UNIVERSIDAD DE EXTREMADURA

ESCUELA POLITÉCNICA

MÁSTER EN INGENIERÍA DE TELECOMUNICACIÓN

TRABAJO FIN DE MÁSTER

Adaptive Personal Space Modelling and Human-Robot Interaction Planner for Social Navigation

UNIVERSIDAD DE EXTREMADURA

ESCUELA POLITÉCNICA

MÁSTER EN INGENIERÍA DE TELECOMUNICACIÓN

TRABAJO FIN DE MÁSTER

Adaptive Personal Space Modelling and Human-Robot Interaction Planner for Social Navigation

Autor: Araceli Vega Magro

Tutor: Pedro Miguel Núñez Trujillo

Co-Tutor: Luis Manso Fernández-Argüéllez

Resumen

En los últimos años el estudio de la navegación social de robots autónomos se ha convertido en un tema de investigación de gran importancia debido a su creciente uso en diferentes sectores de la sociedad.

Al navegar en entornos con humanos es necesario los robots tengan la capacidad de comportarse de una manera socialmente aceptable: deben reconocer a las personas y darles un trato especial. Además, deben ser capaces de adaptar su comportamiento al entorno que les rodea, ya sean espacios abiertos o pasillos estrechos. En estos escenarios, la interacción humano-robot se convierte en un tema de gran importancia, pues los robots deben tener en cuenta las decisiones humanas para un correcto comportamiento social.

Este Trabajo Fin de Máster se trata de una continuación del Trabajo Fin de Grado del mismo autor, en el que se presentaba un algoritmo de navegación social, considerando únicamente entornos con grupos de humanos presentes. En este trabajo se proponen varias mejoras para el agoritmo anterior. Entre ellas: tracking de humanos en tiempo real; la adaptación del espacio personal de los humanos al entorno en el que se encuentran; la consideración de espacios interactivos; y finalmente, la propuesta de un dominio de planificación de interacciones humano-robot (HRI).

Se han llevado a cabo numerosos ensayos que proporcionan una visión estadística del rendimiento de los diferentes métodos propuestos, utilizando para ello métricas comunes en este tipo de evaluaciones. Las distintas pruebas realizadas aportan como resultado principal una mejora notable en el comportamiento social del robot, en comparación con los resultados obtenidos con el algoritmo inicialmente propuesto.

Abstract

In recent years the study of social navigation of autonomous robots has become a major research topic due to its increasing use in different sectors of society.

When navigating in human environments it is necessary for robots to have the ability to behave in a socially acceptable way: they must recognize people and give them special treatment. In addition, they must be able to adapt their behaviour to the environment around them, such as open spaces or narrow corridors. In these scenarios, human-robot interaction becomes an important issue, as robots must take human decisions into account for proper social behaviour.

This end-of-master project is a continuation of the end-of-degree project by the same author, in which a social navigation algorithm was presented, considering only environments with groups of humans present. In this work several improvements are proposed for the previous agorithm. Among them: tracking of humans in real time; the adaptation of the personal space of humans to the environment in which they are found; the consideration of interactive spaces; and finally, the proposal of a human-robot interaction (HRI) planning domain.

Numerous tests have been carried out that provide a statistical view of the performance of the different methods proposed, using common metrics in this type of evaluation. The main result of the different tests performed is a notable improvement in the social behaviour of the robot, in comparison with the results obtained with the initially proposed algorithm.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

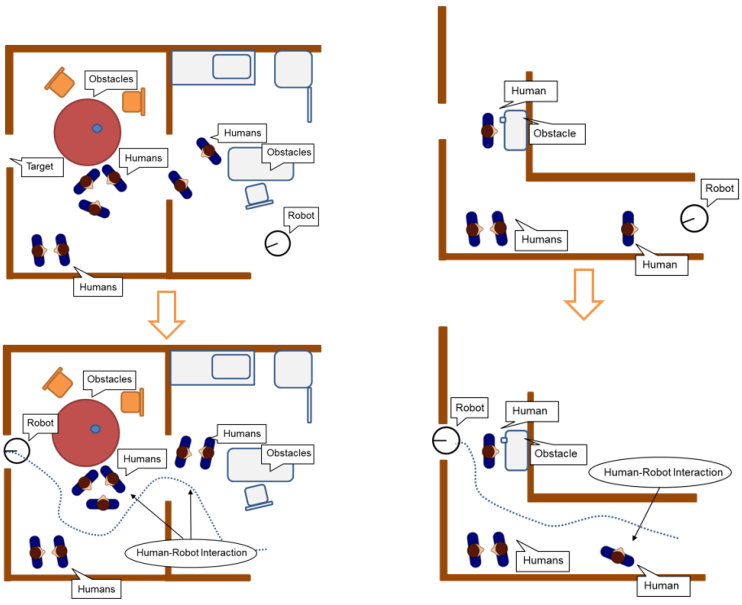
Social robotics is a growing area whose benefits are beginning to be reflected in society. Nowadays it is common to use assistance robots in different environment such as schools, museums, airports, or even homes. In these scenarios humans cohabit with robots, reason why the robot needs to carry out its daily activities in a socially accepted way.

In recent years, the concept of *Socially Acceptable Navigation* has become a topic of great interest among the scientific community. Social robotics extends the definition of mobile robots to human-populated environments. To achieve a proper social navigation, certain non-verbal rules must be respected, especially those related to the personal space of humans. These rules are studied in *proxemic* (i.e., relationships between distances and type of interaction), term introduced by Edward T. Hall in [2]. Social robots must be able to generate different socially accepted routes as well as respecting the personal space of humans, showing socially acceptable behaviors and interacting with them in a friendly manner. For this reason, the study of Human-Robot interaction (HRI) to analyze the different aspects inherent to the interactions between humans and robots, becomes crucial in Social Navigation.

Currently, one of the main problems of Social Navigation is indoor navigation. This problem has traditionally been addressed from *Metric Maps* built by the robot from its own sensors. As an alternative, the research evolved to the use of semantic information

of the environment, *Semantic Maps*, which allowed the navigation of the robot through high level entities within the environment (go to the kitchen, look for where the dining room is). The problem of social navigation establishes a union of both systems, metric and semantic, giving the first semantic information and relating these with the criteria for a navigation accepted by humans. It is what is known as *Social Mapping*: forbidden navigation zones or penalized for disturbing during a conversation or for interrupting the visual field of the human towards a certain object. The concept of social mapping was introduced in [3].

Building a map of the environment is not the only problem that social navigation must solve. In order to achieve socially acceptable navigation, it is also necessary to take into account a large number of procedures, such as path planning or robot's localization. Solving all these problems in a end-of-master project is impossible. For this reason, this work focuses on the development of a social indoor navigation system for human populated environments that also includes the planning of human-robot interactions.



(a) Office environment composed of two rooms

(b) Narrow corridor

Figure 1.1: Examples of scenarios in which robots have navigate using social rules.

This work is based on the social navigation system proposed in [1]. The system was developed in the Robotics framework Robocomp, created by Robolab (University of Extremadura). This algorithm proposed a *Social Map* of the environment, establishing the personal space of a human, clustering interacting people and defining regions where navigation was forbidden. It was tested in simple environments, where there were no obstacles near the human. However, in real and complex environments, such as an office or a home, there are certain situations that must be taken into account when developing a realistic human-centred navigation system. This work proposes the improvement of the original social navigation algorithm in many aspects:

When a robot navigates in human-populated environments, a data acquisition system capable of capturing human-related information in real time is required. In addition, human information is not fixed, as they move freely around the environment. The system must be able to track the human during this movement, so that for any instant of time, its pose (i.e., position and orientation) is estimated. As a first improvement, this work proposes the use of Astra RGBD sensors for the recognition and tracking of people.

Once humans have been detected, this work proposes two methods to achieve a social navigation system adapted to the spatial context, in order to avoid the robot being blocked in certain scenarios, such as narrow corridors. First, an algorithm to deform the personal space according to the environment in which the person is located (corridors, small or large rooms) is presented. Secondly, an algorithm that defines different interactions spaces around humans and modifies the robot behavior according this areas is proposed.

It has been taken into account that there may be situations in which humans do not interact with each other, but with objects in the environment. In these cases, the robot must also behave in a social way, avoiding interfering in the interaction. For this reason, interactive spaces are also considered in this work, including the concept of *Spaces Affordance* [4], areas located around objects with which people usually interact. The robot must respect these spaces and avoid them if humans are interacting with the

object (*Activity Spaces*).

In Fig. 1.1 are shown different situations where the robot must navigate in human-populated scenarios in which there are blocked areas in the routes planned. In this situations the robot must approach the person, or group of people, and interact with them to ask for permission or collaboration to pass. As a final contribution, this works proposes a method that allow robots to have the ability to develop human-robot interactions (HRI) when navigating in human-populated environments. The domain of action that must have the navigation component is defined in this work.

Chapter 2

Goals

This work is based on the hypothesis that it is possible to improve the human experience with an assistance robot in real environments with a robust social navigation system and a human-robot interaction planner for human-centered navigation. Therefore, the main objective of this work is to advance in the line of social navigation of autonomous robots in real environments with humans.

There are different challenges to carry out this goal, reason why in the develop of this work, a set of specific objectives has been necessary to achieve:

- To adapt of the algorithm proposed in [1], basis of this work, to the changes suffered during the last year in the cognitive architecture CORTEX used in the Robotics framework Robocomp.
- To detect and track of people in real time, in addition to insert in the robot cognitive architecture their pose in order that the data of the person can be used by other agents within RoboComp.
- To compare different methods to give flexibility to the personal space, in order to avoid the robot being blocked in certain situations.
- To consider interactive spaces by introducing the concept of Spaces Affordances, areas that the robot should avoid due to the fact that people usually perform interactions inside.

- To propose a human-robot interaction planning system to fill the gaps of the model. For example, to ask for permission when a person blocks the path.
- To validate the system with different tests and experiments.

Chapter 3

Related Work

Robot navigation in crowded environments has been extensively studied in the last years and several theories and methods have been proposed since then. Particularly interesting reviews have been presented in [5, 6] and more recently [7]. Classic social navigation paradigms are based on using well-known navigation algorithms, and therefore adding social conventions and/or social constraints. Different works such as [8, 9], have shown that the same proxemic zones that exist in human-human interaction can also be applied to human-robot interaction scenarios. A broad survey and discussion regarding the social concepts of proxemics theory applied in the context of human-aware autonomous navigation was presented in [6].

Most works based on proxemics, do not take into account that these social zones often depend on the human intentions and the environment. This idea was briefly described in [10], where authors extended the previous work [11] and suggested a skew-normal probability density in order to model the social space.

The work proposed goes further and proposed two methods of adapting the personal space to the environment. The first one defines a new spatial density function that extends the previous work [1] in order to satisfy these real scenarios. A flexible mathematical model based upon the use of a modified two-dimensional Gaussian function [12] is proposed to dynamically model the personal space of an individual adapting its shape to the spatial context. The second method proposed in this work is

the definition of three spaces around the person: intimate, personal and social space; in order to plan the path of the robot trying to according these spaces. This idea was introduced in [6].

