows the developer to configure the area where Pepper will try to detect a person. The default configuration is shown in Figure 2-4.

ALFaceDetection This module controls the detection and recognition of faces around Pepper. If the detected face corresponds to one of the learned faces in Pepper's memory, the ID of this face will be prompted as a callable event. [1]

Face detection is applied in the guiding behavior thought the analysis of the `FaceDetected Event`. The information retrieved from the callback is presented as a nested list, which include the following data:

- TimeStamp
- FaceInfo[N], Time_Filtered_Reco_Info

ALPeoplePerception "ALPeoplePerception is an extractor which keeps track of the people around the robot and provides basic information about them. It gathers visual information from RGB cameras and a 3D sensor if available" [1]. The significant information about the detected people can be obtained through the callback of the `PeoplePerception/PeopleDetected()`

[1] The ID of the recognized face is logged inside the `FaceInfo[N]:ExtraInfo[N]` information set.

and `PeoplePerception/PopulationUpdated()` events. In order to access this information, the developer needs to use the following *Memory Keys*:

- PeoplePerception/Person/<ID>/AnglesYawPitch
- PeoplePerception/Person/<ID>/PositionInRobotFrame

2-1-5-4 NAOqi Event

A majority of the modules explained in the previous sections work predominantly with events triggered thought the Memory module of Pepper. In order to use these events, whenever they are triggered, the developer needs to subscribe and unsubscribe from the corresponding `Memory Key`. A full list of events is presented in [14].

2-2 Robot Operating System (ROS)

"The Robot Operating System (ROS) is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms." [15]

The main advantage from ROS' structure is the flexible communication between different *Nodes* of the system. These nodes are executable programs related to specific components or control elements. On top of the hierarchical architecture the *Master* manages the communication between Nodes by providing names and registration to the nodes in the system. The transfer of information is organized through *Topics* that receives data packages in specific formats defined by *Messages*. Figure 2-5 represents a basic structure of a ROS system.

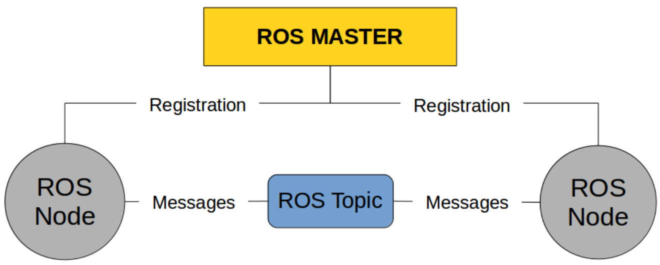


Figure 2-5: ROS Structure.

There are two types of nodes that compose a ROS system: *Subscriber* and *Publisher*. A publisher node advertises information to a Topic, while the subscriber node subscribes to a Topic and retrieves the information the instance that its published.

Additional features are provided by *Services* and *Actions* that rely on servers to trigger specific applications that can be checked continuously, in the latter case, or once the application has finished, in the case of Services.

Finally, the Master also provides configurable global variables in the form of a *Parameter Server* that nodes can use to store and retrieve parameters at runtime.

2-3 ROS-NAOqi Bridge

As mentioned in previos sections, the control of Pepper is managed by Aldebaran's NAOqi framework. A ROS package is provided by Aldebaran to link specific parts of NAOqi's API as components of a ROS system. The main component of this package is incorporated inside the `NAOqi Driver` that could also be used in other SoftBank robots that run with NAOqi, i.e. Nao and Romeo.

2-3-1 Hardware Driver

The NAOqi Driver [16] builds a connector between specific modules of NAOqi's Framework as a ROS package. There are two version of this driver written in Python and C++ respectively. This thesis works entirely on top of the C++ version of this connector assembled in ROS' Indigo version [17].

The driver retrieves data from the robot sensors straight from the lowest levels of NAOqi, therefore, secures low latency and CPU usage. Sensor data, Joint states and Robot position are published following ROS standards for robot packages. For example, all transformation frames follow the `/tf` structure and velocity commands are sent trough the `/cmd_vel` topic.

2-3-2 Basic Configuration

Apart from the NAOqi Driver, several tools are needed to built a complete systems with Pepper in ROS. These package are:

- `Startup Files`
  This package [18] includes all configuration and launch files that can be used to start all the necessary nodes linked to the Naoqi Driver and robot description.
- `Robot model`
  Simulation and visualization of Pepper within ROS is provided in this package through the description and URDF files [19].
- `Pepper Sensor Specific`
  Since the NAOqi driver established a link between NAOqi framework and ROS for all robots of Aldebaran/Softbank Robotics, additional files are required to control specific sensors and actuators of the individual robots. Pepper, for example, needs a specific controllers for the depth camera and laser. These specific nodes are included in this package.

Chapter 3

Socially-Aware Navigation

3-1 Social Interaction

There are three specific characteristics that build the notion of socially-aware navigation, as proposed by Kruse *et al*, these are grouped in *Comfort*, *Naturalness* and *Sociability.* A short definition can be summarized as:

Human Comfort Defined as the absence of stress during any human-robot interaction, this aspect is often mistaken with the concept of *safety*, i.e avoiding harming situations with a human being. In social navigation, this definition is broader and considers *avoiding any undesirable behavior* in the presence of people.

Naturalness Describes the resemblance between human and robot behaviors. The approach of most studies is to imitate human-like behaviors instead of avoiding specific reactions. An important quality involved in *natural* trajectories is the *smoothness* of human trajectories. This characteristic is not only related to the curvature or form of the trajectories, but also to the amount of energy used to reach a target.

Sociability Treated as the ability for robots to follow common cultural conventions, i.e. social protocols that people follow to avoid conflicts and maintain civil order. Achieving sociability with a robot is a task imitating human-like behaviors as well, but these characteristics are more susceptible to cultural differences.

A more general approach tends to group all of these categories into *Comfort*, since both the concept of *Naturalness* and *Sociability* describe the desire to imitate human beings in order to avoid unwanted behaviors. This project uses the latter approach to analyze the relation between a robot and human.

Furthermore, before we introduce the design of the planning algorithm used in this project, we need to define key aspects of human-robot interaction that serve as a base for our design. These consideration are summarized in Table 3-1

HRI Quality	
Comfort	People preferred moving with a robot that shows more human-like navigation behavior [21].
Naturalness	People walking together tend to walk in a side-by-side formation, only braking the pattern to avoid obstacles [22].
Naturalness	The task of walking together is performed in a collaborative manner, hence, cooperative planning is a good approach to overcome navigational problems [22].
Comfort	People find that speeds bellow 1 m/s are comfortable values for a robot (slower than average human walking speed).
Comfort	There is a different reaction of people to humanoid and non-humanoid robots [23].
Naturalness	In a guiding behavior, it's better that the robot shares the direction of the path, but not the path itself with its partner (Gockley et al. 2007).
Comfort	The distance between people increases if there is few eye-contact between people [23].

Table 3-1: Key aspects of human-robot interaction.

3-2 Modified Extended Social Force Model

3-2-1 Concept

The *Social Force Model* (SFM) was first developed in 1995 by Helbing *et al.* who proposed that, although human behavior seems chaotic and random, there are certain patterns in it's behavior that follow a so-called *Social Field.* This model for pedestrians claims that decisions made by human beings are generally automatic and determined by previous experiences, and therefore, can be modeled into an equation of motion. Explicitly, pedestrian's preferred velocity can be described by a vector quantity known as the pedestrian *Social Force.* This force is the quantification of the pedestrian's desired to move under the influence -or effect- of the forces present in the environment, i.e., obstacles, pedestrians or fixed structures, as depicted in Figure 3-1. One can model the distance to other pedestrians as a private sphere. The smaller the sphere is, the more uncomfortable the subject will feel, hence, the more repulsion the environment generates on the walking subject. This repulsion is modeled as a *Repulsive Potential* or *Repulsive Force.* In a similar way, in case the subjects wants to approach a person or an specific location, an *Attractive Force* is added to the relation. Finally, in joint human-robot behaviors, the SFM can be extended with an additional force linked to the person walking with the robot. This force is the key aspect of the motion planner (MESFM) built for this project.

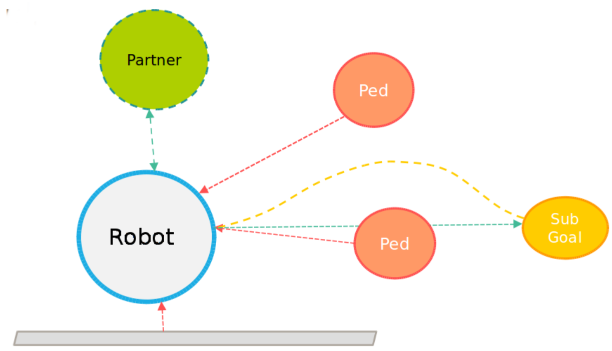


Figure 3-1: Social Forces present in human-robot interaction.

3-2-2 Forces

The original formulation of the SFM included only repulsive and attractive forces. In joint behaviors, the additional force takes a dual nature (attractive-repulsive). These forces are described ahead.

Attractive Force to Goal: This force considers the current speed of the walking subject and the future location to be reached. Mathematically, we can express it as:

$$\mathbf{f}_{i,g} = \frac{\mathbf{v}_d - \mathbf{v}}{\tau} \tag{3-1}$$

where the subscript i, g denotes the relation between the navigating agent i and the goal location g, the vector $\mathbf{v_d}$ corresponds to the desired velocity, $\mathbf{v}$ is the current velocity of the navigating agent and τ is a constant known as *relaxation time*.

Repulsive Force: The repulsive force has a exponentially increasing effect to the distance of the obstacles or pedestrians. This quantity is expressed as:

$$\mathbf{f}_{i,q} = A_q \cdot \exp \frac{d_q - d_{i,q}}{B_q} \cdot \frac{\mathbf{d}_{i,q}}{d_{i,q}} \tag{3-2}$$

where the subscript i, q denotes the relation between the navigating agent i and the feature q, e.i. an obstacle or pedestrian. Moreover, $\mathbf{d}_{i,q}$ is the vector from the navigating agent towards the detected feature and $d_{i,q}$ is the euclidean distance between these two points, e.i. the module of the vector $\mathbf{d}_{i,q}$. Finally A_q represents the strength of interaction and B_q is the range of interaction.

In case of the force is generated by pedestrians -not obstacles- a *an-isotropic* term is included in equation (3-2):

$$\omega(\phi_{i,q}) = \lambda_q + (1 - \lambda_q) \cdot \frac{1 + \cos \phi_{i,q}}{2} \tag{3-3}$$

Force	τ	A_q	B_q	λ_q
Goal	3.0	-	-	-
Pedestrian	-	1.0	2.0	1.5
Obstacles	-	0.8	1.6	-
Partner	-	0.0008	0.15	-

Table 3-2: Modified Social Force Model parameters.

where the constant λ_q represents the strength of an-isotropic relation between the navigating agent and the pedestrian; and $\phi_{i,q}$ is the relative angle of the repulsive force. This additional term accounts for the an-isotropic influence of the pedestrian in the navigating agent. Consequently, equation (3-2) for pedestrians is expressed in this form:

$$\mathbf{f}_{i,p} = A_p \cdot \exp \frac{d_p - d_{i,p}}{B_p} \cdot \frac{\mathbf{d}_{i,q}}{d_{i,q}} \cdot (\lambda_p + (1 - \lambda_p) \cdot \frac{1 + \cos \phi_{i,p}}{2}) \tag{3-4}$$

Partner Force: The inclusion of this force is fundamental for the objective of guiding a person. As mentioned in Table 3-1, a key aspect of joint-navigation is walking together in a side-by-side formation. Ferrer *et al.* [9] applied this force to accompany a person's trajectory, but the nature of this force was only attractive to create a *following* motion. A similar approach was tested by Zhang *et al.* [10] with a *Partner Potential* acting as a repulsive and attractive potential. Unfortunately, applying the same concept to the original definition of the SFM (equations (3-1) - (3-2)), resulted in awkward and *jerky* motions in our experience. From this perspective, we though that this relation would better be represented as a exponential function that generates a smooth transitions between attractive and repulsive velocity commands. The force was finally configured in the next form:

$$\mathbf{f}_{i,a}(d_{i,a}) = \begin{cases} d_{i,a} < 0.6 & -A_r \cdot \exp \frac{-d_{i,a}}{B_r} \cdot \frac{\mathbf{d}_{i,a}}{d_{i,a}} \\ \\ d_{i,a} \geq 0.6 & A_a \cdot \exp \frac{d_{i,a}}{B_a} \cdot \frac{\mathbf{d}_{i,a}}{d_{i,a}} \end{cases}$$

Total Force: The final force governing the motion of the navigating agent is obtained with the summation of all forces actuating on the agent at the same time instance:

$$\mathbf{f}_i = \alpha \cdot \mathbf{f}_{i,g} + \beta \cdot \mathbf{f}_{i,a} + \gamma \cdot \sum_{p \in P} \mathbf{f}_{i,j} + \delta \cdot \sum_{o \in O} \mathbf{f}_{i,o} \tag{3-5}$$

where the subscript p, P denote pedestrians and o, O represents obstacles and the factor α, β, γ and δ represent the trade-off between forces. Assuming a *unity mass*, the resulting *Total Force* will output the velocity for the navigating agent.

3-2-3 Parameter Tuning

Tuning of the parameters of the MESFM was first accomplished by understating the reaction of each force in simulated static scenarios. These values were then tested in real experiments with Pepper and modified accordingly. Table 3-2 presents the final parameter values used for implementation. The nature of these forces is observe by plotting their magnitude around the interactive zone of Pepper, as shown in Figure 3-2, Figure 3-4 and Figure 3-3.