Regarding interactive scenarios (*i.e.*, spaces in which people and buildings engage in a mutual relationship) some authors define regions next to objects in which robot navigation is forbidden. The space affordances are defined in [4] as potential activity spaces, which are social spaces constituted by means of actions performed by humans. A similar idea was previously introduced in [13, 14], where these areas are considered by means of a spatial Poisson process that allows encoding the probability of human activity events. In [15] a concept similar to space affordances is considered by taking into account areas frequently visited in the environment. This work uses a similar region as the introduced in [4] and updates and adapts the robot's navigation plan according to this information. Unlike the proposal in [4], this work defines the forbidden region for robot navigation as polylines, which are then used to update the free space graph and to adapt the path planned by the robot during the navigation.

When a robot plans the best path in human-populated environments, it must avoid passing between two people talking or getting inside the field of view of the people when they are observing a particular object. Social mapping is an interesting concept recently introduced in the robotics community in order to manage the shared space between humans and robots. The concept of *Social mapping* was introduced in [3]. It deals with the problem of human-aware robot navigation and considers factors like human comfort, sociability, predictability, safety and naturalness [5]. More recently, the concept of *behavioral mapping* has been introduced in [16], where the authors extend social mapping to a behavioral model acting as a mediator that facilitates seamless cooperation among the humans. In [1], the basis of this final master project, a model of social map was proposed, defining a proxemic-based adaptive spatial density function for clustering groups of people and define forbidden spaces.

As the number of skills robots have increases, human-robot collaboration becomes more feasible. In [17], the requirements for effective human-robot collaboration in

interactive navigation scenarios are listed. Additionally, authors present three different human-robot collaborative planners. However, they only focus on secure navigation and not on HRI. Other works such as [18], anticipate the human trajectory in order to update social constraints during robot navigation. Similar works are presented in [19]. Again, authors do not take into account interaction with people for robot navigation. Planning for HRI has been used in manipulation tasks [20] and task allocation in collaborative industrial assembly processes. However, there are no works where HRI has been used to improve robot navigation in crowed environment using social conventions. The proposed work introduces a planning domain for social navigation where HRI is crucial for solving real situations where the robot's path is blocked due to social limitations. The goal is for the robot to execute actions that optimize social navigation and human satisfaction.

Chapter 4

System Overview

4.1 RoboComp

This work has been developed in Robocomp, an open-source Robotics framework that allows an easy and fast software development. Robocomp provides the tools to create and modify software components that communicate through public interfaces.

RoboComp is based on Ice (*Internet Comunication Engine*) [21]. It provides an extension of the middleware in several aspects, such as the tool set and classes that Robocomp facilitates. Some of these tools, such as the component generator, the manager, the monitor or the simulator have been very helpful in this work.

The components of Robocomp are created using two domain languages: IDSL and CDSL. With IDSL is defined the inferface of the component and with CDSL is specified how the component will comunicate with the rest of the components. Based on this information, the code is generated, in C++ or Python languages.

One of the main advantages of Robocomp is that the components can be regenerated easily without the loss of the user's code, if it is needed to change their features.

The CORTEX Cognitive Architecture is implemented in this framework. This architecture, which is explained bellow, is used in the social navigation algorithm in which this work is centered.

4.2 Cognitive Architecture

It is important to know what is the cognitive architecture CORTEX, presented in [22], in order to understand the proposed work.

The robotics cognitive architecture CORTEX is defined as a network of cooperative *software agents* connected through a *shared representation* shown in Fig. 4.1. This shared representation was defined in [22] as *"a directed multi-labelled graph where nodes represent symbolic or geometric entities and edges represent symbolic or geometric relationships"*.

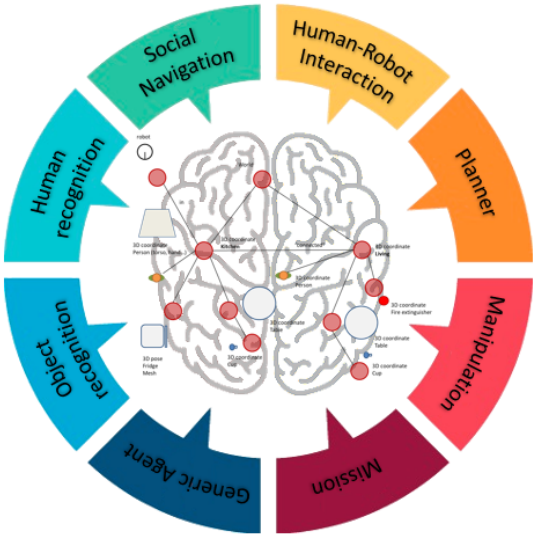


Figure 4.1: Diagram of CORTEX with the main software agents involved in this work. The shared representation of the environment is represented in the centre.

In the proposal of a improvement for the Social Navigation algorithm, many CORTEX agents are involved. First, in the higher layer of the architecture the robot must have the capability of detecting objects in the path and updating the symbolic model accordingly. Additionally, the skill of detecting and tracking humans is also mandatory because robots need to know about humans to get commands, avoid collisions and provide feedback. Also, a conversational agent is required in order to interact with humans. The final, and most important agent for social navigation, is the

one implementing the navigation algorithms that allows robots to navigate in a social manner.

The concept of *deep representations* was initially described by Beetz et al. [23] and it advocates the integrated representation of robot's knowledge at various levels of abstraction in a unique, articulated structure such as a graph. Based on this concept, a shared representation, *Deep State Representation (DSR)*, to hold the robot's belief as a combination of symbolic and geometric information, is proposed in [22] and used in this work. This graph represents knowledge about the robot itself and the world around it in a flexible way.

4.3 CORTEX Agents

An agent within CORTEX was defined in [22] as *a computational entity in charge of a well-defined functionality, whether it be reactive, deliberative of hybrid, that interacts with other agents inside a well-defined framework, to enact a larger system*. In CORTEX, agents define the classic skills of cognitive robotics architectures, such as navigation, manipulation, person perception, object perception, dialogue, reasoning, planning, symbolic learning or executing. These agents operate in a goal-oriented regime and their goals can come from outside through the agent interface, and can also be part of the agent normal operation. Next, a description of the main software agents involved in the social navigation algorithm is shown. This description of CORTEX and its agents is based on the original proposal, before the applying the methods described in this work.

4.3.1 Human detection and representation

The person detector agent is responsible of detecting and tracking people in the environment. The presence of humans in the robot's path or in their environment may determine changes in the navigation route in order to make it socially acceptable.

Originally, people were included in the model through a component named

FakeHuman. This component included the human in the symbolic model given a person's position and rotation. It also allowed to move the human through a interface. At the end of this work, a real time human pose detection and tracking algorithm is provided.

4.3.2 Human-Robot Interaction

The conversation agent performs speech-based human-robot interaction. In social environments, HRI provides tools to the robot and/or human to communicate and collaborate. Automatic Speech Recognition and Text-to-Speech algorithms allow robot to send and receive information to/from humans during its social navigation.

4.3.3 Executive

The Executive is responsible for computing plans to achieve the current mission, managing the changes made to the DSR by the agents as a result of their interaction with the world, and monitoring the execution of the plan. Each time a structural change is included in the model, the Executive uses the domain knowledge, the current model, the target and the previous plan to update the current plan accordingly.

4.3.4 Social Navigation

The Social Navigation Agent is the most important in this work. It is in charge of analyzing the environment and calculate socially acceptable routes for the robot.

The original social navigation algorithm, presented in [1], was divided in three fundamental steps: the modelling of personal space, the grouping of individuals and the modification of the navigation architecture to achieve a socially acceptable navigation. An overview of current navigation framework is shown in the Fig. 4.2.

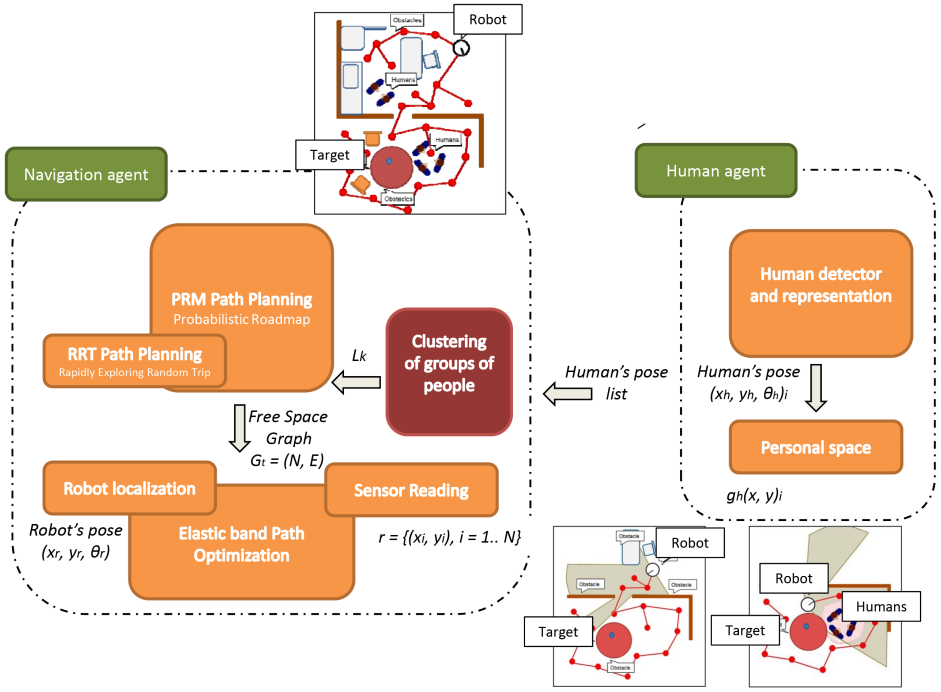


Figure 4.2: Overview of the social navigation framework

- *Personal Space Modelling*: The social navigation proposal presented in [1] is based upon the use of an asymmetric two-dimensional Gaussian function [12] to represent the personal space of an individual. The asymmetry of the Gaussian function is used to represent the discomfort of the human when the robot circulates around him. A mathematical description of this algorithm is included in the annex A.1.

- *Clustering of people*: Once the personal space of each human has been modelled, a global density function is used in order to separate humans into groups accordingly to its distances. This method discriminates the group contour to which each individual belongs in order to define forbidden regions for the navigation. This is accomplished by using a modified version of the method described in Viera's work [24]. A mathematical description of this algorithm is also included in the annex A.2.

- *Path planner and optimization*: Originally, the social navigation architecture

uses the *Probabilistic Road Mapping (PRM)* [25] and *Rapidly-exploring Random Tree (RRT)* [26] planners. These planners employs an irregular free space graph, which represents the space free of objects in the world, in conjunction with the inner representation of the world, called *InnerModel*, in order to calculate the path for the robot. Then, a modified version of the *elastic band* algorithm for path optimization [27] is used.

Chapter 5

Methodology

The original social navigation algorithm agent was developed in order to be used in simulated scenarios. However, real scenarios are complex, and thus, the system should be robust and have the capability to operate in real time. Therefore, this work proposed a series of improvements to the social navigation algorithm presented in [1], in order to be used in real environments. Fig. 5.1 shows a block diagram of the improvements proposed.

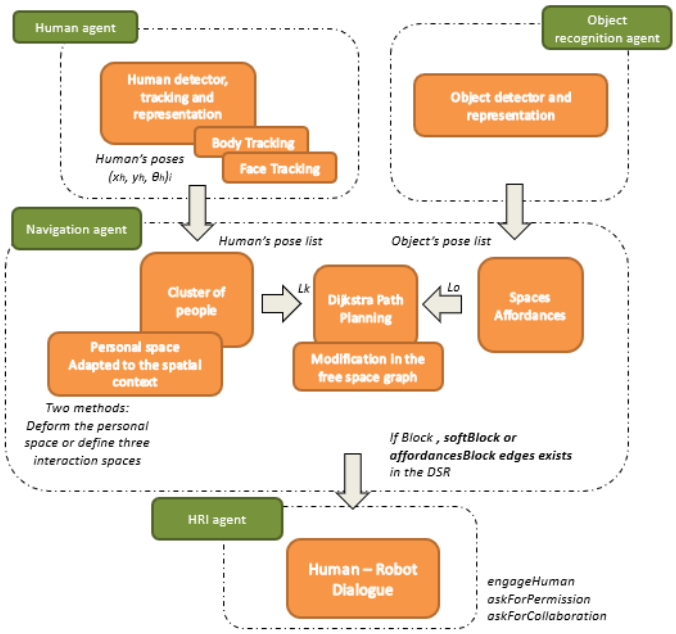


Figure 5.1: Block diagram of the methodology proposed

This chapter is organized as follows:

First of all, the main changes that CORTEX cognitive architecture has undergone over time, are explained. As explained in 4.2, CORTEX is the kernel of the social navigation system used in Robolab for their autonomous robots. After this, this chapter describes the methods and algorithms developed in this work. In first place, a method for the detection and tracking of humans in real time is proposed. This is mandatory for a robot social navigation in real environments. In addition,two different methods are proposed to adapt the personal space of a human to the social environment, since it is not the same to navigate in a wide environment than in a corridor. It is aimed at ensuring that the personal space of human beings is not fixed, but flexible according to the environment in which they find themselves. This work also includes the concept of Spaces Affordances, areas in the environment in which humans tend to develop certain interaction, as a way of adapting the environment to human activities. Finally, a domain of planning HRI is proposed, for those situations in which the robot needs to interact with the person.

5.1 Adaptation to changes in CORTEX

Robocomp is a robotics framework in continuous development. With the course of time changes have been made in the cognitive architecture of the robot and it has been necessary to adapt the social navigation algorithm presented in [1] to these changes. The main changes that have taken place are explained below:

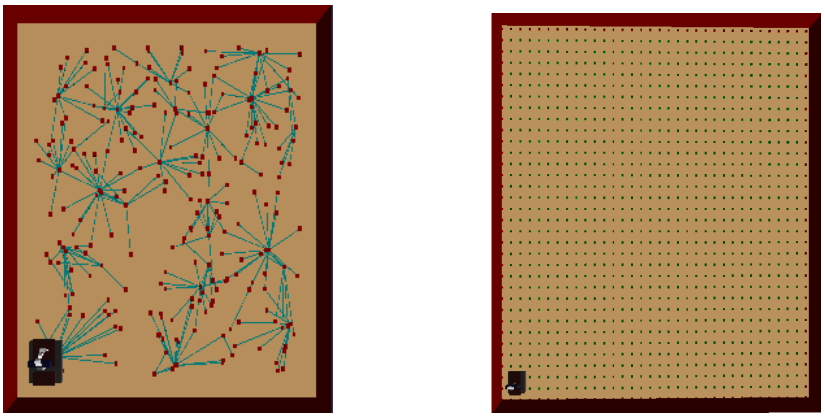
5.1.1 Free space graph

One of the most important changes that have been made to the robot's architecture is the modification of the free space graph that the robot uses for path planning.

The original graph in [1] was defined by a set of asymmetrically distributed nodes N and edges E, $G = (E,N)$, that described the free space. To consider the personal space of humans, the original graph had to be modified. It was required to remove the

edges and the nodes that crossed the personal space in order to take into account these areas in the path planning. These forbidden areas were defined as polygonal curves L_k, as is explained in the annex A. For removing the nodes, it was defined for each polyline $l_i \in L_k$ a polygon $P_i = \{a_1, ...a_m\}$, being $a_j = (x,y)_j$ the points by which the polyline was formed. All the free space nodes $N_i \in N$ contained in that polygon P_i were removed. The edges were also removed if the edge E_i intersected with the polygon P_i. That meant that the edge E_i crossed the polyline and it was necessary to remove it to prevent the planned route going through this space.

The edges and nodes removal caused some issues when the people moved because it was necessary to regenerate again the nodes and edges of the graph. This regeneration was a very slow process. In addition, when humans were inserted into the graph, unconnected areas appeared due to the disappearance of the nodes and edges. When the PRM planner found unconnected areas, the RRT planner was used, which introduced a considerable delay in the planning of the route.



(a) Previous Graph (b) Current Graph

Figure 5.2: Comparison between the current and previous free space network

Currently, the free space is represented only by a set of nodes N, $G(N)$ regularly distributed in the environment. When considering a free space graph whose nodes are distributed uniformly, there is no problem of unconnected areas, so it is no longer necessary to use the RRT planner. Each node N_i has two parameters: availability, A, and cost, C . The availability of a node, $A[N_i]$ indicates if the space is free or occupied;

and the cost, $C[N_i]$, indicates the weight of a node, i.e. what it takes for the robot to travel between nodes. Initially, $C[N] = 1$, what means that all nodes have the same cost.

To consider the personal space of humans, for each polyline $l_i \in Ł_k$, a polygon P_i is formed with the polyline points. $A[N_i]$ is set to occupied if the node N_i is contained inside that polygon P_i. If the person moves, and N_i is not longer contained in P_i, $A[N_i]$ is set to free, unless in the initial graph $G_0(N)$, in which no persons are considered, the node was originally occupied. In the Fig. 5.2 a comparison between the previous and the current graph is illustrated.

As will be explained in the following section, the Dijkstra algorithm is currently used in the path planning. This algorithm employs this novel free space graph $G(N)$. One of the main advantages of this type of graph is that each node, apart from indicating whether it is occupied or free, it has a cost. It will be very useful in the next sections, where it is explained a new method to plan a trajectory based on the personal spaces of humans.

5.1.2 Use of Dijkstra algorithm in the path planning

Another change in Robocomp's architecture has been the modification of the algorithm used to plan the robot's trajectory. As explained in the section 4.3.4, the PRM and RRT planning algorithms were previously used in the original navigation system. The main drawbacks of these path planning algorithms were the delay in the re-planning.

The Dijkstra algorithm is currently employed. This algorithm is used for the determination of the shortest path between an initial position and a target to which the robot must travel. Given a node of origin, the algorithm calculates the distance from origin to the rest of the nodes of the free space graph, explained above, taking into account the cost (weights) of the nodes. The cost of a path is the total of the cost of the nodes that conform it.

5.2 Real Time Human Detection

The first step to achieve a social navigation algorithm is the detection and tracking of humans in the environment. In the previous work [1] this was done through a component that allowed to insert the person in the model given its pose (i.e., position and rotation). This was useful in simulation, however in real environments it is necessary to detect people and track them in real time.

In real scenarios people must be detected and tracked by the sensors of the robot in conjunction with sensors placed in the environment. One of the contributions of this work is the insertion of people in the cognitive architecture CORTEX, acquiring the information using a RGBD sensor. For each detected person the agent inserts in the DSR its pose.

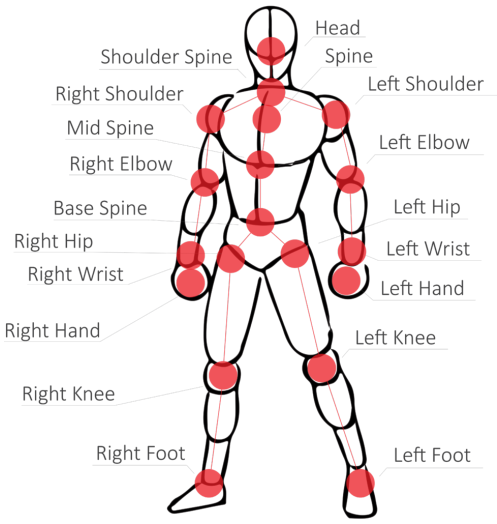


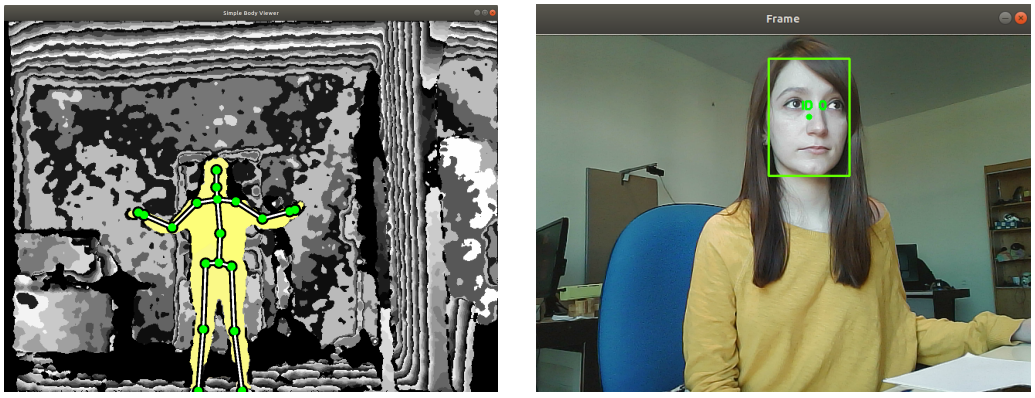
Figure 5.3: List of joints provided by Orbbec Astra SDK [28]

The sensor chosen for the detection and tracking of humans is the Orbbec Astra Pro camera. It is a 3D camera, with a range of 0.6 to 8m (Optimal 0.6 - 5m), which offers depth and RGB images with high resolution. One of the advantages of Orbbec is that it offers an extensive set of tools in its SDK (Software Development Kit) [28]. In Orbbec Astra SDK is integrated a Body Tracking software, which provides a detection and tracking of people. Orbbec Body Tracking provides access to a person's list of joints (i.e. the points that make up their skeleton). The software provided by Orbbec

is capable of detecting a large number of humans in the environment, however it can only track two skeletons at the same time. The image 5.3 shows a list of joints that Orbbec's software allows to obtain.

5.2.1 Multi-Modal Tracking

The software provided by Astra for the detection of a person (or a group of them) would lead to errors in the representation and modeling of the human pose if it is used alone in the system. This software, despite providing an appropriate detection, does not provide a very accurate tracking. For instance, when the camera detects a person, the body tracking software shows the joints of its skeleton, at the same time that assigns him an identifier (*id*), but if this person leaves the camera range and reappears, the software will give the person a new *id*.