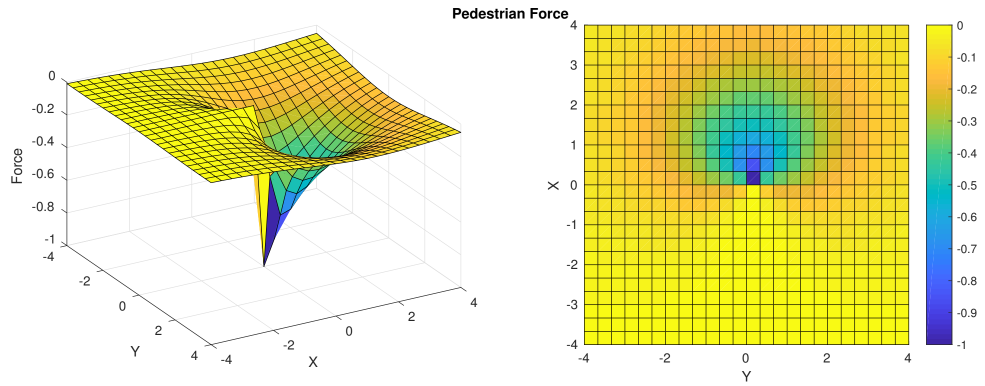


Figure 3-2: Pedestrian force expanded through Pepper's interactive zone.

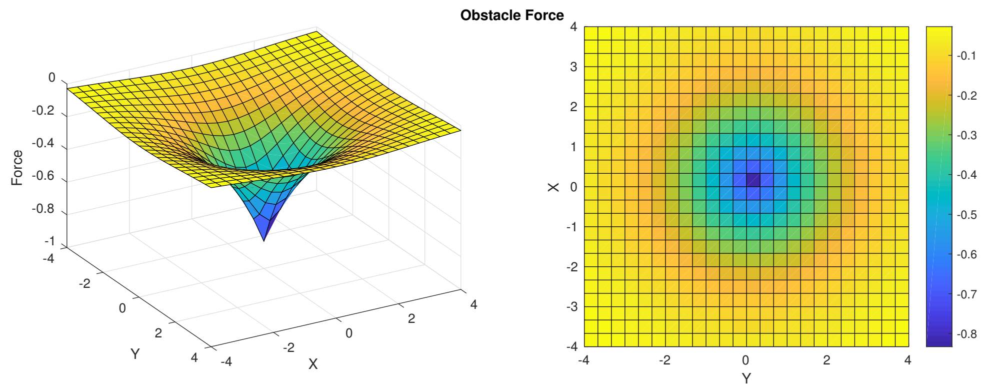


Figure 3-3: Obstacle force expanded through Pepper's interactive zone.

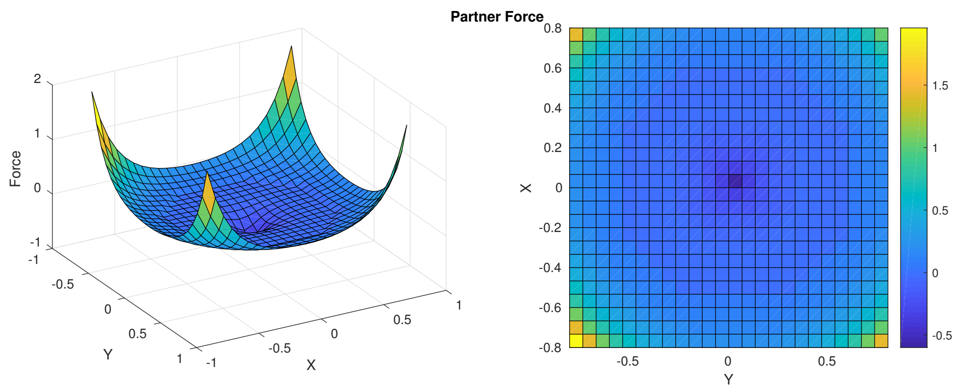


Figure 3-4: Partner force expanded through Pepper's interactive zone.

3-3 Mode Control

Once the local planner was designed, we needed to specify the general algorithm that would manage the complete behavior, e.i. the high level control. In order to address all possible outcomes and inputs to our system, it was necessary to analyze the scope of all events that could occur while Pepper guides its Partner. Recall that in Section 1-3 we mentioned how other authors addressed this concern. Garrell *et al.* [8], for example, used a group of robots to re-engage people when they moved away from the guided path. Fiore [7] also tried to re-engage people when they stopped unexpectedly by showing interesting information on the robot and waiting an specific time until the robot disengaged from the activity. In our case, we try to exploit Pepper's interactive design to inquire the intention of the departed Partner.

3-3-1 Standby Event

We expect to position Pepper around an area that has the highest field of view for incoming visitors. We will refer to this specific location as `Home`. Since the scope of this project focuses on the navigation behavior, we won't study the procedure that Pepper follows when it waits for a visitor's request. Hence, the simple control loop for this event can be summarized in the flow chart of Figure 3-5.

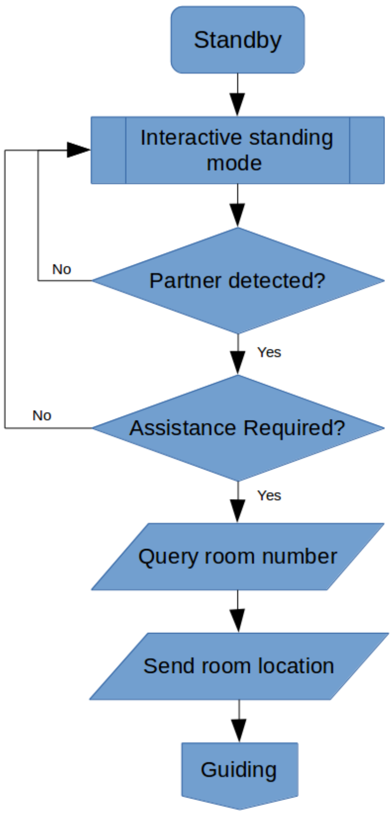


Figure 3-5: Pepper awaiting request to provide guiding service.

3-3-2 Guiding Event

The flow chart in Figure 3-6 shows the general algorithm that Pepper follows when a *Guiding request* is triggered by a visitor of the floor. The main loop begins as soon as Pepper knows where its Partner wants to go. In this loop, Pepper checks its Partner location constantly. If this condition is not fulfilled, a new control event is triggered, otherwise, Pepper guides its Partner towards the specified location. After arriving to the goal, Pepper will pronounce its farewell to its Partner and return to its `Standby` position.

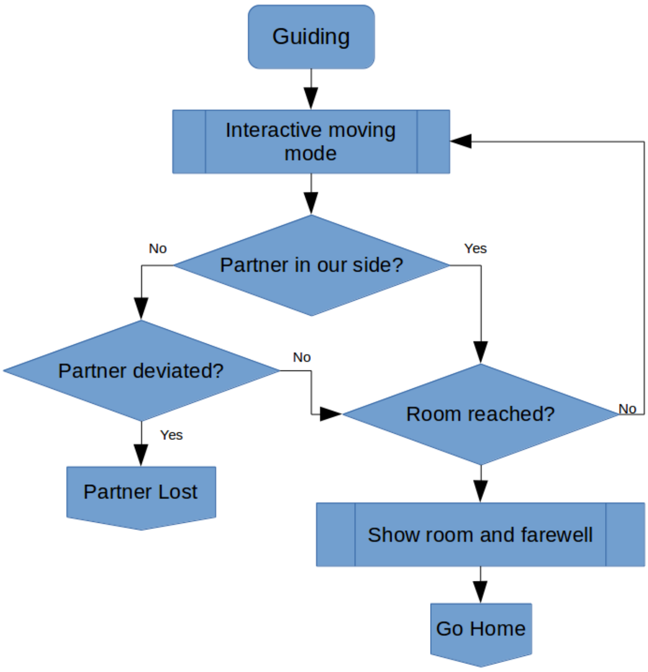


Figure 3-6: Pepper guiding known Partner.

3-3-3 Partner Deviation Event

As soon as Pepper understands that its Partner has deviated from the guiding task, Pepper will slowly stop and face the side where the person was last tracked. Pepper, then, proceeds to move to the `last known location` where the presence of its Partner was confirmed as an initial step to search for its Partner.

As mentioned in the beginning of this document, monitoring our Partner is a complex task as the only reliable detection of this person is achieved with the cameras located in the front of Pepper's head. Specific details on the solution of this problem are presented in the next chapter.

Moving alone, if during the searching procedure Pepper detects its Partner again, Pepper will inquire the intentions of the guided person. If its Partner wants to be guided, Pepper switches to `Guiding mode`, otherwise returns `Home`. On the other hand, if Pepper reaches the

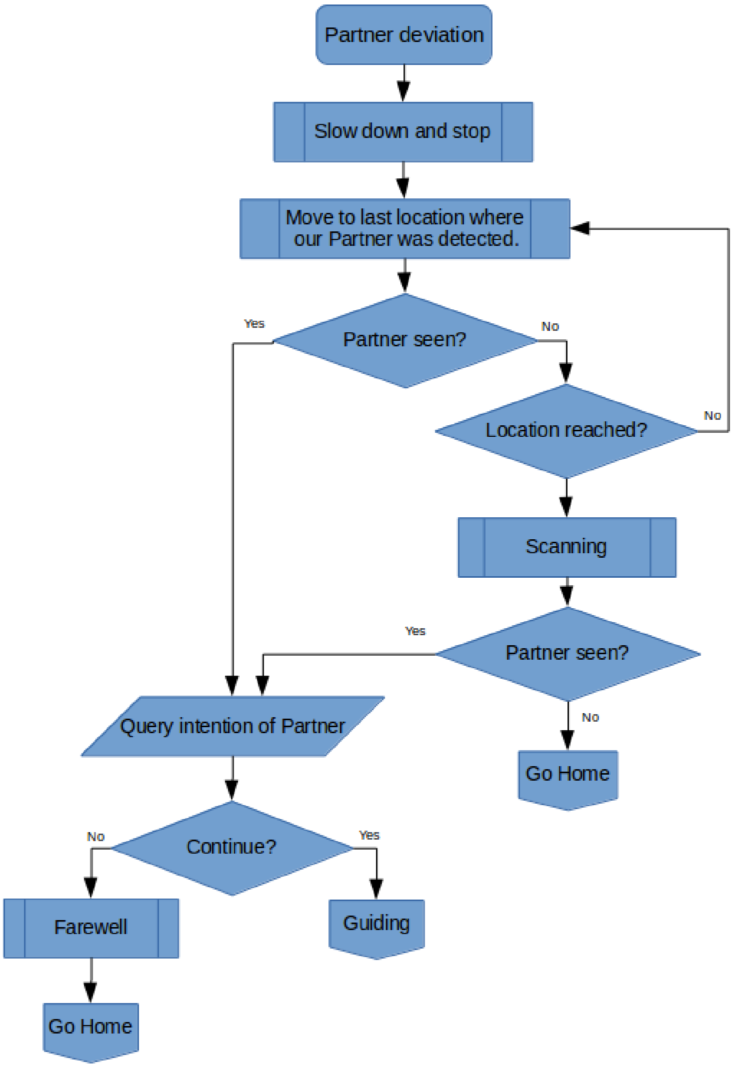


Figure 3-7: Pepper reacts to Partner's deviation from path.

last known location of its Partner and no detection was prompted in this routine, Pepper will perform a final scanning rotation. After a short time interval, Pepper stops the searching procedure and returns `Home` to wait in `Standby mode`.

The concept behind *searching* for the Partner is a consequence of the limitations on constantly detecting the Partner, but it serves as a natural reaction of any human guide when the other person is lost unexpectedly. In a side-by-side walking pattern, humans tend to focus predominately to the front of the path, only turning our head at momentary interactions with the other persons. Therefore, our sense of detection is also highly dependent on our sight, hence, only visual confirmation secures the presence of the guided person. Following these basic understandings, the design of the *Guiding Event* and *Partner Deviation Event* seem accurate enough to imitate common human reactions.

Chapter 4

System Configuration

One of the main concerns of this project was the feasibility of implementing a navigational algorithm that would run optimally in Pepper. The initial step required in this process was to build a map of the hallway. After the mapping process was finished, a localization algorithm needed to match the real localization of Pepper to the virtual map. An efficient implementation of both algorithms would allow Pepper to navigate autonomously with any motion planner. On top of these components, a high level control of the system needs to track all events to send specific decision to Pepper.

4-1 Odometry

The initial step to built a navigation system is to check the accuracy of the odometry provided by the robot. In our case, odometry measurements are published as a topic inside the ROS package that takes the data directly from `NAOqi`. Two specific test were performed on Pepper to check this information. These tests seek to evaluate rotational odometry and translation odometry separately.

4-1-1 Rotation Odometry Test

The first test aimed at checking the precision and consistency of the rotational odometry. The following steps were performed on various occasions; one of them shown in Figure 4-1:

1. Visualize laser scans in a 2D map centered on the robot,
2. Increase the time each laser point is displayed on screen (Decay Time),
3. Perform an in-place rotation of at least 720°,
4. Verify that the detected features overlap its previous location on the image.

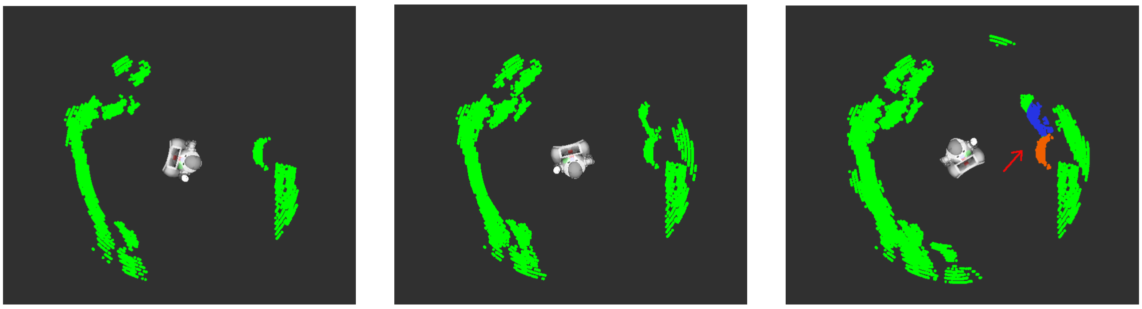


Figure 4-1: Rotational odometry test shown in Rviz.

In theory, the laser scans would match identically on each rotation, but a small drift is expected on robotic systems. Generally, this difference is not greater than a few degrees. In this robot, the laser scans begin to show noisy information after the initial rotation. For example, observe the round feature (a trash container placed near Pepper) pointed in red on the right image of Figure 4-1; after just one rotation its position has shifted more than 10°, hinting an accumulating error in rotational odometry.