(a) Body Tracking (b) Face Tracking

Figure 5.4: Types of tracking used for the detection of humans

In order to improve the accurate and robustness of the detection and tracking algorithm, a mixture of different types of tracking has been used. This work uses two different human detection and tracking algorithms: one of them is the 3D body tracking provided by Astra. In the Fig. 5.4a can be seen graphically the joints obtained by this algorithm. The second one is a typical 2D face detection and tracking algorithm. In the Fig. 5.4b can be seen graphically the detection of a face using this algorithm. The

information obtained with both of them is fused in order to estimate the pose of each human in the environment.

The face tracking used is implemented with OpenCV (Open Source Computer Vision Library) [29] . The face detection is done using deep learning face detector, available in OpenCV version 3.3 or higher. As a result of the detection, the bounding boxes of all the detected faces are obtained for each video frame. The centre of each bounding box is calculated and an *id* is assigned to it. If a new bounding box appears in the next video frame, its centre will be calculated and checked if it corresponds to any of the people present, calculating the Euclidian distance with the centres obtained in the previous frame. If it coincides, the *id* will be the one of the previous frame, otherwise a new one will be assigned to it.

Since each tracking employed assigns an *id* to each person detected, it is necessary to match the people detected in each one, in order to obtain a generic person *id*. Let $l_{id}^{body} = \{id_1^b, id_2^b, ..., id_n^b\}$ be the list of identifiers associated to n different humans detected (and tracked) by the body tracking algorithm. Besides, let $l_{id}^{face} = \left\{id_1^f, id_2^f, ..., id_m^f\right\}$ be the list of identifiers associated to m different humans detected (and tracked) by the face tracking algorithm. Next, a matching stage between both lists of humans is needed.

To perform the matching between the information obtained by the two tracking algorithms, let B be the set of all the bounding boxes obtained in the face detection (i.e., $B[id_j^f]$ is the bounding box of the face id_j^f). For each $id_i^b \in l_{id}^{body}$, and for each $id_j^f \in l_{idf}$ it is checked if the pixel coordinates of the joint corresponding to the head, J_H, is contained inside the limits of the bounding box $B[id_j^f]$. If it happens, it is considered that id_i^b is the same person than id_j^f.

Then, the algorithm estimates human poses as explained bellow:

Position calculation

To estimate the person's position, firstly the list of joints of its trunk is obtained, $J_{trunk} = \{J_H, J_N, J_{SS}, J_{MS}, J_{BS}\}$, being H, N, SS, MS and BS the head, neck, shoulder

spine, middle spine and the base spine, respectively. Each $J_i = [x,y,z]^{Astra}$ is the 3D coordinates of the joint in the camera's reference frame. In order to obtain the real position of the person, it is necessary to transform the position of each joint to real world coordinates $J = [x,y,z]^{World}$. For this purpose, *InnerModel* class from RoboComp is used, since it provides the necessary tools to transform points to different reference frames.

For each human h_i detected in the environment, its position $h_i[\overline{x},\ \overline{z}]^{World}$ is calculated as the mean of x and z positions of the trunk joints. It may happen that some joint is not found but at least one of the joints must be found, otherwise the human pose estimation cannot be performed. The number of joints found for each person can be interpreted as a measure of the confidence of the data obtained for that person.

Rotation calculation

Person's rotation is estimate based on the positions of their shoulders. It is necessary to obtain the joints of both shoulders ,J_{LS} and J_{RS}, (i.e, left shoulder and right shoulder), to perform the calculation. Similar to the estimation of the position, the obtained information is in the reference frame of the camera $J_i = [x,y,z]^{Astra}$. *InnerModel* is also used to transform the coordinates of the joints to the reference frame of the world $J_i = [x,y,z]^{World}$.

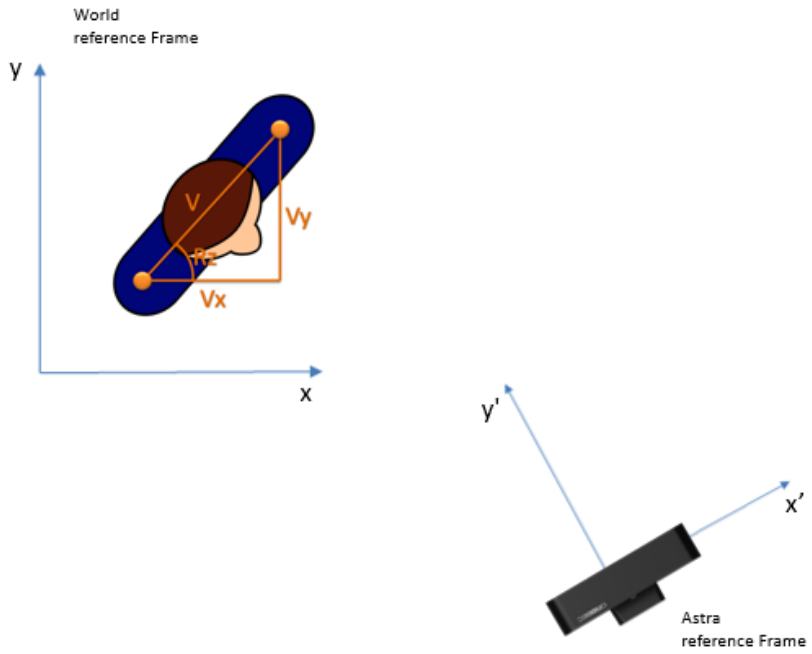


Figure 5.5: Parameters used in the rotation calculation. The different reference frames of Astra and World are also illustrated

The vector V that connects both shoulders is calculated as it is indicated in the eq. 5.1. Since the y-coordinate in the world represents height, only x and z are used for the calculation of V, being $V_i = [x,z]^{World}$. The rotation of the Z-axis is the tangent arc of the division of the Y-position of the vector V by its X-position, as it is show in eq. 5.2. In the Fig. 5.5 the parameters used for the rotation calculation and the different reference frames are graphically represented.

$$V = V_{LS} - V_{RS} \tag{5.1}$$

$$R_z = \arctan \frac{V_y}{V_x} \tag{5.2}$$

After calculating the position and rotation of a person, this information is included in the internal model of the robot (*i.e.*, DSR), in order that its information can be used by other agents of the architecture.

Once a reliable detection and tracking has been achieved, the aim is to improve the current navigation algorithm, turning it into an algorithm in which the personal space

of humans can be adapted to the environment. The improvements proposed to achieve this objective are explained in the following sections.

5.3 Space Affordances

In real-life scenarios, human often interact with objects in the environment. The robot should be able to detect these situations and behave in a social way, avoiding interfering in the interaction.

The concept of *Space Affordances* refers to areas where humans usually perform particular activities [4]. In interactive scenarios, these spaces are related to objects with which humans often interact, for example, the space near a poster or a coffee machine. These space affordances are called activity spaces when humans interact with objects. In this work, the space affordances has been defined as trapezoidal spaces around objects, which are considered forbidden for navigation when humans interact with the objects.

Let $O_n = \{o_1, ..., o_n\}$ be the set of objects with which humans usually interact in the environment, where n is the number of objects detected by the corresponding agent. It is assumed that these objects are detected by the robot's perception system [30]. Each object o_i stores the interaction space i_{o_i} as an attribute, which is associated to the space required to interact with this object, and also its pose $p_{o_i} = (x, y, \theta_i)$,

$$o_i = (p_{o_i}, i_{o_i})$$

Different objects in the environments have different interaction spaces, for instance, to use a coffee machine it is needed less space than the needed to read a poster, because it can be done from a farther distance. Next, the space affordance A_{o_i} is defined for each object $o_i \in O$. In this work, the shape of these spaces has been modeled as an symmetrical trapezoid with height a_h and widths (a_{w1}, a_{w2}), as is shown in Fig. 5.6, being $a_{w2} = (a_{w1} \cdot a_h) \ / \ 4$.

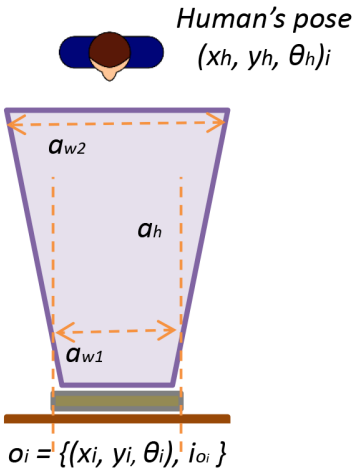
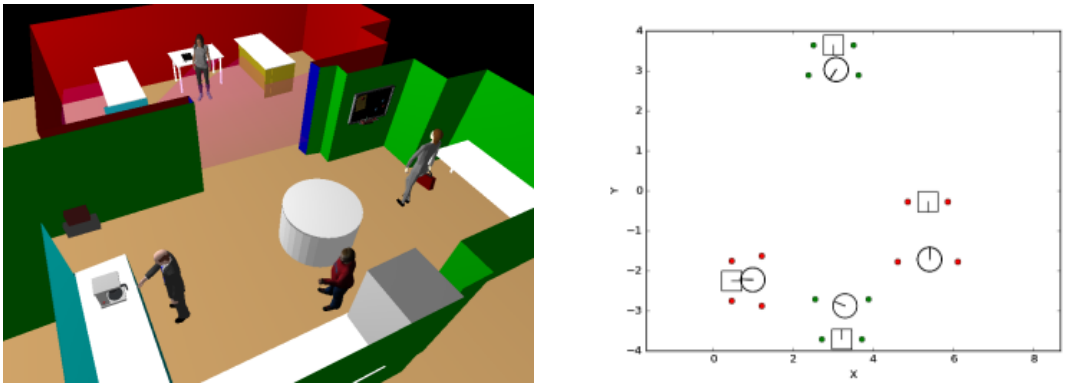


Figure 5.6: Space Affordance Modeling: The Space affordance of an interactive object is modeled by a symmetrical trapezoid.

Once the space affordance A_{o_i} is created, it is checked if is being used as an activity space, what means that the person is interacting with the object. Two conditions have to be fulfilled to consider that an activity is being carried out: the person h_i has to be inside the space affordance and has to be looking at the object o_i. The space will be forbidden for navigation if these conditions are true. Similar to the personal space described in the annex A, A_{o_i} is also modeled by a polyline that is described by four vertices v_a that will be used to delimit forbidden areas for navigation. Finally, $L_o = \{A_{o_1}, ..., A_{o_n}\}$ describes the set of polylines used by the navigation algorithm for defining forbidden navigation areas.

In Fig. 5.7a four humans in different positions and four objects are shown (a coffee machine, a fridge, a phone, and a pin board). Some of the humans are interacting with the objects. The position of the humans, the objects and the shapes of the spaces created for these objects are shown in Fig. 5.7b. The vertices v_a are shown in green if the space is being considered as free. These vertices are in green even if the person is inside the space but is not looking at the object. If v_a is shown in red means that the person is inside the space affordance and looking at the object (i.e, the person is interacting with the object), so the space is being used as an activity space and, therefore, considered as occupied.



(a) Humans interacting with different objects

(b) Spaces Affordances generated

Figure 5.7: Example of Spaces Affordances in a simulation with humans and four objects

5.4 Personal Space adapted to the Spatial Context

The proxemics in robotics, the basis of the previous work [1], has been studied mainly in large environments, where there are no obstacles in the vicinity of humans. In real situations, it is not the same to navigate in a hall than to do it in a corridor with humans. The robot should be able to adapt their personal space according to the dimensions of the available space. This could be done since there is some flexibility in the comfort space reserved for the human.

This section presents the evolution that research has undergone with the aim of achieving a navigation algorithm adapted to the social environment. In this end-of-master work, two methods are proposed to adapt the human space to the environment:

5.4.1 Environment-dependent personal space modeling

The first proposed method evaluates distances from the position of the human $h_i = (x_h, y_h, \theta_h)$ to the walls of the room in four different orientations $\theta_i = (0, \pi/2, \pi, 3\pi/2)$ being $\theta_i = 0$ the orientation of the person. Then, for each direction, the algorithm evaluates if the robot is able to navigate in this space. If not, the personal space,

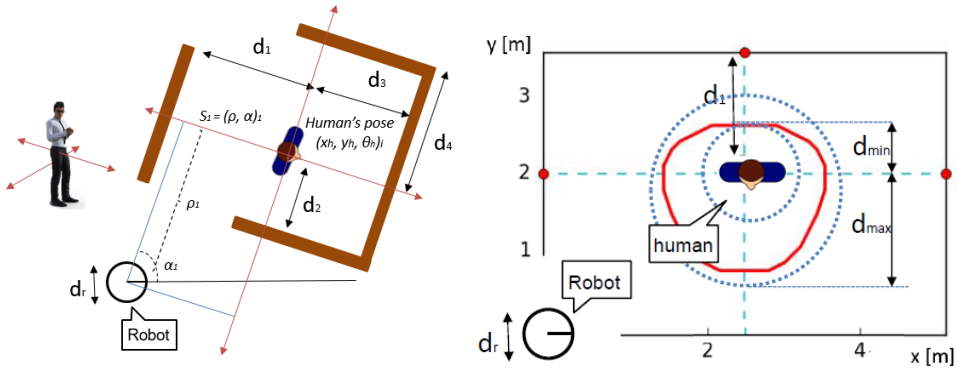
explained mathematically in the annex A, is adapted by varying the corresponding variance of the asymmetric Gaussian function σ (*i.e.*, σ_s, σ_h and/or σ_r). This algorithm is detailed next:

Calculation of the distance to walls

If a human h_i is inside a room or a corridor, the algorithm evaluates the distance from human position to walls. First, Let $R_i = \{\omega_1, \omega_2, ...\omega_n\}$ be the room where person h_i is located, being ω_k the wall k that composes the complete room (*e.g.*, a corridor has a minimum of two walls, while on the contrary in a typical room there are four different walls). Each wall ω_k is described by a plane $p_{\omega_k} \in \mathbb{R}^3$.

In order to measure distances to each wall, the proposed algorithm generates two different straight lines, S_1 and S_2, being S_j defined as $S_j = (\rho, \alpha)_j$, where ρ is the length of the normal drawn from the origin (*i.e.*, robot) to the line, which subtend an angle α with the positive direction of x-axis.

In Fig. 5.8a is shown a human located in a room R composed of four walls $R_1 = \{\omega_1, \omega_2, \omega_3, \omega_4\}$ and where $S_1 = (\rho, \alpha)_1$ is also drawn. Next, intersection point between S_1 (and S_2) and the plane p_{ω_k} define the distance d_k from the human position to ω_k. In Fig. 5.8a four different distances are shown: line S_1 defines two different distances, d_1 and d_3, while S_2 defines the distances d_2 and d_4 (intersection between S_2 and walls ω_2 and ω_4, respectively). The set of all the distances from human to walls defines the distance vector d_T.



(a) Intersection between the walls and the lines S_1 and S_2.

(b) Distances used in the proposed work.

Figure 5.8: Description of the method for measuring distances between human and walls

Evaluation of the spatial context

Once the distance vector d_T has been calculated, the next step is to evaluate if the personal space must adapt its shape to the spatial context. Let d_r be the diameter of the robot plus a safety margin. Besides, let d_{min} and d_{max} be the minimum and maximum distances that define the comfort zone, respectively. These values are presented in 5.8b as circumferences with center $h_i(x,y)$ and radius d_{min} and d_{max}. For each $d_i \in d_T$, the proposed method calculates the distance d_s:

$$d_{s_i} = d_i - d_r \tag{5.3}$$

Considering different values of d_{s_i}:

- If $d_{min} \leq d_{s_i} \leq d_{max}$: It is possible to adjust the comfort area in order to allows robot navigation.
- If $d_{s_i} \leq d_{min}$: The robot is unable to navigate in this orientation, since the personal space can not be adjusted.
- If $d_{s_i} \geq d_{max}$: There is no need to modify the shape of the personal space.

In Fig. 5.9, the new Gaussian is presented after consider the spatial context of the human-robot interaction. In the figure, d_{s_2} and d_{s_3} involve a modification of the personal space in the scenario presented.

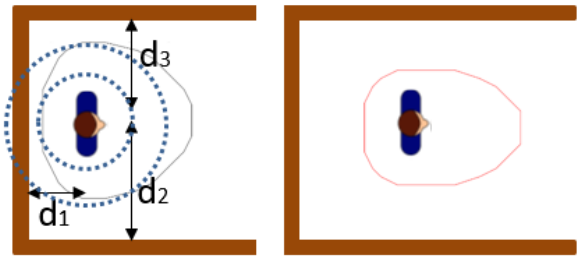


Figure 5.9: Evaluation and adaptation of the spatial context in a typical narrow corridor

Environment-dependent personal space

In this stage the personal space model is adapted to the spatial-context. As was described in the appendix A, the model used in [1] is an asymmetric Gaussian $g_{h_i}(x,y)$ defined by σ_s, σ_h and σ_r. By varying some of these values, $g_{h_i}(x,y)$ adapts its shape to the environment. Let $g'_{h_i}(x,y)$ be the new model, where now σ'_s, σ'_h and σ'_r are dependent of the spatial context. This dependence is modeled as non-linear regression $\sigma' = a \cdot {d_s}^b$, being a and b parameters that define the shape of the curve and the magnitude of the σ' value.

The a and b parameters have been chosen as the coefficient that better fit the data to the curve giving by the expression $\sigma' = a \cdot d_s^b$. A set of σ values are used in order to get the corresponding contours of the asymmetric gaussian $g_h(x,y)$. By fixing the density threshold, ϕ, of the cost function, these contours define different distances as is shown in Fig. 5.10. From these data, $a = 1.2$ and $b = -0.976$, where $R^2 = 0,97$.

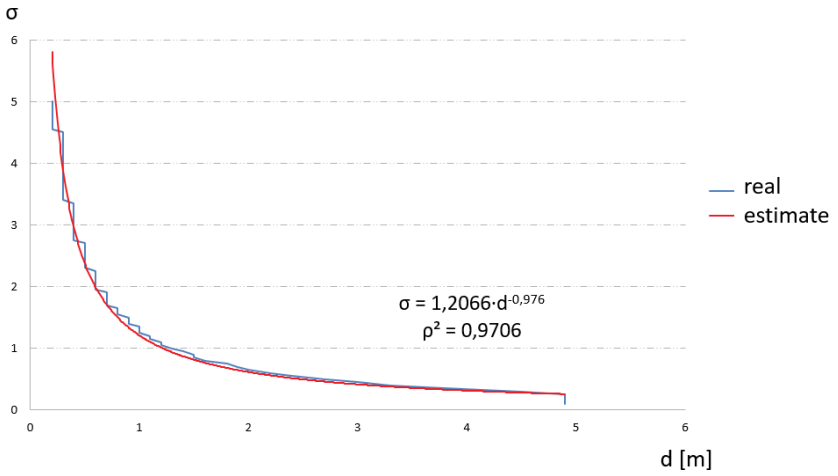


Figure 5.10: Regression used to model the dependence between *sigma* and the distance to the sides, back and front.

Finally, this personal space describes a region where robots' navigation is forbidden, such was described in [1].

5.4.2 Path planning dependent on Personal Space

The second method proposed to adapt the personal space of a human h_i to the environment consists of defining three spaces around the person (intimate, personal and social space) and planning the robot's trajectory on the basis of these spaces, modifying the costs of the nodes of the free space graph. This algorithm is detailed next:

Definition of different personal spaces

The original navigation algorithm employs an asymmetric Gaussian function to establish the personal space of an individual h_i. This algorithm establishes a fixed density threshold ϕ, which defines how close to human the prohibited boundary J will be for navigation, in order to make the robot respect the human's personal space. The contour J delimits how near the person the robot can navigate. In Fig. 5.11 are shown several 2D views of the representation of the Gaussian curve generated for a human looking to the right. If the personal space of the human is understood as a "cut" in the Gaussian curve in function of the density threshold ϕ, it can be observed how the

personal space would be modified in function of this one, being the personal space bigger the lower the threshold.

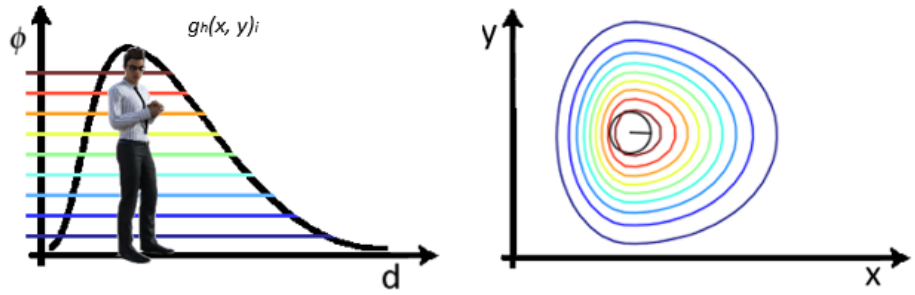


Figure 5.11: 2D representation of the upper and lateral views of the asymmetric Gaussian curve generated for a person looking to the right. The different dimensions that the contour J would have in function of ϕ are shown.

In view of the foregoing, this method proposes the variation of the density threshold ϕ for the definition of not one, but three contours J_i of different sizes around the person, each one defined by the same asymmetric gaussian curve $g_{h_i}(x,y)$, explained in the annex A.

Each human h_i present in the environment will have three associated spaces: the intimate space, delimited by the contour $J_{intimate}$; the personal space, delimited by $J_{personal}$; and the social space, delimited by J_{social} , each of them being larger than the previous one, as it was introduced in [6]. The Gaussian curve $g_{h_i}(x,y)$ that generate these contours is defined with the same variance σ as it was done in [1] (i.e $\sigma_h = 2$, $\sigma_r = 1$ and $\sigma_s = 4/3$).