In order to confirm this idea, an additional test measured the angular displacement published by the built-in Odometry compared to a fixed grid on the ground. The difference between Odometry and real angular position are shown in Table 4-1.

Rotation on Grid (°)	Rotation Odometry (°)	Error
0	0	0.0
-90	-78.81	-11.18
-180	-162.03	-17.97
90	105.93	-15.93
0	33.41	-33.41
-90	-43.78	-46.21
-180	-132.53	-47.47
90	136.64	-46.64
0	60.28	-60.28

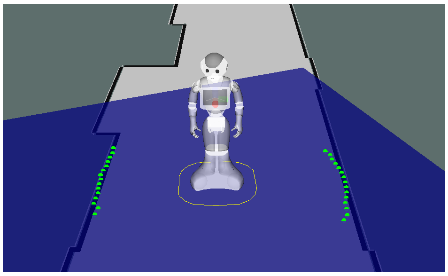
Table 4-1: Comparison between Odometry and a real values on consecutive rotations.

As its observed, the error is not constant through rotations, therefore, no fixed value can be introduced to eliminate this problem. Unfortunately, there is no permission to manipulate the process that outputs the Odometry from `NAOqi`, which implies that no direct solution was found to eliminate the error in Odometry.

4-1-2 Translation Odometry Test

Furthermore, a test for translation odometry was performed on Pepper as well. In this case the examination follows the next steps:

.



(a) Distortion shown in the right side.

(b) Pepper in the middle of the hallway.

Figure 4-2: Curvature of laser on straight surfaces.

1. Place the robot in front of a straight feature, e.g. wall,
2. Visualize laser scans (RViz),
3. Increase the time each laser point is displayed on screen (Decay Time),
4. Command a forward move on the robot until its close to the straight feature,
5. Check the laser scans forming a single line on screen.

Positive results should show that the laser scans mark a constant line as the robot moves towards the straight feature. In this occasion, Pepper showed satisfactory results with scans spreading no more than 5 cm.

4-1-3 Laser Scans

A verification of the laser scans is also important to avoid inconsistencies while mapping a room. As mention in Section 2-1-2-1, the maximum distance of detection from the lasers is set to *10 m*. Nevertheless, after some basic work with Pepper and validation of the sensors, the optimal distance for a proper detection was found to be around *1.2 m*. Above this distance the sensor's scans show distortion, as the example presented in Figure 4-2. Notice how the lasers incorrectly bend out from the wall that the laser is detecting. Since the data from the lasers are directly published from `NAOqi`, no modification is feasible with this problem.

4-1-4 Depth Camera as Laser Scan

A feasible solution to retrieve information from the environment is to use the 3D camera set in Pepper's head. The 3D camera outputs a `depth image` that detects physical objects up to *8 m* in distance. This image can then be converted to a simulated `laser scan` by taking a specific pixel row of the matrix returned by the camera. In order to check for objects around Pepper, the lowest pixel row of the image is used to create the simulated laser scans. The results of this conversion are shown in Figure 4-3.

Nonetheless, the limitation on the trade between built-in laser and the simulated laser from the depth camera is the position where the 3D camera is mounted. This location is bound to be blocked by people with more frequency, as Pepper constantly interacts with humans.

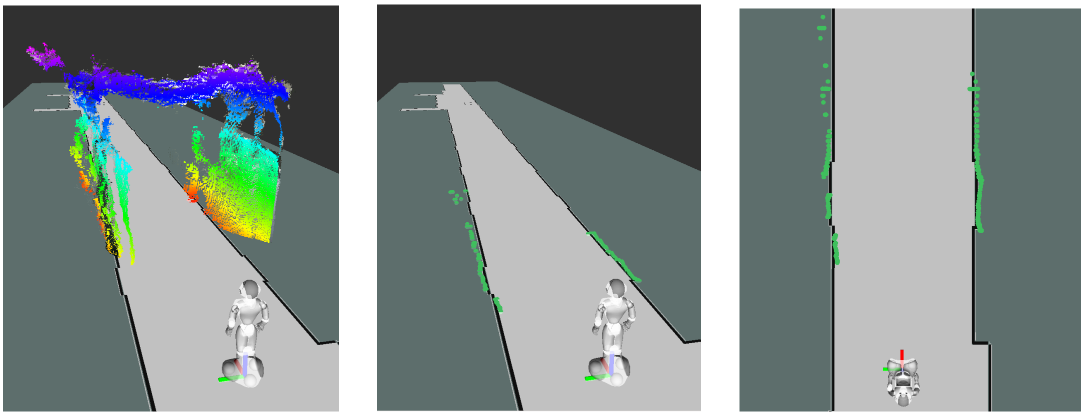


Figure 4-3: Transformation from depth image to laser scan.

Consider, for example, how the upper torso of a person can completely block the view of the camera, whereas, the legs of the same person would only partially obstruct the lower built-in lasers. The decision to use the *Depth Camera* will also restrict the implementation of frequent head movements with Pepper as the head should look forwards most of the time.

4-2 Mapping

The process of mapping the hallway of the office floor was achieved through the use of a *Simultaneous Localization and Mapping (SLAM)* algorithm. Fortunately, this algorithm is already implemented as a *ROS Package* called `GMapping` [24]. The basic requirement to run this package is a robot that provides *Odometry* data and has a fixed horizontal *laser range-finder*. Although both conditions are fulfilled by Pepper, the technical issues mentioned in the previous section proved to make this process harder to finalize.

4-2-1 Gmapping-SLAM

The preceding information was necessary to adapt the SLAM algorithm to work optimally on Pepper. The algorithm built in ROS provides parameters that can be tuned to accommodate the hardware specification of the robotic platform. The following modifications were necessary to obtain better results with Pepper, with the final values presented in Appendix D. In this process the best results were obtained through the built-in lasers in the bottom part of Pepper, therefore, the following information was adapted to these sensors.

`map_update_interval` Decreasing this value updates the occupancy grid more often at the expense of greater computational load. The limited number of laser beams in Pepper forces us to update the map at a higher frequency to avoid loosing information of the environment.

`angular/linear update` A similar behavior is expected with these values. These parameters define how fast the map is created as Pepper moves. Decreasing these values significantly will generate issues when the robot makes *re-runs* of previously visited locations, but allows the algorithm to gather more information of the environment.

`minimumScore` During initial runs Pepper would *skip* between locations. In order to avoid these *jumps* between pose estimates this value was increased to reduce the interval between matching particles in the map.

`maxUrange and maxRange` The value of *maxUrange* needs to be lower than *maxRange* to clear areas of the map where a laser beam traverses but does not hit an obstacle within the range of the laser.

`particles` High values will increase the required computational power, but the output of the algorithm will be more precise. The limited amount of laser beams and small detection zone forced us to increase this value. Furthermore, the velocity of Pepper while mapping was considerably reduced.

`srr, srt, str, stt` The odometry model noise parameters are important to reflect the confidence of the odometry information. Surprisingly, after several tests, these parameters performed better when set close the default values.

4-2-2 Results

Once the settings of the algorithm were adapted to Pepper, different approaches were tested to improve the output of the map. Initially only sections of the hallway were mapped in a zig-zag pattern by turning to a different wall every 3 meters approximately. The results proved that on every rotation the error in Odometry would introduce deviations in the map.

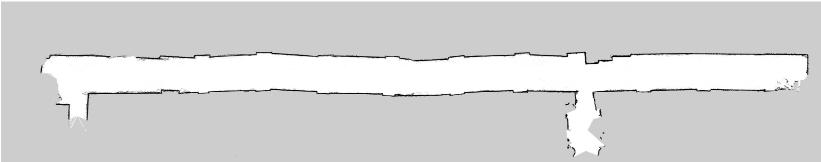
In the next experiment Pepper moved closer the walls of the hallway. This time the perimeters of the hallway were mapped correctly, but on the *way back*, i.e. returning on the other side of the hallway, the map would still shift its orientation generating unwanted curvatures.

Finally the conclusion that Pepper would generate the best results if minimum rotations were performed during the procedure was definite. This idea is backed by the accumulating error in rotational Odometry found in section 4-1. Moreover, on each run only sections of the hallway were mapped and joined manually after they were produced. It's important to mention that the velocity of Pepper was reduced to 1/4 of the maximum value and rotations were only commanded to keep Pepper moving on a straight line.

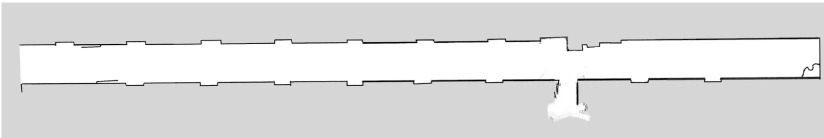
The final maps used for navigation were manually edited to correct errors and improve on the overall appearance. An example of this procedure is show in Figure 4-4. The image shown in Figure 4-4c is the final version of the map used on this project.

4-3 Localization

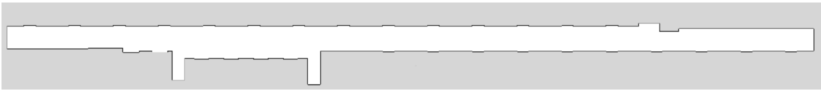
The creation of a map set the background to test a localization algorithm that would estimate the position of Pepper during navigation. The Adaptive Monte Carlo Localization algorithm built as a ROS Package [25] is a common tool used for this purpose. Once again, this algorithm had to be adapted to Pepper's hardware specification.



(a) Raw map created with Pepper.



(b) Digitally fixed map.



(c) Final version of the map used for navigation.

Figure 4-4: Final maps created with Pepper.

4-3-1 AMCL

AMCL is a probabilistic localization system frequently used in navigation. The algorithm is based on a particle filter that, given the map of the environment, estimates the position and orientation of a robot as it moves and senses the environment.

The process of tuning these parameters is better accomplished by recording a `rosbag` of the complete system or only the odometry and laser topics. For tuning purposes, the advantage of using a Rosbag is the fact that the user can constantly repeat the same conditions by replaying the recorded information and modify the default parameters to observe the change in the performance of the algorithm. The final values used in the implementation of the AMCL algorithm for Pepper are presented in Appendix D. The analysis behind these modifications is described ahead:

`min_particles` Increasing this value proved to improve the convergence of the pose estimation.

`max_particles` Allow to process more particles in each iteration to overcome the problem of only having laser scans in front of Pepper (Depth image laser scan).

`laser_likelihood _max_dist` Matching this value to the maximum theoretical range of the laser helps to gather more information about the environment.

`odom_alpha` These values were increased to spread out the samples and still obtain a good sample set.

4-3-2 Results

The performance of a localization algorithm is better tested by simple comparison of the estimated pose and the real location of the robot. A check on the convergence of the pose estimates is also commonly used in this process. In this analysis, the expected behavior of

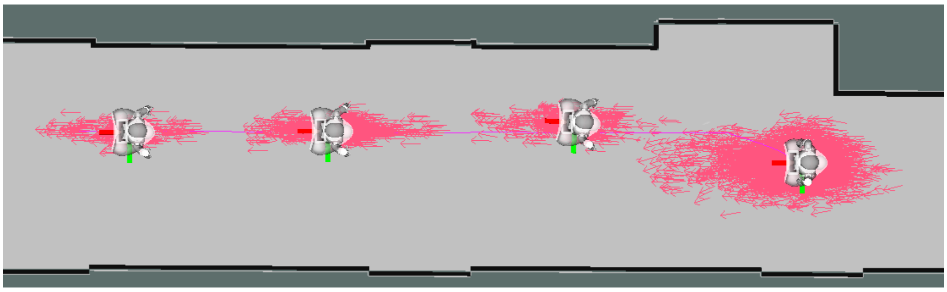


Figure 4-5: Pose estimate converging to expected value.

the algorithm should show a pose estimation that converges to a reduced area as the robot navigates an environment. In the specific case of this project, the reduced amount of features in the hallway will generate a cluster of estimates poses that stretch along the corridor, since most matches are generated with the walls. Figure 4-5 presents a sequence of images captured from RViz while Pepper navigates autonomously in the corridor of the hallway.

4-4 Partner Detection

The biggest limitation on implementing the guiding behavior with Pepper was the limited resources to constantly detect the position of the Partner. The only way to obtain accurate information is through `NAOqi` face detection, as mentioned in Section 2-1-5-3. Unfortunately, in order to continuously detect the Partner's position, Pepper's head needs to be rotated toward one side so that both stereo cameras point towards the person. Obviously, this alternative presents three major problems:

- Localization will be deteriorated with a limited view of the environment.
- Incoming pedestrian are not detected, nor avoided.
- This awkward head position does not resemble human behavior.

Therefore, it was necessary to find a different solution. The other two sensors available to recover information from the environment are the sonar and the laser. The former option provides the distance of the closest object placed within a narrow detection zone in the back of the robot. Although this information is quite limited, it does provide a minimum notion of the presence of someone -or something- in the back of Pepper. Furthermore, the laser's detection zone is located within the expected position of the Partner if the walking partner is side-by-side. Hence, a joint actuation of the laser and the back sonar is the optimal solution for this problem.

Recall that the sonar can detect up to 10 meters in distance, but we need to limit this zone up to *1.0 m* of maximum detection. This is the expected area where a person would follow Pepper, shown in the bottom yellow zone of Figure 4-6. On the other hand, the range for the laser scan covers the blue-stripped area shown in the left of Figure 4-6. In order to avoid false-positives, e.g. the detection of objects other than the Partner, the laser zone is limited to *1.2 m* away from Pepper, as shown with the yellow area of the laser range. This approach

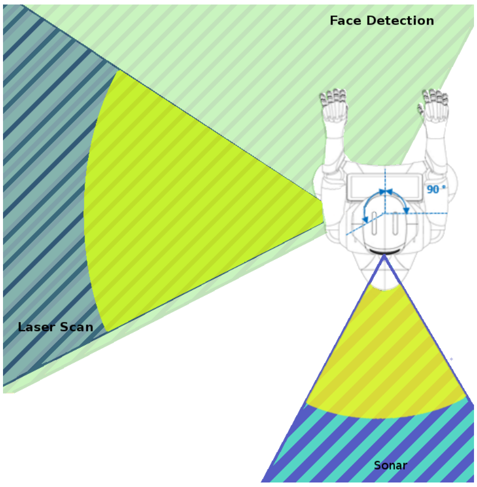
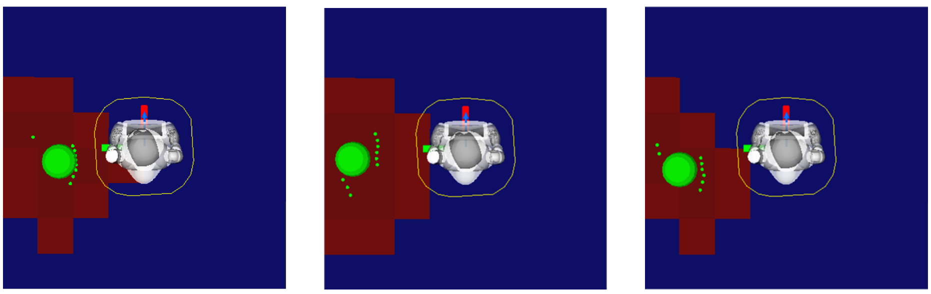


Figure 4-6: Partner detection zone.

assumes the biggest feature that can be scanned by the lasers and the sonar will be the Partner.