According to [6] it is possible to classify the space around a person into four zones, depending on social interaction. This distances to the human are listed bellow:

- Public zone $>$ 3.6 m
- Social zone$>$ 1.2 m
- Persona zone $>$ 0.45 m
- Intimate zone $<=$ 0.45 m

Considering the previous data, the distances D_h, D_s, D_r have been evaluated for each value of ϕ. These values are the distances from the person to the contour J in the front direction D_h, side direction, D_s, and rear direction, D_r. The goal is to find a suitable ϕ that provides a value of D_h similar to the proposed by [6].

Table 5.1: Variation of the distance from the person to the forbidden contour for navigation J, depending on the density threshold ϕ

ϕ	D_h (m)	D_s (m)	D_r (m)
0,1	2,1	1,6	1
0,2	1,7	1,3	0,8
0,3	1,5	1,1	0,7
0,4	1,3	1	0,6
0,5	1,1	0,8	0,5
0,6	1	0,7	0,5
0,7	0,8	0,6	0,4
0,8	0,6	0,5	0,3
0,9	0,4	0,3	0,2

For the selection of ϕ only the distance D_h has been considered. The chosen values have been $\phi = 0.1$ for the social space; $\phi = 0.4$ for the personal space (which is same value used in [1] to generate the personal space) and $\phi = 0.8$ for the intimate space.

The three J_i contours obtained are defined, as it was done in [1] by a set of k polygonal chain (*i.e.*, polyline) $L_k = \{l_1, ..., l_k\}$, where k is the number of regions detected by the algorithm. The curve l_i is described as l_i= $\{a_1, ..., a_m\}$, being $a_j = (x, y)_j$ the vertices of the curve, which are located in the contour of the region J_i. In this way, each person h_i will have three polylines associated ${L_i}^{intimate}$, ${L_i}^{personal}$ and ${L_i}^{social}$. In the Fig. 5.12 are shown these polygonal curves: in color red is represented the intimate space, in purple the personal one and in color blue the social space.

Modification of the free space graph costs

As it was introduced in section 5.1.1 the current free space graph is represented by a set of nodes, N considered free or occupied depending on its availability $A[N_i]$. These nodes also have a parameter that represent its cost $C[N_i]$. The robot plans the shortest path between the points of the graph using the Dijkstra algorithm and taking into

account the cost of each node: the path is considered ”longer” when the cost of the nodes are higher. The road is not physically longer, however can be interpreted as if it is. Initially, all the nodes of the graph have $C[N_i] = 1$.

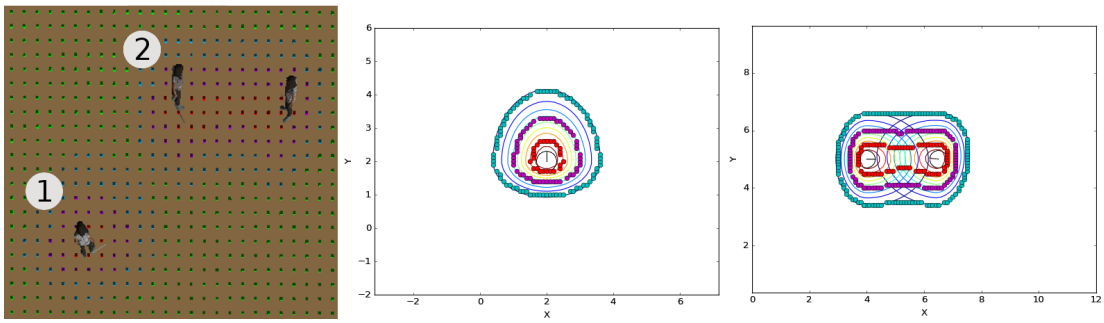


Figure 5.12: Definition of three spaces of interaction around the person and modification of the costs of the graph to take into account these spaces

Being A the matrix formed by the availability of each node and C the matrix formed by the costs and considering the set of polygonal curves defined bellow, ${L_k}^{intimate}$, ${L_k}^{personal}$ and ${L_k}^{social}$, this method consist on modify the cost and availability of the nodes of the graph according to these interaction spaces.

In first place, considering only the intimate space around the person h_i, for each polyline $l_i^{intimate}$ is defined a polygon $P_i^{intimate}$ formed by the points of the polyline. The availability $A[N_i]$ of all the nodes $N_i \in N$ contained in the space formed by $P_o^{intimate}$ is set to occupied, $A[N_i] = occupied$. This means that the robot will not be able to invade this space, as it would be too annoying for the person. For personal and social spaces, the availability of the nodes of the graph will not be modified, but its cost will be changed.

Considering the personal space around the human h_i, in the same way that has been done for the intimate space, for each polyline $l_i^{personal}$ a polygon $P_i^{personal}$ has been defined. The cost $C[N_i]$ of all the nodes $N_i \in N$, contained in the space formed by $P_p^{personal}$ will be modified and set to $C[N_i] = 4.0$. In the same manner, for the social space, a polygon P_p^{social} is defined for each polyline $l_i^{personal}$. All the nodes $N_i \in N$ contained in the space formed by P_i^{social} will have cost $C[N_i] = 2.0$. The public space will be the rest of the graph whose costs remain unchanged.

Therefore, the intimate space has been set to occupied, in order that the robot never crosses this area. The personal and the social spaces are set as free, but its costs have been increased, giving the personal space a higher cost than the social space. In this way, when the robot plans the shortest path, it will move away from the person. The social and personal spaces are not considered occupied so if the robot does not have enough space to navigate, for example in a corridor, it won't be blocked, but it will navigate through the social space, even if its cost is higher. If still does not have space, it will cross the personal space, but it will never cross the intimate one.

This same technique has been used for space affordances described in the section 5.3. Being $L_o = \{A_{o_1}, ..., A_{o_n}\}$ the set of polylines that describe the defined Space affordances, for each A_{o_i} the polygon P_i^{aff} is formed. The nodes of the free space graph $N_i \in N$ contained in P_i^{aff} are modified in order to set its cost to $C[N_i] = 1.5$. In this way, the spaces affordances have less weight in the graph than the social space of the person, so if the robot have to go through one of them, it will go through the space affordance.

In addition, the definition of three spaces of interaction around a person can be very useful when establishing interactions between human and robot, as will be explained in the next section.

5.5 Planning Human-Robot interaction

Once a personal space adapted to the social environment has been achieved, the last stage is the definition of the action domain that the navigation agent must have, in order to elaborate social plans in environments with humans.

Planning human-aware navigation tasks entails defining the elements of the planning problem: an initial world model, a mission, and a set of actions (*i.e.*, the planning domain). In CORTEX, planning is performed with the symbolic information in the DSR, using the nodes of the representation as symbols and the edges of the graph as predicates.

In this section the symbols and predicates (nodes and edges used in the DSR) that

are used in order to plan HRI for social navigation are described:

5.5.1 Symbols and predicates

For social navigation purposes, the robot uses three types of symbols: *human*, *robot*, and *room*. This work considers cases in which a *robot* is found in the model, and also the existence of several humans and rooms is possible.

The robot and each person must be located within an existing room; for this purpose *in* predicate (edge in the DSR) is used. Robots and humans might be paying attention to other robots and humans; for this purpose *interact* predicate is used. To represent that a robot is close enough to establish social interaction with a human the robot includes *reach* predicates.

For the definition of the action domain for HRI, different edges have been added to the DSR, from which the robot will have knowledge of the situation that is occurring at each moment. Initially, only three edges are considered, these are:

- *block* edges: Humans might block the path of the robot. If a person is physically blocking the robot, these edges are added between the person and the robot. When there is a *block* edge, the robot must launch an interaction with the person and ask for its **collaboration to pass**.

- *softBlock* edges: These edges are used when the robot can navigate around a person, but its path would get too close to the person, going through its personal spaces. These edges are also used when a a social block exists (*i.e.*, robots are not supposed to interfere visually when two people interact). When the *softblock* edge exists, the robot must launch an interaction with the person, or group of person, and ask **permission to pass** .

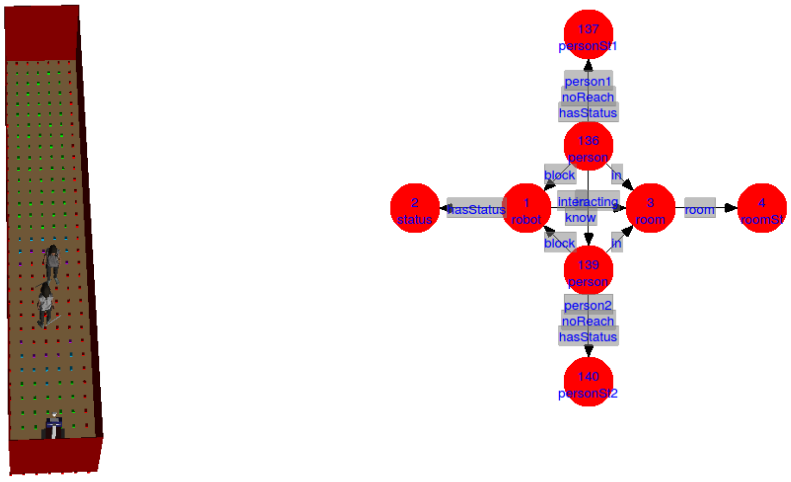
- *affordancesBlock* edges: If the robot cannot navigate because the person is interacting with an object in a Space affordance, the edge *affordanceBlock* is added between the person and the robot. When the edge exists, in the same way

as with *softblock* edges, the robot must launch an interaction with the person and ask **permission to pass**.

Below is described when these links are added to the DSR:

Block edges

As explained above, these edges are added between a person and the robot, when the person blocks the robot's path. The problem appears at the time of identifying if it is a person who is blocking the path, or if on the contrary the robot cannot navigate for other reasons. In addition, if there are more people in the environment, it is necessary to identify the person whose personal space is the one that is blocking the robot's path. In the Fig. 5.13 an example of a person blocking the robot is shown, as well as the DSR graph with the *block* edge between the person and the robot.



(a) Humans blocking the robot in a narrow corridor

(b) DSR graph showing the *block* edge

Figure 5.13: Block Edge example

Given a target N_t, the robot plans the shorter trajectory $T = N_1, N_2, ..., N_t$ using

the Dijkstra algorithm, being N_i the nodes of the graph which forms the shortest path. When planning the path, it can happen that the robot does not find any way to reach its target. It there are humans in the environment, the intimate space of these, delimited by the polylines $L_k^{intimate}$, can be the cause of not finding a path (as explained in section 5.4.2, the intimate space of each person is considered forbidden for navigation).

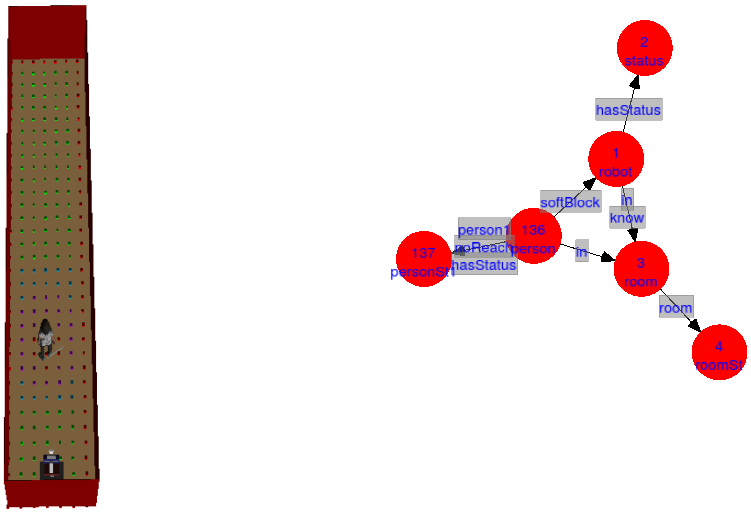
To check if it is a human that blocks the path, first it is considered a free space graph $G_0(N)$ with no humans in the environment. If when planning a trajectory T using the $G_0(N)$ graph no path is found, it means that humans are not the cause of the robot's blocking. If, on the other hand, a path is found, it is checked which human blocks the path of the robot.

Being $L_k^{intimate}$ the list of k intimate regions detected and H the set of humans in the environment, each $l_i^{intimate} \in L_k^{intimate}$ is inserted, one by one, in the free space graph, as explained in section 5.1.1, in different iterations M. In each iteration M_i the path T is calculated again. If in the iteration M_i no trajectory is obtained, it means that the last inserted polyline $l_i^{intimate}$ is the one that blocks the path. Next, it is verified which human $h_i \in H$ is the owner of the polyline $l_i^{intimate}$ that blocks the robot. If the position of the human h_i is contained in the space formed by the polygon $P_p^{intimate}$ (i.e, polygon formed by the points of the polyline), the *block* edge must be added between that human and the robot. If there are two people within this space (i.e., the two persons block the robot by maintaining an interaction), the *block* edge is added between each persons and the robot.

SoftBlock edges

It may happen that the robot finds a path, but it gets too close to a person. In addition, humans in the environment can also socially block the robot. If this happens the edge (softBlock) is added between the person and the robot. The robot should approach the person and launch an interaction to ask permission to pass through that space. In the Fig. 5.14 an example of a person whose personal space the robot needs to cross to reach its target (i.e., the person is "softblocking" the path of the robot), as well as the

DSR graph with the *softblock* edge between the person and the robot.



(a) Human "softblocking" the robot's path

(b) *softblock* edge added between the person and the robot

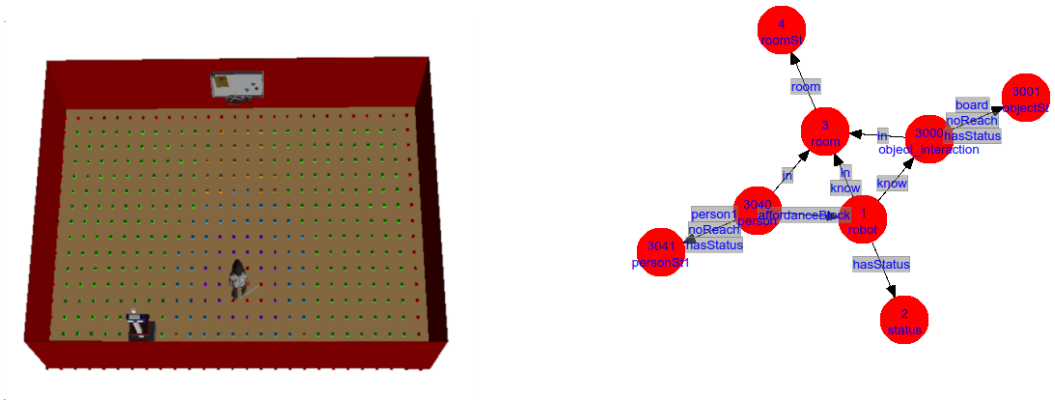
Figure 5.14: SoftBlock Edge example

Being $L_k^{personal}$ the list of k personal regions detected, for each $l_i^{personal} \in L_k^{personal}$ is checked if the planned trajectory T crosses this space. To do this, the polygon $P_p^{personal}$ is formed. For each $N_i \in T$ it is checked if $N : i$ is contained in the space formed by the polygon $P_p^{personal}$. If this happens, the robot needs to cross the personal space of the person, defined by $l_i^{personal}$. Then, for each $h_i \in H$ is checked if its position $h_i = (x, y)$ is contained in the same polygon $P_p^{personal}$. Once the person whose personal space the robot needs to cross is found, the edge (softBlock) is added between this and the robot.

AffordancesBlock edge

When a person is interacting with an object, the Space Affordance defined for that object is called an Activity Space. If the robot needs to cross an Activity space, the edge *affordanceBlock* is added between the person and the robot. When the edge exists,

in the same way as with *softblock* edges, the robot must launch an interaction with the person and ask permission to pass. In the Fig. 5.15 an example of a person interacting with an object in a space affordance is shown, as well as the DSR graph with the *affordancesBlock* edge between the person inside the affordance and the robot.



(a) Person interacting with a pin board

(b) *affordancesBlock* edge added at the DSR between the person and the robot

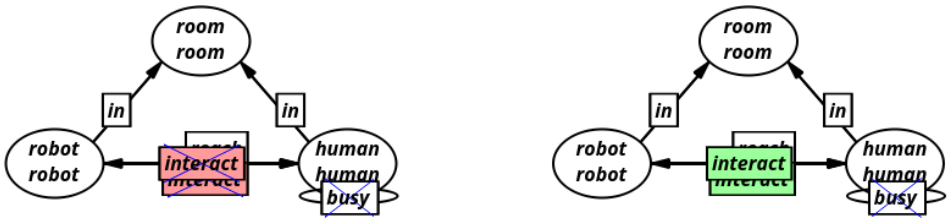
Figure 5.15: AffordancesBlock Edge example

Let $L_o = \{A_{o_1}, ..., A_{o_n}\}$ be the set of polylines that describe the defined Space affordances, for each A_{o_i} is checked if the planned trajectory T crosses this space. To do this a polygon P_{aff}, is defined . For each $N_i \in T$ is it checked if this node is contained in the space formed for the polygon P_{aff}. If this happens, the robot needs to cross the space when it is considered as an Activity Space. Then, for each $h_i \in H$ is checked if its position is contained in the polygon P_{aff}. Once found the person, the edge (affordancesBlock) is added between the person and the robot.

5.5.2 Navigation domain

Although the robot's navigation domain is composed of a large number of actions, in this section are described the most relevant actions for a Human-Robot interaction. These are engageHuman, askForPermissionToPass, and askForCollaborationToPass, which will be explained below:

engageHuman

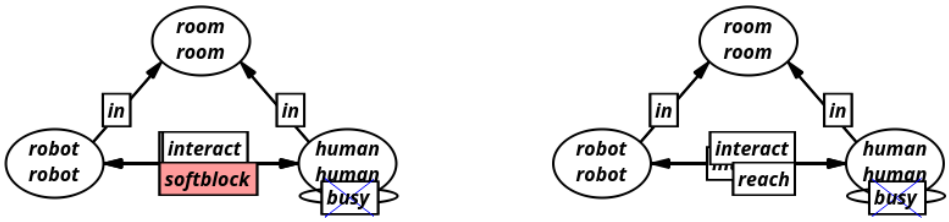


(a) Current DSR (b) Desired DSR graph

Figure 5.16: Action *engage*

The first step to ask for help or permission when navigating is engaging human interaction. This goal is the purpose of the *engageHuman* action. The action states that if a human is reachable, the robot can interact with him unless its symbol is marked as busy (which is done when a human explicitly says that she or he does not want to be disturbed). The effect of the correct execution of the action is two new *interact* edges, one from the human to the robot and *vice versa*. In Fig. 5.16 this action is graphically represented. Fig.5.16b shows the DSR graph desired for the action, i.e, an *interact* edge between the person and the robot.

askForPermissionToPass



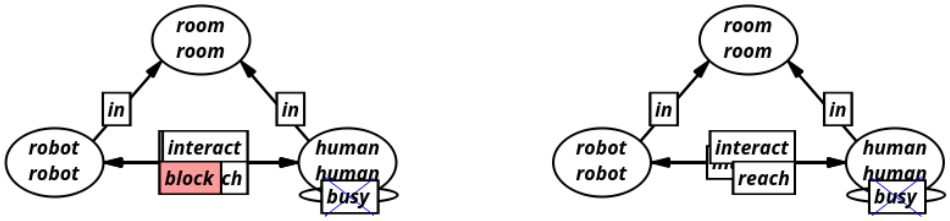
(a) Current DSR graph (b) Desired DSR graph

Figure 5.17: Action *askForPermissionToPass*

Once the robot is interacting with a particular human, it can ask for permission to pass in the case it needs to cross a determinate space. The precondition for the robot to be able to execute the *askForPermissionToPass* action would be the existence of a

softblock edge or *affordancesBlock* edge. The result of the successful execution of the action is that the human gives permission to the robot to pass through a certain space (i.e. his personal space in case that a *softBlock* edge exists, or go through the Space Affordance if the edge is *affordancesBlock*). In Fig.5.17 the action is represented graphically. Fig.5.17a shows the pre-contion to the action: the existence of a *softBlock* edge; and Fig. 5.17b shows the desired DSR graph for the realization of the action: an *interact* edge between the person and the robot.

askForCollaboration



(a) Current DSR graph

(b) Desired DSR graph

Figure 5.18: Action *askForCollaboration*.

The action *askForCollaboration* is similar to *askForPermission*. The difference is that in this case the human blocks the path physically. The *block* edge is added between the person and the robot. Therefore, in this case the robot asks the human to move and waits. In Fig.5.18 a visual definition of the action is shown. Fig.5.18a shows the pre-condition for the action: the existence of a *block* edge; and in Fig. 5.18b shows the desired DSR graph for the realization of the action: an *interact* edge between the person and the robot.

An example of the HRI planning is shown in Fig. 5.19: the social robot (labeled as '1') has an approach behavior with which it can initiate conversation with people (labeled as '2'). As the path is blocked, the robot asks for cooperation (labeled as '3'). Once the path is free, the social robot navigates until its target (labeled as '4'). A zoom of this test in '2' and '3' robot position is drawn on the right, where the changes in the graph of free space are illustrated.

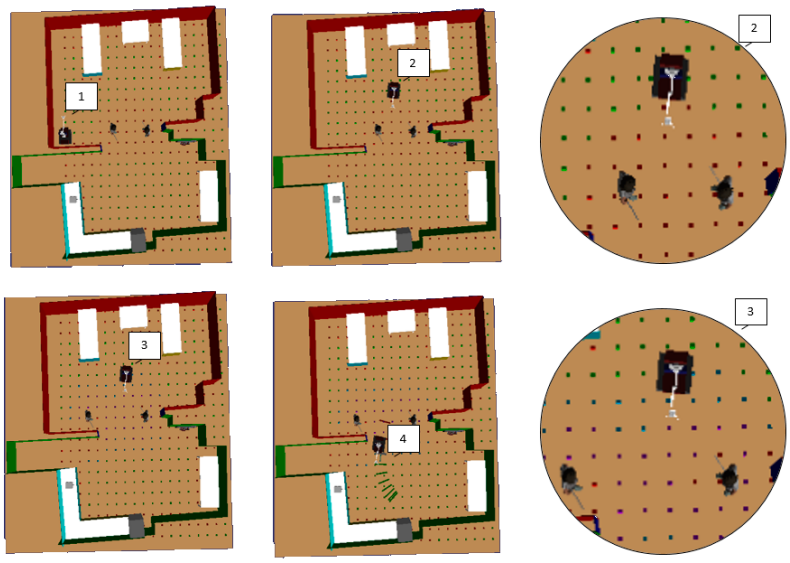


Figure 5.19: An example of the HRI planning described in this work: ask for collaboration.