Once this area is defined, we need to analyze the information provided by the sensors. From Section 2-1-5-3, we know that the moment a known face is recognized by Pepper, the local pose of that person is generated through a `NAOqi Event`. We can easily retrieve this information through a ROS node subscribed to this event. In the case of the laser, the amount of information is quite limited. A maximum of 15 points can be generated through the laser scan at any time instance. The feasible solution to this problem is to define a *human-standard form* generated by a person inside the detecting zone and compare any other laser scan to this standard shape.

Consider Figure 4-7 showing the laser scans of a person standing/walking next to Pepper. The number of points within the detection zone is generally found to be to around 9 points set in a consecutive order. Without anymore information, these two characteristics can determine the presence of a person next to Pepper. Hence, we define the probability of a person located next to Pepper in the next form:



(a) Human standing. **(b)** Human walking. **(c)** Human walking.

Figure 4-7: Partner detection method.

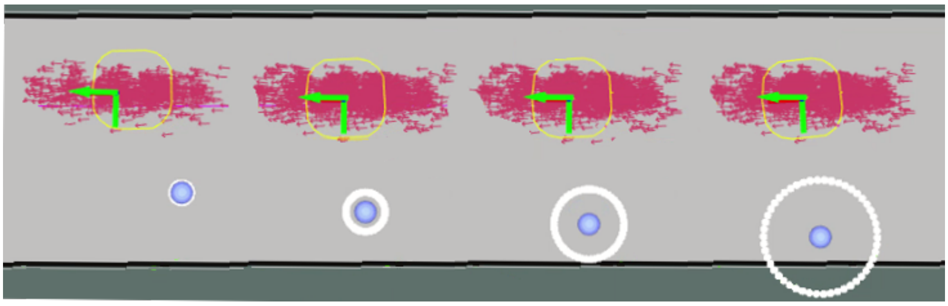


Figure 4-8: Growing uncertainty on Partner tracking.

$$P_{par} = \frac{points_k}{points_{max}} \tag{4-1}$$

where $points_k$ is the number of consecutive laser scan points in Pepper's close range and $points_{max}$ is the average of consecutive points that could indicate a person detected by the laser.

If the probability computed in this process is higher that a minimum level, the position of our partner will be computed as the average location of these points. If this probability is lower than the minimum value, we can assume that no one is near Pepper.

4-4-1 Filtering Position

Although the previous solution provides a good notion of the presence of a person next to Pepper, the estimated pose tends to skip around the predicted location, giving an incorrect impression of a person moving along with Pepper. A clear solution for this effect is to filtered out the estimated poses. We used a *Kalman Filter* with constant velocity model to improve this procedure. Consequently, the estimated poses follow a smoother trajectory next to Pepper. Moreover, the updated uncertainty provides a better indication of whether a person is walking next to Pepper, as this value will increase if no input is retrieved from the sensors. Figure 4-8 shows an example of this behavior. Obverse how the uncertainty (shown with the white circle) of the estimated pose starts to grow as Pepper looses track of the person, but the predicted pose moves in a steady manner out of the detectable zone of Pepper.

4-5 Obstacle and Pedestrian Detection

Obstacle detection is achieved by processing the data from the `Built-in Lasers` and comparing the position of these obstacles to the position of any person detected with `Naoqi's People Detection` module. If both positions are different, an obstacle is included in the area around Pepper, otherwise this obstacle is considered as a person and eliminated from the obstacle list. A simple example of this procedure is shown in Figure 4-9.

(a) Real scenario. **(b)** Pedestrian and obstacle pose.

Figure 4-9: Obstacle detection and People detection.

4-6 Face Learning/Recognition

It's important to mention that the stage for *Face Learning* was never addressed in this report, since it was defined to be outside of the scope of the project. Nonetheless, Pepper uses `face recognition` for two specific events in the guiding behavior:

- Detect visitor to engage in the Guiding Behavior.
- Detect Partner after tracking was interrupted.

This task can be easily achieved with the use of Aldebaran's multi-platform desktop application `Choregraphe` [26]. The behavior used in `Choregraphe` is shown in Figure 4-10. We can safely assume that with this application, Pepper will learn the face of its Partner before starting the experimental trials.

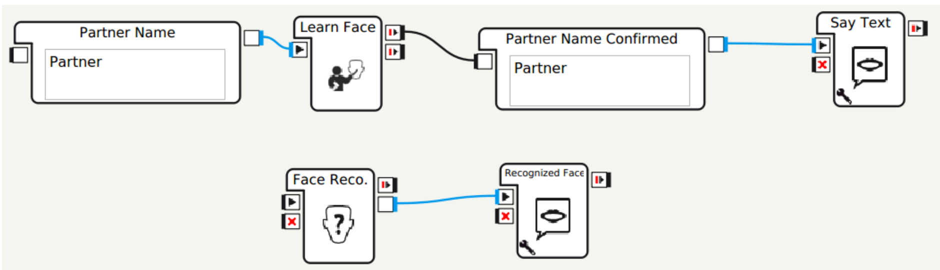


Figure 4-10: Simple Face Learning program in `Choregraphe`.

4-7 Navigation

The previous sections provide the basic setup to configure a robotic navigation system with Pepper. The overview of this structure is shown in Figure 4-11. Two sections of this system are left to be discussed: Global Planner and Local Planner. The former uses an `A* search algorithm` to generate a global plan on top of a costmap. Since no further work was performed with this planner, no additional details need to be addressed.

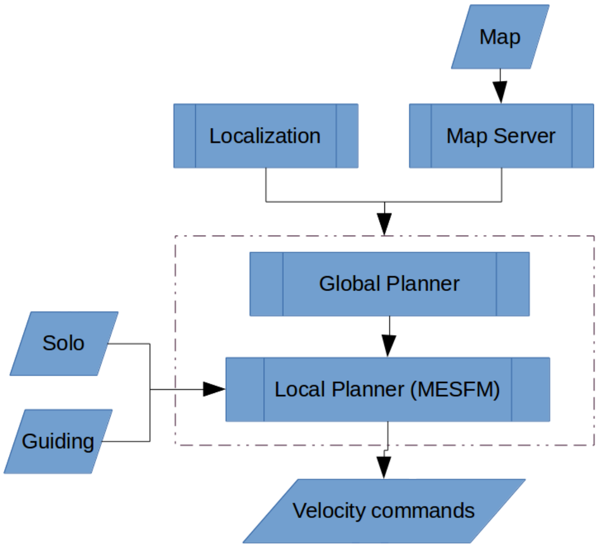


Figure 4-11: High Level view for Trajectory Generation.

4-7-1 Local Planner characteristics

On the other hand, although the algorithm behind the local planner was explained in Section 3-2, there are a few details of the local planner module that were not covered explicitly. These characteristics are briefly described ahead:

Navigation Modes Even though the local planner will always use the same algorithm to maneuver between different locations, we need to specify if this procedure will be done while guiding or moving alone. The former option is define as `Guiding` and the latter as `Solo`.

Initial Rotation At the beginning of the navigation task, the local planner makes the robot rotate in the direction of the next sub goal without any translation. The objective of the initial rotation is to show our Partner the direction of the path that we will take.

Final Rotation The final rotation pursues the same objective. Once the robot reaches the final location, the local planner sends a final rotation to show the placed that was initially solicited by the guided person. This characteristic is related to the office-guide scenario and does not occur in `solo` navigation.

Direction of rotation While testing the local planner, we found that the rotations that Pepper performs through out the entire behavior were not necessarily human-like. Consider, for example, a rotation were Pepper mistakenly faces the wall while moving. From a human perspective, such movement is not considered appropriate. Therefore, an additional constraint was introduced to the direction that Pepper chooses to rotate:

- Rotations are performed facing the open areas of the corridor and/or

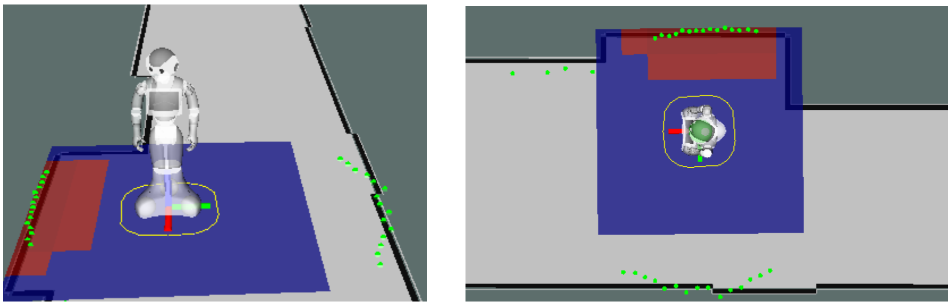


Figure 4-12: Occupancy Map shown in red and blue.

- Rotate in the direction of least action.

The first rule uses an `Occupancy Map` built around Pepper to determine the space where the least physical features are located. These features could be walls, furniture or any other object detectable by the lasers. A screen-shot of this local map is displayed in Figure 4-12. The second constraint checks if the rotational displacement defined by the `Occupancy Map` is much greater than the normal rotation Pepper chooses by calculating the difference between poses. If the former value is greater that 90° in comparison to the normal difference, Pepper will choose to take a normal rotation, as this is the solution with the least effort. This idea follows the general understanding that humans tend to perform the most efficient actions while navigating [23].

4-7-2 Hallway trajectory

The last concern that we addressed in the social-interaction of our robot is the expected behavior of walking on one side of the hallway. We modified the trajectory generated by global planner to achieve this objective. The local path is then restricted to the right side of the hallway, thereby limiting the passing maneuvers to the left side of the hallway. The distance from the walls can be modified, but, from different trials, this distance was set to 0.30 m away from the center of the corridor.

4-8 Interaction Control

Once the navigation system was implemented with optimal results from Pepper, it was necessary to investigate which functions from NAOqi were important to evaluate in the guiding behavior. Recall from Section 3-3 that 4 modes need to be design for different routines within the control structure:

- Interactive Standing mode
- Interactive Guiding mode
- Show Room and Farewell behavior
- Search for Partner behavior

4-8-1 Interactive Standing

While Pepper waits for a visitor to trigger a guiding request, Pepper is set to *Awareness On* mode. This feature from NAOqi enables Pepper to react to sound and face detection. The default settings for sound detection are not modified, but face detection is set to a maximum distance of 4 meters.

It's important to mention that Pepper is easily attracted to sounds around its environment. This characteristic can create awkward reactions from Pepper, i.e. it could end up facing the wall if a sound is triggered next to this structure. In order to avoid this irrational behavior, we set the awareness mode to *Head.* In this mode Pepper will track sounds/people only with its head without any body displacements. This decision sets Pepper to a fix location facing the most probable entrance of visitors.

4-8-2 Interactive Guiding

The purpose of augmenting the cooperative action of guiding a person was addressed from the motion planning point of view in Section 3-2-2. Now we aim to increase this interaction by exploiting Pepper's humanoid design. Recall from Table 3-1, three key features that we need to attend:

- People prefer moving with a robot that shows human-like behaviors [21].
- There is a different reaction of people to humanoid and non- humanoid robots [23].
- The distance between humans increases if there is few eye-contact between people [23].

4-8-2-1 Following pattern

Due to limitations on the sensors of Pepper, a key aspect of the implemented behavior is the requirement that the guided person prefers to walk on the left side of the robot, where a laser is able to scan items inside a narrow detection zone. In order to persuade the person to walk on Pepper's left side, Pepper says to the person being guided: *Would you mind walking by my left side* and points its left arm to the front-left area.

4-8-2-2 Partner engagement

This point is addressed by using verbal interaction between Pepper and its Partner. We won't devote research into the communication skills of robots, since this field of research is complex enough on its own. Engaging in a normal conversation or *small talk* is far too complex to be implemented inside this project. Therefore, we are limited to generate verbal feedback of the progress of the guiding task. It is clear that we don't expect any reaction from the Partner.

Beginning of task: As soon as Pepper confirms the objective location, Pepper pronounces at random one of the following phrases:

- *Please follow me.*
- *We are ready to go.*
- *We may leave now.*
- *Come with me please.*

During navigation: Once Pepper starts to move with its Partner, a friendly reminder of the progress towards the final goal will be announced by Pepper. Only three progression rates are considered in this mode, with a corresponding phrase pronounced by Pepper:

- 30%: *We are not far away.*
- 55%: *Just a little more walking.*
- 80%: *Almost there.*

4-8-3 Show Room and Farewell

Once Pepper and the Partner reach the target location, Pepper rotates in front of the room and lifts one arm to point towards the door. This gesture is design to give a clear indication that the task is completed and where the Partner should direct after Pepper leaves toward the `home` location. This gesture is accompanied by one of the following expression, selected at random:

- *We've done it.*
- *Here we are.*
- *We've reached our destination.*

After a couple of seconds Pepper gives a warm farewell by saying:

- *I need to go back now. Have a nice day. Goodbye.*

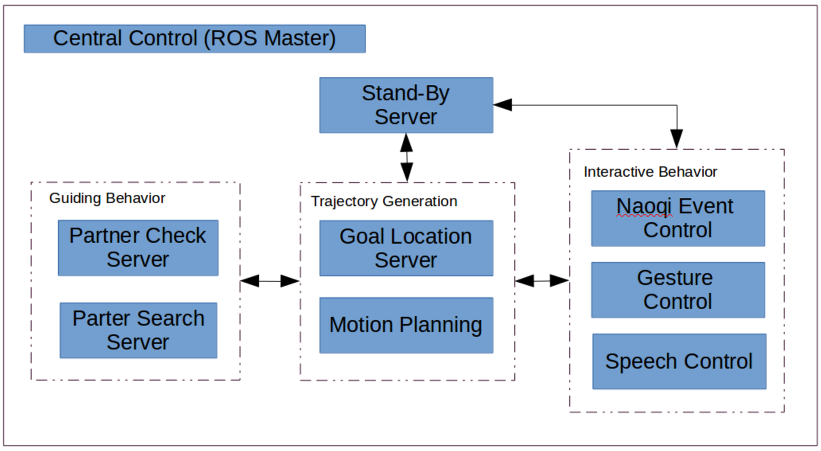


Figure 4-13: High level structure of system.

4-8-4 Searching for Partner

Although the primary design of this project was focused on the guiding behavior, a natural event that could occur during this task is the unexpected deviation of the Partner from the guiding path. In this situation, the control loop explained in Section 3-3-3 triggers a *Partner deviation mode*, shown in Figure 3-7. In this control loop we intent to search for the Partner. The initial action is set to move to the last position where we had visual confirmation of our partner. This task is managed by the motion planning control of Section 4-7 acting in autonomous mode.

Once Pepper has reached this location, a searching mode needs to be triggered. A natural reaction in human beings would be to scan the environment to find the lost person. On a humanoid robot this motion needs to be emulated such that it closely imitates a human reaction. Unfortunately, this solution might end up looking over-choreographed if implemented in Pepper. Alternatively, we found that Pepper's `Awareness On` mode generates a natural movement when it is set to respond to sound or people. Hence, the moment Pepper reaches the last know location, we activate `Awareness On` mode which causes Pepper to rotate towards sounds and people. Two benefits are achieved in this way:

- Pepper turns in different direction in a natural way, avoiding a *machine-like* scanning process,
- Partner can seek Pepper's attention by generating any sound that attracts Peppers awareness.

This process will last a up to *1 minute* of duration. After this time is finished Pepper will abandon the guiding task and navigate towards its `Home` position.

4-9 System Structure

The high level structure of the system implemented in Pepper is based in 4 components working simultaneously under the supervision of a Central Manager, i.e. the ROS Master.

The purpose of each sections is described as follows:

Stand By Serves the purpose of waiting for a *guiding request* from a person. Basic interaction is enabled and provides the goal for motion planning.

Guiding Behavior In charge of analyzing the partner within the guiding task. The constant *position checking* and *deviation event* are handled in this component.

Trajectory Generation Accepts goal locations and generates trajectories based on the global and local planner. It sends motion commands as an output.

Interactive Behavior Provides filtered information from *people detection module* and *speech recognition*. Furthermore, it sends information and controls body gestures.

The structure show in Figure 4-13 is based on the general structure of a ROS system. Both *Guiding Behaviour* and *Trajectory Generation* are built as an `Action Server` that triggers specific actions when a request is sent to the *Central Control*. The *Interactive Behaviour*, on the other hand, links the *NAOqi framework* and performs specific actions with Pepper.

Chapter 5

Experiment

5-1 Experimental Design

User studies can assess how users experience the result of a developed application. In order to secure the validity of these experiments, a protocol must be set in place to describe the sequential steps that need to be followed on each experimental trial, including situations that might interrupt or alter the experiment. We have divided this setup in three main components: the *Location* of the experiment, the *Number of participants*, and the *Procedure* to follow in each trial.

5-1-1 Location

The experiment is carried out on the 12th floor of the EWI building inside the campus of TU Delft. The elevators, marked with purple lines in Figure 5-1, are the main point of entry of people to this floor. The area in front of the elevators expands to more than 4 meters, providing the largest field of view for incoming visitors. As a result, we defined Pepper's initial position to be in the perimeter of this area, highlighted with a green cross in Figure 5-1. We refer to this specific location as `Home` throughout the test.

The corridor that links every room in this floor covers a distance of approximately 70 meters with a width of 2.3 meters. The denomination of each office-room, colored in orange on Figure 5-1, has been shorten to a number between 1 and 24. This adjustment reduces the chances of having false-positives in the `Speech Recognition` module of Pepper. Since *speech recognition* is not a testable feature of this project, this decision does not affect the outcome of the experiment.

5-1-2 Sample Size

The general assumption for user-experience tests indicates that, on average, a number of 5 people reveals at least 85% of problems found in applications if a *problem discoverability* of

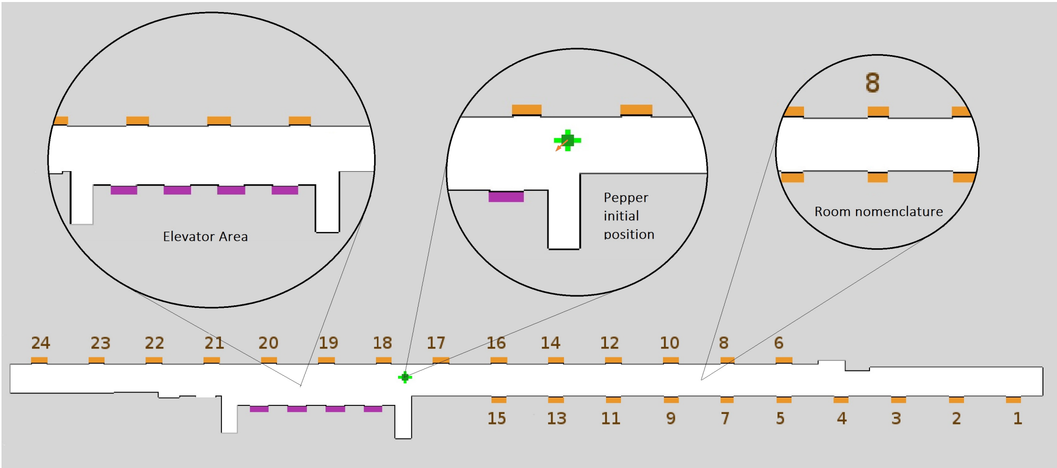


Figure 5-1: Description of experiment location. *Elevators* are colored in Purple, *rooms* in orange and *Pepper initial position* in Green.

0.31 is assumed in this calculation [27] [28]. Variation of the *problem discoverability* have hinted that 4 or 5 participants can expose 80% of usability problems [29]. Nevertheless, these values have been debated with the argument that the percentage of exposure of problems (80% or 85%) should be taken as an average, rather than a minimum.

Sauro and Lewis provide [30] an easier way to define the sample size with a problem discoverability in mind. Following the procedure in [30], let's assume that inside this user experiment the reactions we seek to find are slightly hard to find. This assumption can be quantified with a *discoverability* p equal to 0.1. Considering that human perception is quite relative and many problems could occur, our aim is to find at least an 85% of the problems that might be generated in this behavior. These two values indicate that 19 people should be use to test the guiding application.

5-1-3 Procedure

In order to describe the set of steps that are followed on every trial, we begin by defining the three subjects that form part of the test:

Instructor: Manages and supervises each experimental trial.

Participant: Assumes the role of the person being guided by the robot.

Pepper: Humanoid robot programmed as a guide.

Other people might be present during the development of the experiment, as the location remains open for normal activities in the office building. These people are considered to be part of the environment and do not receive any instructions from the experimenter. The expectation is that their behavior is relatively unaffected by the presence of Pepper and the participant, reproducing normal conditions of a human-crowded area.

5-1-3-1 Introduction

The experiment begins with a short description of the guiding behavior. Each participant receives the same information in the form of a short -A4- handout attached to this report in Appendix B. Four aspects are explained as part of this document:

- Presentation of Pepper,
- Description of Guiding Behavior,
- Explanation of participant's role,
- General procedure.
- Safety measures.

This information is reviewed in the presence of the participant. At the end of the presentation, the instructor can answer any unresolved doubts.

5-1-3-2 Face Learning/Recognition

Recall that Pepper uses `face recognition` for two important events: The first facial detection starts the guiding behavior by triggering Pepper's question about the target room, and, during the guiding behavior, Pepper uses face recognition to search for the lost person. In the latter situation, face recognition is the only feasible solution to confirm the presence of this person. Because of these design specifications, the participant's face needs to be recorded in Pepper's memory before the start of the experiment. The steps to follow in this procedure are:

1. Instructor: Launch and upload the application in `Choregraphe`.
2. Instructor: Confirm that Pepper is set to *Head* following mode.
3. Instructor: Indicate that the participant needs to stand in front of Pepper.
4. Participant: Stand in front of Pepper until Instructor says its okay to move again.
5. Instructor: Confirm that a human face is detected and is tracked by Pepper.
6. Instructor: Run application in `Choregraphe` and wait until process finishes.
7. Participant: Watch Pepper's eyes for feedback of the status of the process. Green leds indicate a successful outcome, while red leds indicate Pepper could not learn this face.
8. Instructor: Confirm that Pepper has successfully learnt the participant's face. If the outcome is successful, Pepper also pronounces the word *Success*.

In case the results of the Face Learning application are not satisfactory, two options have been tested to improve the outcome: *Change location* or *Increase light intensity*. If these alternatives do not improve the result, the participant can be dropped from the experiment.

5-1-3-3 Guiding Behavior

The previous instructions correspond to the setup of the experiment. The next set of steps define the standard sequence of activities that need to be followed in each trial.

1. Instructor: Uploads ROS launch file that starts the application.
2. Participant: Stands in front of Pepper at a distance of 1 meter approximately.
3. Instructor: Checks that Pepper is set to *Head* following mode.
4. Instructor: Confirms that Participant is ready to begin test.
5. Instructor: Launches application from command window.
6. Pepper: Recognizes Participant and inquires which room number the person wants to visit.
   - If Pepper doesn't recognize the person, the application needs to be restarted.
   - If error persists, the person needs to go through the Learning Face stage again.
7. Participant: Clearly pronounces a number from 1 to 24.
8. Pepper: Confirms Participant's *room number* by repeating it out loud.
   - If Pepper doesn't understand *room number* within a 60 second period, the application needs to be restarted.
   - If previous option is unsuccessful, the Instructor can manually enter the *room number*.
9. Pepper: Begins the guiding behavior.
10. Participant: Follows Pepper's instructions and motions.
    - If at some point during this stage Pepper behaves in an uncontrolled manner, the Instructor has the option to kill the application and stop Pepper immediately.
    - If the Participant feels unsafe at any moment, the Participant can press the Emergency Button located in the back of Pepper.

Once Pepper and the Participant have reached the final destination the experiment is over. At this point the Participant is asked to proceed with the questionnaire.

5-1-3-4 Evaluation

After completing the guiding scenario, a one-page questionnaire is handed to participant. A room close to the initial location of the experiment is used to provide a private space where the participants can fill in the survey.

5-1-3-5 Data

Each trial is recorded automatically every time the application is launched by the Instructor. These files are saved with the name of the time the application was started. A copy of each file is manually transfered to a different location where a scanned copy of the questionnaire is also kept.

The collected information is used to evaluate the outcome of the project. This information is explained in the next section.

5-2 Objectives

In order to evaluate the aspects that interests us in this project, our focus is directed to 3 specific topics: *User engagement*, *Trajectory Comfort* and *Person Tracking*. Each of these aspects are analyzed through the social constraints of *comfort*. Recall from Section 3-1 that *comfort* was linked directly to the absence of stressful situations, but it also relates the correct adherence to social rules and the robot resemblance to a human being. Both objective metrics and subjective notions are used to examine these aspects.

5-2-1 User engagement

Although both *side-by-side* and *front-back* walking patterns can be accomplished with Pepper, we want to evaluate the reaction of people following the instruction to walk on the left, as this is the preferred pattern for interactive guiding with Pepper.

Furthermore, along the guiding task, Pepper notifies the person the progression of the behavior by informing how far they are from the target location. This aspects tries to augment the human-robot interaction by attracting the focus of the guided person. At the end of the behavior, Pepper indicates the position of the target room and says goodbye to the visitor.

As this behavior is triggered in a programmed order, no objective measurements seem appropriate to evaluate it efficiency, but user feedback is obtained to gain a better understanding of how participants appreciate the guiding behavior.

The metrics used for evaluation are:

Objective metrics:

Comply to walk on left side : Only the initial position of the participant is used to define whether the person follows the instruction to walk on the left side of Pepper. This value is calculated as the average position of the 20 first measurements logged in the system.

Subjective metrics:

Level of comprehension Participants provide feedback on whether the physical gesture and verbal expression are good indicators to follow the initial instruction. This topic is inquired using a Likert Scale with the question:

- `Pepper's walking instructions at the start were clear to me.`
- `Pepper's indication of the target room was clear to me.`

Level of engagement Using a Liktert scale, participants are asked to indicate whether Pepper's comments draw their attention during the behavior with the questions:

- `I liked Pepper talking to me from the start of the behavior.`
- `I liked Pepper's remarks of our current progression towards the room.`
- `I liked how Pepper presented the room and said goodbye at the end.`

Both Comply to walk on left side and Level of comprehension are correlated metrics. The expected outcome is that if the participant followed Pepper's instructions, the designed behavior is rated positively in the level of comprehension.

The Level of engagement can be compared to the Guiding distance measured through the experiments. We expect that greater levels of attention can hint a closer distance to Pepper.

Finally, two more questions are introduced in this section to indicate the different effects of verbal communication compared to non-verbal interaction:

- `Bodily expressions helped to understand and follow Pepper.`
- `Verbal expressions helped to understand and follow Pepper.`

5-2-2 Trajectory Comfort

The common parameter evaluated in socially-aware navigation is the quality of the trajectories used by the robot to displace itself between different location. The most important objective of a guiding behavior is the successful achievement of a natural interaction between the robot and the human. The Modified Extension of the Social Force Model seeks to produce this quality in the guiding behavior by generating trajectories that respond to the human walking motions. In order to evaluate the outcome of this goal we propose to use the following set of metrics:

Objective metrics:

Guiding distance: Corresponds to the euclidean distance between the Partner and Pepper. Studies have hinted that this distance oscillates between 1 and 1.2 meters, but in the case of Pepper, this value is considerably reduced by two factors: narrow corridors and low speed. A formal approach to analyze this value is through a comparison of personal zones defined in *Proxemics Theory*.

Path smoothness: Sudden or unexpected motions cause discomfort in users of a robotic application. Smooth movements, on the other hand, create the feeling of security. In order to evaluate this aspect, we fit Pepper's trajectory to a second order curve and compute the normalized sum of the squares of the errors as a measure of *smoothness*. Lower values of error indicate a *smoother trajectory*.

Motion jerkiness: Abrupt changes in speed can also decrease the level of comfort in a person walking with Pepper. In this case, both change in speed and acceleration are registered to analyze this metric. Lower change in speed and acceleration can hint that motions were generally more comfortable.

Preferred guiding position: The participants pose is continuously logged in the system as soon as the behavior begins. These poses are filtered into two area segments shown in Figure 5-2. This information can be used to identify the preferred position to follow Pepper.

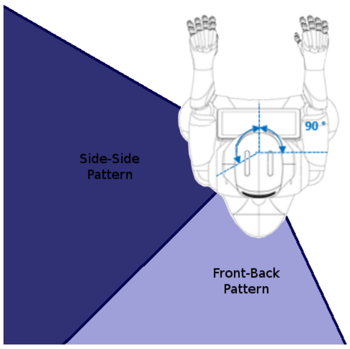


Figure 5-2: Guiding patterns evaluated.

Subjective metrics:

Perception of comfort: Participants will rate Pepper's motions in relation to their personal feeling of comfort using a Likert Scale questionnaire:

- `I liked the distance Pepper kept while guiding.`
- `I liked to walk on the left side of Pepper.`
- `I liked how Pepper moved through the corridor.`

Intuitive motions: Participants provide feedback on whether Pepper's motions were easy to follow through the guiding behavior by answering the following Likert Scale questions:

- `It was clear to me where Pepper was going.`

Perception of comfort is related to Path smoothness, Motion jerkiness and Guiding distance. The analysis of these metrics can provide a clear understanding of the level of comfort that this application can generate.

5-2-3 Person Tracking

Either caused by a faulty detection of the sensors, an occlusion event where the Partner is momentarily blocked or the Partner's decision to abandon its guide, there is a good probability that the guided person is temporally lost while performing the guiding task. As a reaction to this event, Pepper triggers a `Searching Mode` in hope to find its Partner. The metrics that provide feedback for this mode of operation are:

Objective metrics:

Time between *Lost* and *Found* events: The effectiveness of this property can be measured through the time it takes Pepper to locate its visitor after it lost track of the participant. Long periods of time can indicate that the participant did not understand Pepper's reaction or that this reaction is not a natural answer from a guiding robot.

Subjective metrics:

Joint navigation: Participants can indicate if they feel that Pepper includes their presence in the guiding task with the following question:

- `Pepper knew that I was following all the time.`
- `Pepper always reacted to my presence.`

Tracking comfort: Participants can rate whether Pepper's `searching mode` is easy to *relate* and *comfortable* to follow by answering these questions:

- `I liked the way Pepper indicated that it lost track of me.`
- `I liked how Pepper reacted after it lost track of me.`
- `I liked how Pepper recognized me.`

5-2-4 Questionnaire

A paper-based questionnaire is presented at the end of each experiment. The questionnaire is divided in four sections corresponding to the four objectives we seek to evaluate. The questions presented to the Participants have *Likert Scale* format of 5 points. The questionnaire is attached in Appendix C.

Chapter 6

Results and Conclusions

6-1 Results

The user experiment was carried out with 13 participants in total; 10 students and 3 faculty members of TU Delft. The average age of the participants was determined to be 26. Out of the 13 participants, 11 were males and 2 females. Except for one user, all other participants have an engineer-technology related profession.

The original plan contemplated the use of 20 users of the application. Due to time constraints, this goal was not achieved. Nonetheless, a noticeable trend was observed through the participants' answers during post-interview, which leads to the idea that the results presented in this section do reflect a conclusive opinion of the objectives we wanted to evaluate.

6-1-1 User Engagement

User engagement was evaluated by measuring the influence of verbal and non-verbal expressions from Pepper. In particular, the initial instruction and the indications at target room were used to determine how participants react to Pepper. The outcome of the experiment, as seen in Figure 6-1, shows that the majority of people accepted the initial instruction and moved towards the left side of Pepper, even though this was not their preferred following-pattern. Question 1 (Q1), depicted Figure 6-2 corroborates this idea, as 83% of subjects found Pepper's expression clear to understand and intuitive to follow. Similar results are seen with Question 2 (Q2) in Figure 6-2 with the evaluation of Pepper's behavior at the target room.

Nevertheless, Question 6 and 7 (Q6-Q7), in Figure 6-2, indicate that these instructions were easier to comprehend through Pepper's verbal expressions, rather than body motions. In the post-interviews, the participants mentioned that some motions were *misleading* or *not evident*, but verbal expressions made the interaction easy to follow. Those participants that liked Pepper's body motions said that these aspects made Pepper *friendlier*, like a human guide.

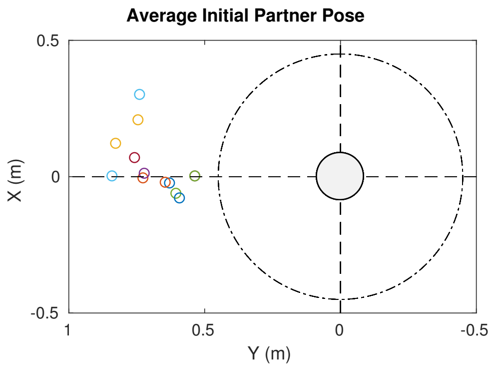


Figure 6-1: Averaged initial pose for participants. Pepper shown as gray circle and *Personal Space* shown with dashed lines.

Furthermore, Pepper's final comments were sufficient to know the task had ended and their target room was reached. Only 1 person felt Pepper could position itself a bit further away from the door, as to not block the path.

Finally, the results from the evaluation of *Level of engagement*, Questions 3 though 5 (Q3-Q5) of Figure 6-2, were notably positive. A majority of participants felt that Pepper's interaction through the guiding task was pleasant and resembled a human guide. Many participants felt that this aspect would serve even better for longer trajectories. Those participants that rated this aspect less desirable had the following remarks:

- Small talk would be a nicer option: *How's your day?, The weather is nice* , etc.
- Fun facts would be more interesting to hear.
- Depending on the mood of the person, interaction is unnecessary.
- Instead of progression, mentioned time until room is reached.

Other results were also proposed on Pepper's structural designed as some participants felt that Pepper's *breathing*, *facial expression*, and *eyes* were awkward.

6-1-2 Trajectory Comfort

The average distance that the participants kept from Pepper is shown in Figure 6-3. It oscillates between *0.8* and *0.7* meters away from Pepper. Although these measurements variate during the guiding task (shown by the error bars), this change is not significant, reaching at most *20 cm*. Furthermore, the results from Question 8 (Q8) show that 67% of the users felt comfortable with walking at this distance away from Pepper.

The effect of walking on the left side, nonetheless, did cause distress with at least half of the participants. As seen in Figure 6-4, the average position of every person falls in Pepper's left side, but the *Level of comfort* shown in Question 9 (Q9) of Figure 6-2 shows that the walking pattern was not perceived positively by 84% of the users. The preferred position s/he would take to follow Pepper are summarized as follows:

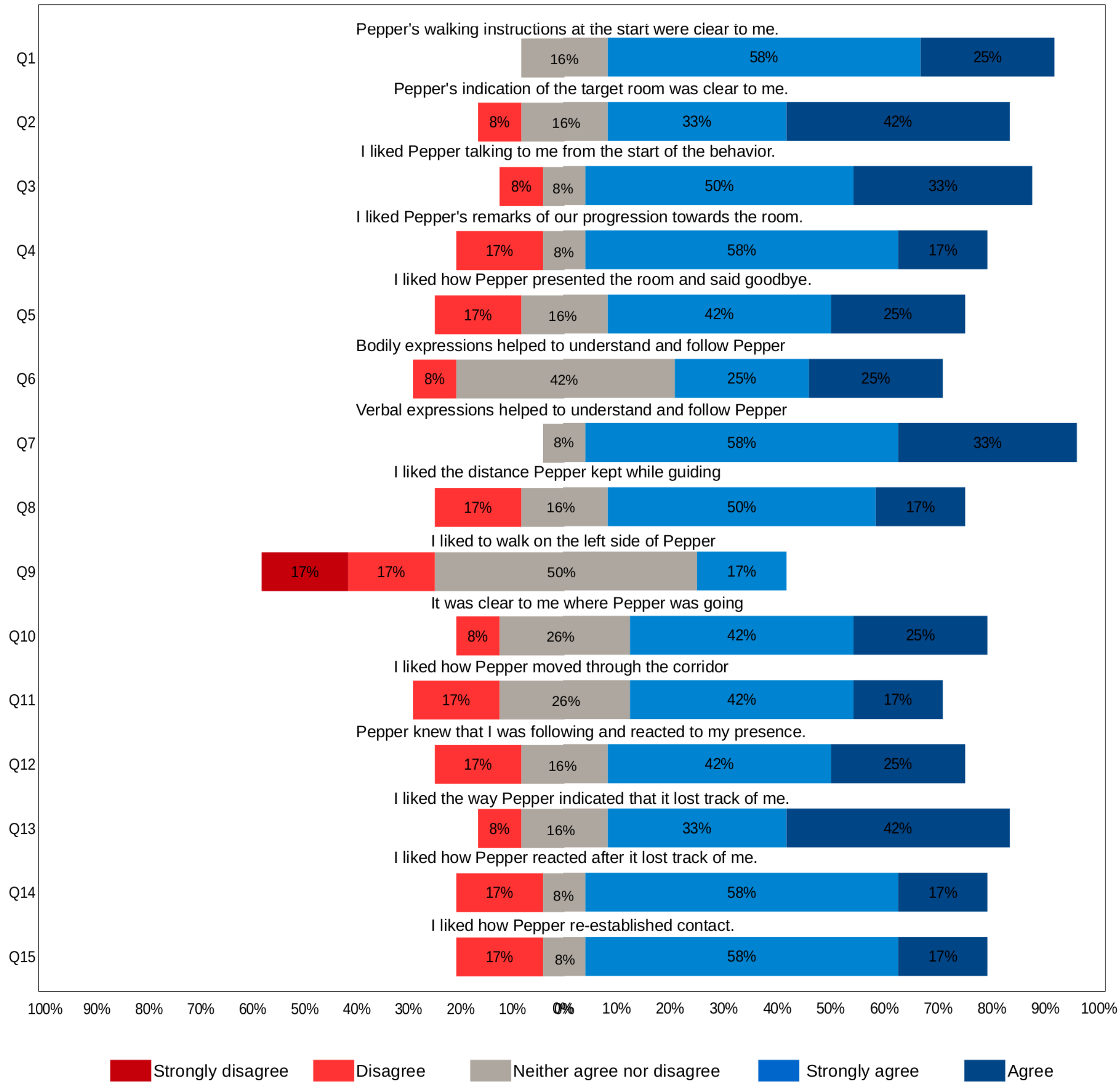


Figure 6-2: Results from questionnaire presented as a Likert Scale Plot.

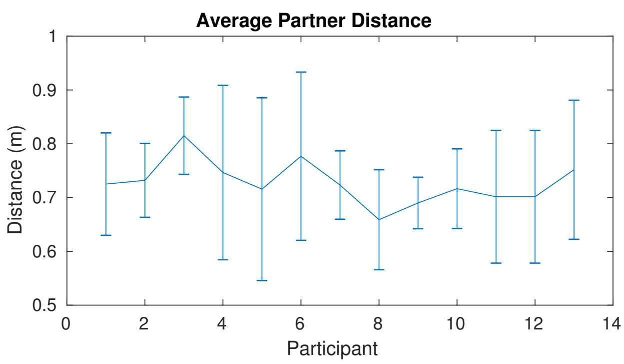


Figure 6-3: Partner Distance.

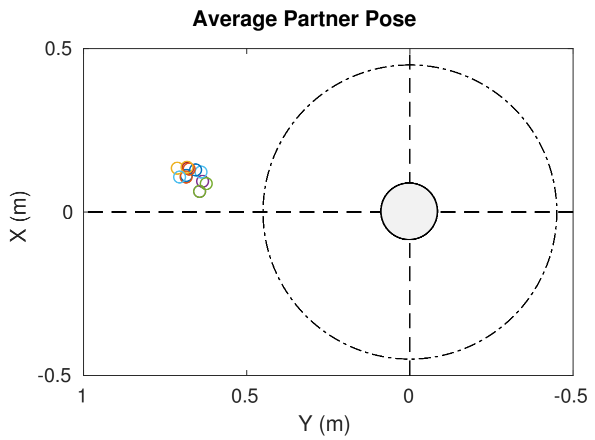


Figure 6-4: Average pose for all participants. Pepper shown as gray circle and *Personal Space* shown with dashed lines.

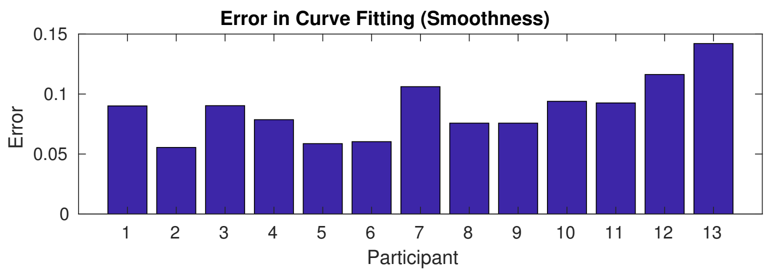


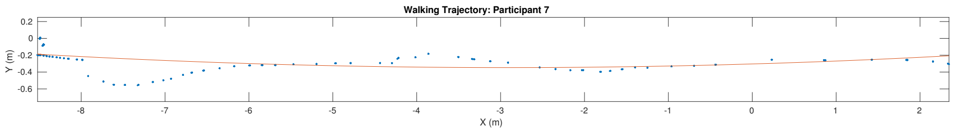
Figure 6-5: Robot Smoothness Trajectory.

- 7 people would rather walk in the same position they had when the experiment began, without shifting to the left.
- 4 people preferred to walk on Pepper's side but slightly positioned behind Pepper.
- 2 people would have walked in a front-back pattern.

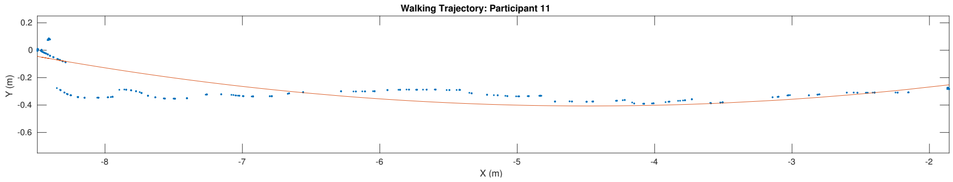
Moreover, the trajectories executed by Pepper were fitted to a second order polynomial to evaluate *Path Smoothness*, as shown with two examples in Figure 6-6. The results are presented in Figure 6-5 with the sum of squared errors for the fitting curve. These values are low in general, suggesting the *smoothness* characteristic of the trajectories was achieved.

Pepper's acceleration was also recorded to analyze the *jerkiness* of the trajectories. Figure 6-7 shows the average acceleration of each trial, with variation around the mean value shown as error bars. Overall, accelerations are minimal through all tests, but in particular circumstances Pepper suddenly stopped during the trial which led to a higher value of change, as shown in Participant 4.

Finally, the Questionnaire and post-interview provided information about the how *intuitive* were Pepper's motions according to the participants. As seen in Figure 6-2, at least 60% of the users said that Pepper's motions were clear to follow (Q10) and 70% of them liked Pepper's navigation (Q11). Those people who responded negatively to both questions felt that Pepper's speed was the aspect that made following the robot an awkward activity.



(a) Participant 7



(b) Participant 11

Figure 6-6: Example trajectories with curve fitting approximation.

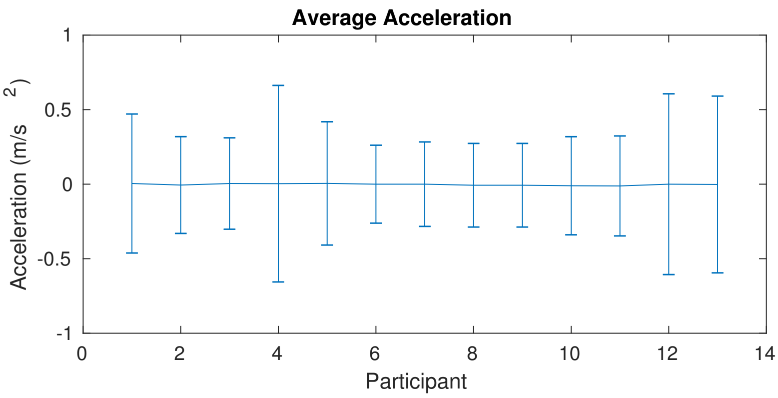


Figure 6-7: Robot Acceleration.

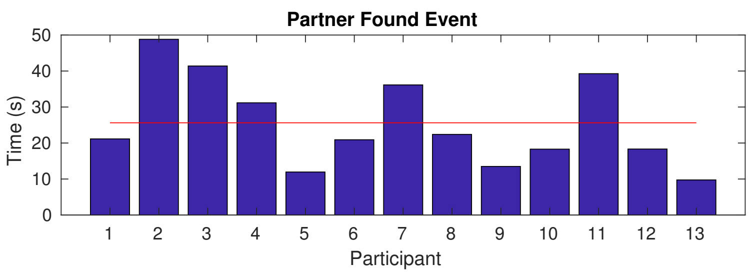


Figure 6-8: Time lapse between Pepper losing track of participant and reengaging with participant.

6-1-3 Partner Tracking

The last aspect we evaluated was the respond to Pepper's *Lost and Found* events. In Figure 6-8 the time between these two events is presented for all participants. The mean value of all experiments rounds up to 25.6 seconds, but the response is quite diverse. Several of the participants intuitively positioned themselves in front of Pepper at a close range, allowing a faster detection. Other participants remained far away from Pepper, such that no face recognition was prompted. These participants then proceeded to seeks Pepper's attention in order to continue the task. These results are confirmed with Questions 13 through 15, seen in Figure 6-2, where more than 75% of the participants graded positively Pepper's reaction. Those participants who responded in a negative manner felt that Pepper should have been able to recognize them again without them approaching Pepper again.

6-1-4 General Feedback

As part of the post-interview, the participants were asked to provide additional feedback on the applicability of Pepper as a guiding robot. Their answers suggest that a guiding robot would be better used in more complex buildings, like hospitals, exhibitions centers, crowded arenas, libraries, museums etc.

Finally, participants were also questioned on specific details that they enjoyed about Pepper and ways that Pepper could improve as a guiding robot. Their responses are summarized as follows:

Positive

- Pepper is friendly, comfortable, polite and makes good company.
- Humanoid motions are interesting.

Improvements

- Higher speed is necessary.
- Communicate more.
- Pepper's design is a bit *creepy.*

6-2 Conclusions

The results obtained from the experiment show a positive review about the characteristics of the behavior built with Pepper. Although the expected number of participants was no achieved, the feedback obtained from the users had a clear tendency, which validates the results presented in this report. The general opinion about the level of comfort that Pepper generated while guiding the participants was really positive. From one perspective, the trajectories generated by the motion planner were comforting and easy to follow. The only inconvenience with these trajectories was related the low speed at which Pepper traversed the corridor. Furthermore, Pepper's interactive gestures were liked by the participants. Most of the application users felt that hand gestures and verbal expressions improved the activity performed with Pepper.

One of the biggest challenges to implement Pepper's guiding task was to efficiently track it's Partner through out the entire behavior. Adding new hardware on Pepper was not a feasible solution from connectivity and functionality perspectives. Furthermore, the original idea behind the project was to use Pepper without any modification. The lasers provided a possible solution, but this prompted the need to influence the users to walk in a specific pattern to optimize tracking. This requirement caused the design of the project to introduce *partner engagement* to suggest the users of the robot to move in a certain direction. This strategy was also applied in the end of the task, when Pepper presents the target room. The interesting result was that by adding verbal and non-verbal indications we could influence the user of the application to comply to our requirements. Although some participants felt a bit uncomfortable with this specification, most people had no problem with complying with Pepper. Nevertheless, body motions had to be carefully designed to avoid misinterpretations.

Moreover, recall that different studies have suggested that the distance between a robot and the person who follows is around 1.2 m during a guiding behavior. The spatial definitions obtained from Proxemics Theory, used commonly in socially-aware navigation, also considers this distance to be correct as it falls outside the personal space of a person. The results from our experiment differ from these notions, suggesting, as many researchers have noted, that spatial configurations can easily change due to different circumstances. Therefore, a more adaptable approach to social navigation seems to be the key characteristic of this field of research. The application of the *Modified Extended Social Force Model* (MESFM) tries to react to this characteristic, but fails to do so as it also depends on a fixed distance between the robot and the person.

On this subject, the *Social Force Model*, as other Potential Field methods, are known to lack *smoothness* in their trajectories with noticeable changes in speed. The results of our experiments hinted a different outcome. Nevertheless, we have to consider that Pepper does not make many turns while navigating, as the experiment is carried out in a corridor, and accelerations were limited by design, since Pepper could only reach a low maximum velocity.

Additionally, socially-aware methods, in a majority, rely only on the trajectories to generate human-robot interaction, but as we tried to show with this project, HRI can also be improved through other aspects like verbal expressions and gestures. One could argue that humans generally show these characteristic to indicate future motions which are easily understood by other pedestrians. We do not propose to enforce a certain reaction on nearby people, like pushing them away from the path of the robot, but rather influence their decision through positive feedback. For example, pointing to an interesting feature around the environment or grabbing their attention through fascinating information could momentarily be used to persuade an specific motion from people.

This idea was partially perceived with the `searching` mode triggered when Pepper lost track of the Partner. Most participants showed the natural decision to reengage with Pepper when they saw that it was looking for them. Their initial approach was to put their face in a range where Pepper could detect them easily, even though this was never explicitly mentioned before the experiment. A similar reaction was observed when Pepper pointed it's arm to the target room and the participants would immediately look for the room number they said in the beginning of the experiment.

On a different topic, the idea of presenting state-of-the-art algorithms to wider range of public was one of the motivations behind this project. One response that many participants added to the feedback of the experiment was the excitement they felt about using a robot for a normal behavior like guiding a person. Even though most of the participants have a technology/engineering background, their experience with humanoid robots is very limited. Projects build on more accessible robots, like Pepper, can improve this lack of interaction. In the same manner, this objective could increase the understanding of human reactions and might provide easier solutions to introduce humanoid robots in daily life activities.

Nonetheless, Pepper did present several limitations on building this application. These restrictions were worsened with the modest documentation available on software design or base-algorithms used in Pepper.

As a final remark, user feedback was a key component of the evolution of this project. Unfortunately, this resource was used at later stages of the implementation. Even so, this feedback helped to fix many incorrect assumptions made originally in the design. For example, people rarely looked directly to Pepper's eyes while walking along the hallway. This characteristic was used in the beginning of the project to confirm the presence of the Partner, but face recognition was never successful while walking with Pepper.

Appendix A

Pepper

A-1 Head Motion

The motion range for the head is important for the development of the project, since the stereo-camera setup and 3d camera are located in the eyes of Pepper. Figure A-1 shows the complete range of movement for this body part. The speed of the head's motion is configurable through software.

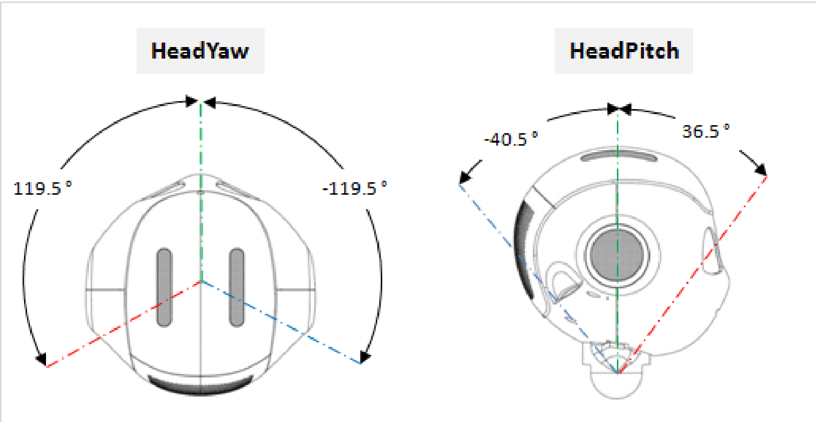


Figure A-1: Range of motion for Pepper's head. [1].

A-2 Cameras

Two identical cameras are located in the top and bottom place of Pepper's head. The relevant characteristics of these sensors are summarized in Table A-1. The depth camera is the commercially available One ASUS Xtion model. The important characteristics of this sensor are summarized in Table A-2.

Camera	Model	OV5640
	Type	SOC Image Sensor
Imaging Array	Resolution	5Mp
	Active Pixels (HxV)	2592x1944
Output	Camera output	640*480@30fps
		2560*1920@1fps
	Data Format	YUV422 color space
View	Field of view	68.2°DFOV (57.2°HFOV,44.3°VFOV)
	Focus range	30cm $\sim$ infinity
	Focus type	Auto focus

Table A-1: General specification of Pepper's stereo cameras [1].

Camera	Model	ASUS XTION
	Type	SOC Image Sensor
Imaging Array	Optical format	1/2 inch (5:4)
	Active Pixels (HxV)	1280x1024
Output	Camera output	320*240@20fps
	Data Format	Depth color space (mm)
View	Field of view	70.0°DFOV (58.0°HFOV,45.0°VFOV)
	Focus range	40cm $\sim$ 8m
	Focus type	Fixed focus

Table A-2: General specification of Pepper's depth camera [1].

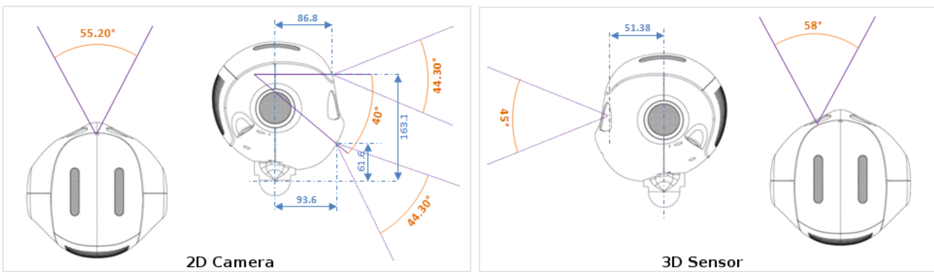


Figure A-2: Top/Bottom cameras and 3D sensor located in Pepper's head [1].

A-3 Motor Drive

The motors attached to the wheelbase have the characteristics detailed in Table A-3. These wheels are built in the configuration shown in Figure A-3.

Parameters	Values
Type of motor	BLDC (Brushless DC)
No load speed (rpm)	6110
Torque constant (mNm/A)	36.9
Stall torque (mNm)	820
Max Continuous torque (mNm)	130

Table A-3: Motor drive technical description [1].

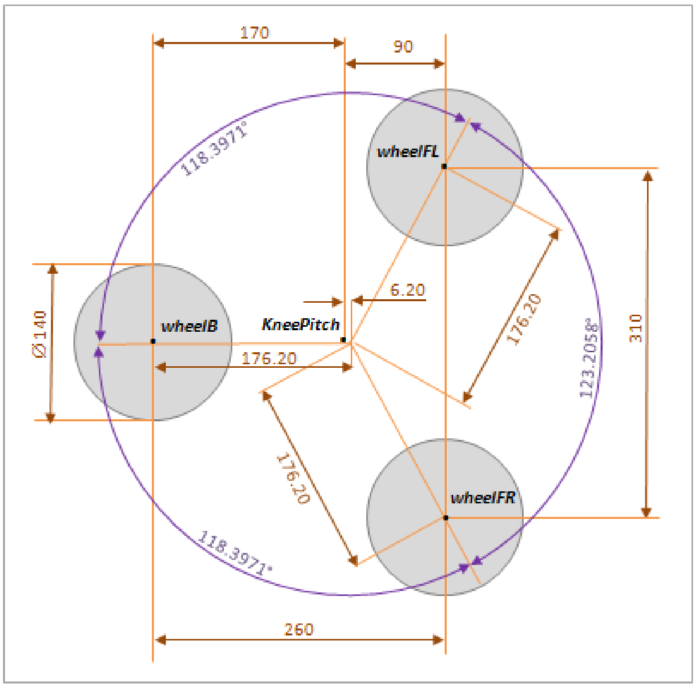


Figure A-3: Wheel base configuration [1].

A-4 NAOqi Framework

A-4-1 ALMotion

This API is frequently use in the `ROS Naoqi Bride`:

- Stiffness control API
  - ALMotionProxy::wakeUp
  - ALMotionProxy::robotIsWakeUp
  - ALMotionProxy::setStiffnesses
  - ALMotionProxy::getStiffnesses
- Joint control API
  - ALMotionProxy::setAngles
  - ALMotionProxy::changeAngles
  - ALMotionProxy::getAngles
- Locomotion control API
  - ALMotionProxy::move
  - ALMotionProxy::moveToward
  - ALMotionProxy::moveTo
  - ALMotionProxy::stopMove
  - ALMotionProxy::getMoveArmsEnabled
  - ALMotionProxy::setMoveArmsEnabled
- External-collision avoidance API
  - ALMotionProxy::getExternalCollisionProtectionEnabled
  - ALMotionProxy::getOrthogonalSecurityDistance
  - ALMotionProxy::getTangentialSecurityDistance
  - ALMotionProxy::setExternalCollisionProtectionEnabled
  - ALMotionProxy::setOrthogonalSecurityDistance
  - ALMotionProxy::setTangentialSecurityDistance

A-4-2 ALTracker

This module allows Pepper to track an specific feature of interest with different movement configurations. The aim of this module is to keep the target feature in the center of the cameras at all times. There are 4 modes of tracking with Pepper:

- Head: The two head's joint are controlled to track the target.
- WholeBody: The robot adapts its posture to track the target.
- Move: robot moves in order to keep a defined distance to the target.
- Navigate: The robot moves in order to keep a defined distance to the target.

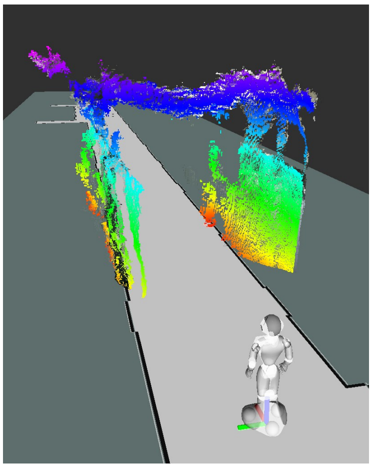
As many of Pepper's capabilities, this module is heavily dependent on the environmental conditions during operation. One characteristic that was noticed since the beginning of the trials with Pepper was the scarce detection of targets under different lighting conditions. This module is directly affected from this condition. As it was mentioned in the ALNavigation module, there is no option for the developer to modify anything from the detection algorithm and, therefore, optimal results from this module is also restricted. Further analysis will be presented on later section of this report.

Appendix B

Experiment Handout

Socially-Aware Navigation: Guiding Robot

Pepper was designed as a friendly robot companion able to understand human emotions. Among a broad range of qualities, Pepper can recognize your face, speak and hear you.



Navigation with a Robot Guide

Navigating autonomously in a human-like manner is a challenging task for a robot. In this project we programmed Pepper to navigate autonomously while it guides a visitor to a target location.

Your role …
As participant of this experiment, you will play the role of a *visitor* of the office floor and *Pepper* will guide you to your desire room. The *instructor* will supervise the entire experiment following the general procedure described ahead:

1. Face Learning: Stare into Pepper's eyes when the instructor tells you to.
2. Face Learned: The instructor will confirm that Pepper learned your face.
3. Standby: The instructor will inform you when the application is running and Pepper is ready. Please stand next to Pepper while waiting.
4. Start behavior: Pepper will recognize your face and ask you which room are you looking for. Choose a number between 1 and 24 and tell it to Pepper.
5. Guiding: Pepper will command the rest of the behavior from this point.
6. Finishing: Pepper will return to the initial position once it reaches the final target. This is the end of the experiment.
7. Questions: Complete the questionnaire that you are provided at this point.
8. Reward: A small gift is waiting for you, as a token of our appreciation.

Important: Pepper has an Emergency Button in the back of the neck. If pressed, this button will stop Pepper immediately.

Appendix C

Questionnaire

Filled by Instructor:	Participant Number:	Time :	Date :

Filled by Participant:	Occupation :	Age :	Gender: M / F

Socially-Aware Navigation: Guiding Robot Pepper

User Engagement

During the experiment, Pepper uses verbal and non-verbal (body) expressions. Based on these events, please answer the following questions:

1. Pepper's walking instructions at the start were clear to me.

Strongly disagree ○ — Disagree ○ — Neither agree nor disagree ○ — Agree ○ — Strongly agree ○

2. Pepper's indication of the target room was clear to me.

Strongly disagree ○ — Disagree ○ — Neither agree nor disagree ○ — Agree ○ — Strongly agree ○

3. I liked Pepper talking to me from the start of the behavior.

Strongly disagree ○ — Disagree ○ — Neither agree nor disagree ○ — Agree ○ — Strongly agree ○

4. I liked Pepper's remarks of our progression towards the room.

Strongly disagree ○ — Disagree ○ — Neither agree nor disagree ○ — Agree ○ — Strongly agree ○

5. I liked how Pepper presented the room and said goodbye.

Strongly disagree ○ — Disagree ○ — Neither agree nor disagree ○ — Agree ○ — Strongly agree ○

6. Bodily expressions helped to understand and follow Pepper.

Strongly disagree ○ — Disagree ○ — Neither agree nor disagree ○ — Agree ○ — Strongly agree ○

7. Verbal expressions helped to understand and follow Pepper.

Strongly disagree ○ — Disagree ○ — Neither agree nor disagree ○ — Agree ○ — Strongly agree ○

Trajectory Comfort
Based on Pepper's motions, please answer the following questions:

8. I liked the distance Pepper kept while guiding .

Strongly disagree ○ Disagree ○ Neither agree nor disagree ○ Agree ○ Strongly agree ○

9. I liked to walk on the left side of Pepper.

Strongly disagree ○ Disagree ○ Neither agree nor disagree ○ Agree ○ Strongly agree ○

10. It was clear to me where Pepper was going .

Strongly disagree ○ Disagree ○ Neither agree nor disagree ○ Agree ○ Strongly agree ○

11. I liked how Pepper moved through the corridor.

Strongly disagree ○ Disagree ○ Neither agree nor disagree ○ Agree ○ Strongly agree ○

Partner Tracking:

Did Pepper indicate to you that Pepper lost you while guiding you? Yes / No
If so, please answers the questions below in this section:

12. Pepper knew that I was following and reacted to my presence.

Strongly disagree ○ Disagree ○ Neither agree nor disagree ○ Agree ○ Strongly agree ○

13. I liked the way Pepper indicated that it lost track of me.

Strongly disagree ○ Disagree ○ Neither agree nor disagree ○ Agree ○ Strongly agree ○

14. I liked how Pepper reacted <u>after</u> it lost track of me.

Strongly disagree ○ Disagree ○ Neither agree nor disagree ○ Agree ○ Strongly agree ○

15. I liked how Pepper re-established contact.

Strongly disagree ○ Disagree ○ Neither agree nor disagree ○ Agree ○ Strongly agree ○

Filled by Instructor:	Participant Number:	Time :	Date :

Post - Interview

Without Pepper’s initial instructions, would you have walked at Pepper's side? How would you start following Pepper? Would you change positions while walking?

Pepper's comments were designed to avoid conversations. Would a conversation with Pepper make your experience more pleasant? If not, how would this interaction be more enjoyable?

Pepper pointed to the room when you reached the target location. Would you expect Pepper to do something else? If yes, how would you improve it?

Pepper uses predefined verbal expressions and corporal motions. Were these characteristics pleasant to see? Would you modified them in any manner?

Did you feel uncomfortable with Pepper's motions? What caused this?

Normally, there would have been no-one waiting for you at the entrance and you would have had to find your location yourself. Did you like that Pepper was there to help you? If not, do you think in other place it would be more helpful?

What did you like about Pepper as a guide? How can we improve our Pepper guide?

Appendix D

Mapping and Localization

D-1 Gmapping Modified Parameters

Parameters	Definition	Default	Pepper
map_update_interval	Time lapse between updates to the map (sec).	5.0	0.1
minimumScore	Minimum score for considering the outcome of the scan matching good.	0.0	100
linearUpdate	Process a scan each time the robot translates this far.	1.0	0.1
angularUpdate	Process a scan each time the robot rotates this far.	0.5	0.1
particles	Number of particles in the filter.	30	100
maxRange	The maximum range of the sensor.	-	1.5
maxUrange	The maximum usable range of the laser.	-	1.2

Table D-1: SLAM-Gmapping modified parameters.

D-2 AMCL Modified Parameters

Parameters	Definition	Default	Pepper
min_particles	Minimum allowed number of particles.	100	200
max_particles	Maximum allowed number of particles.	5000	8000
laser_min_range	Minimum scan range to be considered.	-	0.1
laser_max_range	Maximum scan range to be considered.	-	1.5
laser_max_beams	How many evenly-spaced beams in each scan to be used when updating the filter.	30	61
laser_likelihood _max_dist	Maximum distance to do obstacle inflation on map.	-	1.5
odom_alpha1	Expected noise in odometry's rotation estimate from the rotational component.	0.2	0.25
odom_alpha2	Expected noise in odometry's rotation estimate from translational component.	0.2	0.25
odom_alpha3	Expected noise in odometry's translation estimate from the translational component.	0.2	0.25
odom_alpha4	Expected noise in odometry's translation estimate from the rotational component.	0.2	0.25
odom_alpha5	Translation-related noise parameter	0.2	0.25

Table D-2: Modified AMCL parameters.

Bibliography

[1] A. S. Group, "Aldebaran documentation: Technical overview," 2017. [Online; accessed 08-July-2017].

[2] W. Burgard, A. B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun, "Experiences with an interactive museum tour-guide robot," *Artificial Intelligence*, vol. 114, no. 1, pp. 3 – 55, 1999.

[3] M. B. Sebastian Thrun and, W. Burgard, A. B. Cremers, F. Dellaert, D. Fox, D. Hahnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz, "Minerva: A second-generation museum tour-guide robot," *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, 1999.

[4] I. R. Nourbakhsh, C. Kunz, and T. Willeke, "The mobot museum robot installations: A five year experiment," *Proceedings of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, 2003.

[5] D. Feil-seifer and M. Matarić, "People-aware navigation for goal-oriented behavior involving a human partner," 2011.

[6] E. Pacchierotti, H. I. Christensen, and P. Jensfelt, "Design of an office-guide robot for social interaction studies," *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[7] A. K. Pandey and R. Alami, "An adaptive and proactive human-aware robot guide," *Social Robotics*, pp. 194–203, 2016.

[8] A. Garrell and A. Sanfeliu, "Cooperative social robots to accompany groups of people."

[9] G. Ferrer and A. Sanfeliu, "Robot companion: A social force based approach with human awareness-navigation in crowded environments," in *in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013.

[10] B. Zhang, T. Nakamura, and M. Kaneko, "A framework for adaptive motion control of autonomous sociable guide robot," *IEEJ Transactions on Electrical and Electronic Engineering*, pp. 786–795, 2016.

[11] Wikipedia, "Nao (robot) — wikipedia, the free encyclopedia," 2016. [Online; accessed 20-December-2016].

[12] A. S. Group, "Pepper," 2017. [Online; accessed 09-July-2017].

[13] A. S. Group, "Former naoqi framework," 2017. [Online; accessed 09-July-2017].

[14] A. S. Group, "Naoqi event index," 2017. [Online; accessed 10-July-2017].

[15] O. S. R. Foundation, "Ros about," 2017. [Online; accessed 20-July-2017].

[16] I. Aldebaran, "Alrosbridge documentation," 2015. [Online; accessed 22-July-2017].

[17] I. Aldebaran, "naoqi_driver," 2015. [Online; accessed 22-July-2017].

[18] I. Aldebaran, "pepper_bringup," 2015. [Online; accessed 22-July-2017].

[19] I. Aldebaran, "pepper_description," 2015. [Online; accessed 22-July-2017].

[20] M. Svenstrup, S. T. Hansen, H. J. Andersen, and T. Bak, "Adaptive human-aware robot navigation in close proximity to humans," 2013.

[21] P. Henry, C. Vollmer, B. Ferris, and D. Fox, "Learning to navigate through crowded environments," *Proceedings - IEEE International Conference on Robotics and Automation*, 2010.

[22] Y. Morales, T. Kanda, and N. Hagita, "Walking together: Side-by-side walking model for an interacting robot," *Journal of Humanoid-Robot Interaction, Vol. 3, No: 2*, 2014.

[23] J. Rios-Martinez, A. Spalanzani, and C. Laugier, "From proxemics theory to socially-aware navigation: A survey," *International Journal of Social Robotics*, pp. 7–137, 2015.

[24] "Gmapping," 2015. [Online; accessed 22-July-2017].

[25] "Adaptive monte carlo localization (amcl)," 2015. [Online; accessed 22-July-2017].

[26] A. S. Group, "Choregraphe suite," 2017. [Online; accessed 10-July-2017].

[27] J. Nielsen and T. K. Landauer, "A mathematical model of the finding of usability problems," in *in Proceedings of the INTERCHI '93 conference on Human factors in computing systems*, 1993.

[28] J. Nielsen, "How many test users in a usability study?," 2012. [Online; accessed 09-August-2017].

[29] R. A. Virzi, "Refining the test phase of usability evaluation: How many subjects is enough?," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 1992.

[30] J. Sauro and J. R. Lewis, *Quantifying the User Experience: Practical Statistics for User Research*. Waltham, MA: Morgan Kaufmann, 2012.