Chapter 6

Results

The goal of this work was to advance in the area of social navigation, by improving the social navigation algorithm presented in [1]. In section 5, the set of proposed improvements have been widely presented. These include the idea of a navigation algorithm that adapts the personal space of humans to the environment, the inclusion of interactive spaces and the proposal for a human-robot interaction planning domain. In this section several experiments have been carried out to check the efficiency of the improvements proposed.

Firstly, the metric used in this work is introduced:

6.1 Metrics employed

The methodology of the different experiments carried out has been evaluated accordingly to these metrics: average minimum distance to a human during navigation, d_{min}; distance travelled, d_t; navigation time, τ; cumulative heading changes, CHC; and personal space intrusions, Psi. These metrics have been already established by the scientific community ([15, 31]). A brief description of these metrics are described as follows:

- Average distance to the closest human during navigation: A measure of the average distance from the robot pose, $x_r(x, y, \theta)$, to the closest human $h_i(x, y, \theta)$

along the robot's path $P = \left\{ x_r^j(x,y,\theta) \mid j = 1,2...N \right\}$, being N the number of points of the trajectory.

$$d_{min} = min_i \left\{ \| x_r{}^j(x,y) - h_i(x,y) \| \right\} \tag{6.1}$$

- Distance travelled: length of the robot's path, in meters.

$$d_t = \sum_{j=1}^{j=N-1} \| x_r^j(x,y) - x_r^{j+1}(x,y) \| \tag{6.2}$$

- Navigation time: time since the robot starts the navigation, τ_{ini}, until it arrives to the target, τ_{end}.

$$\tau = \tau_{end} - \tau_{ini} \tag{6.3}$$

- Cumulative heading changes: a measure to count the cumulative heading changes of the robot during navigation [31]. Angles are then normalized between $-\pi$ and π.

$$CHC = \frac{1}{N} \sum_{j=1}^{j=N-1} \| x_r^j(\theta) - x_r^{j+1}(\theta) \| \tag{6.4}$$

- Personal space intrusions (*Psi*): In this work, four different interaction areas are defined: Intimate ($\| x_r^j(x,y) - h_i(x,y) \| \leq 0.45m$); Personal ($0.45m \leq \| x_r^j(x,y) - h_i(x,y) \| \leq 1.2m$); Social ($1.2m \leq \| x_r^j(x,y) - h_i(x,y) \| \leq 3.6m$); and Public ($\| x_r^j(x,y) - h_i(x,y) \| \geq 3.6m$). Along the robot's path, this metric measures the percentage of the time spent in each area as:

$$Psi = \left\{ \frac{1}{N} \sum_{i=1}^{i=N} \mathscr{F} \| x_r^j(x,y) - h_i(x,y) \| \leq \delta^k \right\} \tag{6.5}$$

where δ^k defines the distance range for classification (intimate, personal, social and public), and $\mathscr{F}()$ is the indicator function.

6.2 Evaluation of the methods for adapting personal space to the environment

Two algorithms have been proposed to adapt the personal space of a human the spatial context. In this section both algorithms have been evaluated in order to compare them. A description of the tests carried out to check the performance of each algorithm is given below.

The algorithms have been developed in Python and C++ software and the benchmark tests have been performed on a PC with processor Intel Core i5 2.4GHz with 4Gb of DDR3 RAM and GNU-Linux Ubuntu 16.10.

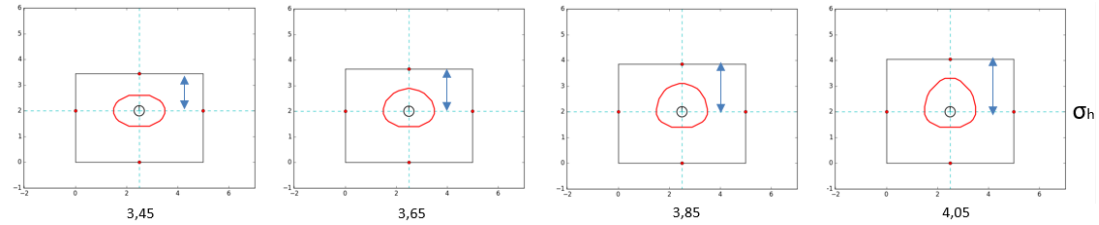
6.2.1 Environment-dependent personal space modeling evaluation

In order to evaluate the method, firstly a basic simulated scenario has been created. This consists of a square room without objects, with dimensions $5m \times 5m$ and the presence of a human. This room has mobile walls, in that way it is possible to reduce the distances from the human to each wall. Fig. 6.1 shows the results of the proposed algorithm for adapting the personal space to the spatial context.

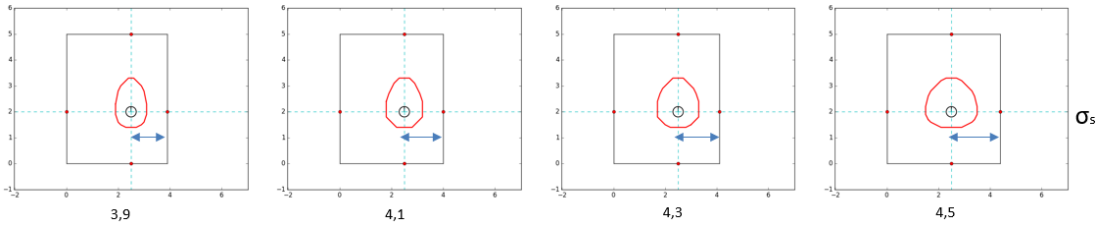
In Fig. 6.1a, four different tests are illustrated, where the distance to the wall in front of the person is increased (from left to the right). The contour map of the personal space is shown in the figure, where is drawn as the gaussian function $g'_h(x,y)$ adapted correctly to the context. Similar results are illustrated in Fig. 6.1b and Fig. 6.1c, where the distance to the wall has been varied in the sides of the person and in the rear, respectively. For the results evaluation, a comparison between the original cost function $g_h(x,y)$ and the adapted function $g'_h(x,y)$ is shown in the tables 6.1,6.2 and 6.3. In this tables is also indicated if the robot crosses between the human and the wall. Depending of the value of d_{free}, the robot will be able to cross or not this space. If $d_{free} \leq d_r$ the robot won't be able to do it. As is shown in Table 6.1, the flexible spatial density function $g'_h(x,y)$ adapts correctly to the spatial context.

It should be highlighted that $d_r = 0,8$. In addition, d_{min} and d_{max} are not the same in the heading than in the sides or in the rear of the person, being this values the following:

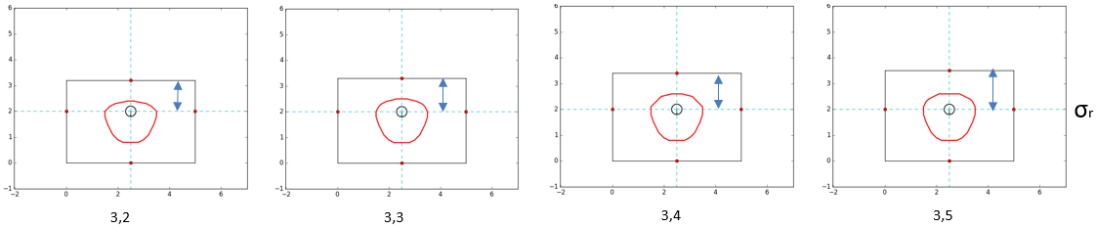
- Heading: $d_{min} = 0,6m$, $d_{max} = 1,3m$
- Sides: $d_{min} = 0,3m$, $d_{max} = 0,6m$
- Rear: $d_{min} = 0,5m$, $d_{max} = 1m$



(a) Adapting σ_h



(b) Adapting σ_s



(c) Adapting σ_r

Figure 6.1: Four different tests were conducted for each σ_h, σ_s and σ_r. Tests consist of varying the distance from human to the corresponding wall. Distances (in meters) are drawn in the figure.

Table 6.1: Comparative results of the cost function defined in [1] and the proposed in this work, by varying the distance to the wall in front of the person.

Heading								
gh(x,y)				g'h(x,y)				
d_3	d_s	d_{free}	cross?	d_3	d'_s	d_{free}	σ'_h	cross?
2,5	1,3	1,2	YES	2,5	1,3	1,2	1,00	YES
2,05	1,3	0,75	NO	2,05	1,2	0,85	1,05	YES
1,95	1,3	0,65	NO	1,95	1,1	0,85	1,10	YES
1,85	1,3	0,55	NO	1,85	1	0,85	1,21	YES
1,75	1,3	0,45	NO	1,75	0,9	0,85	1,34	YES
1,65	1,3	0,35	NO	1,65	0,8	0,85	1,50	YES
1,55	1,3	0,25	NO	1,55	0,7	0,85	1,71	YES
1,45	1,3	0,15	NO	1,45	0,6	0,85	1,99	YES
1,35	1,3	0,05	NO	1,35	1,3**	-	1,00	NO

Table 6.2: Comparative results of the cost functions by varying the distance to the wall to the right of the person.

Sides								
gh(x,y)				g'h(x,y)				
d_1	d_s	d_{free}	cross?	d_1	d'_s	d_{free}	σ'_s	cross?
1,9	1,0	0,9	YES	1,9	1	0,9	1,33	YES
1,8	1,0	0,8	NO	1,8	0,95	0,85	1,05	YES
1,7	1,0	0,7	NO	1,7	0,85	0,85	1,41	YES
1,6	1,0	0,6	NO	1,6	0,75	0,85	1,60	YES
1,5	1,0	0,5	NO	1,5	0,65	0,85	1,84	YES
1,4	1,0	0,4	NO	1,4	0,55	0,85	2,16	YES
1,3	1,0	0,3	NO	1,3	1**	-	1,33	NO

Table 6.3: Comparative results of the cost functions by varying the distance to the wall to the rear of the person.

Rear								
gh(x,y)				g'h(x,y)				
d_2	d_s	d_{free}	cross?	d_2	d'_s	d_{free}	σ_r'	cross?
1,5	0,6	0,9	YES	1,5	0,85	0,65	2,00	YES
1,4	0,6	0,8	NO	1,4	0,55	0,85	1,05	YES
1,3	0,6	0,7	NO	1,3	0,45	0,85	2,63	YES
1,2	0,6	0,6	NO	1,2	0,35	0,85	3,36	YES
1,1	0,6	0,5	NO	1,1	0,6**	-	2,00	NO

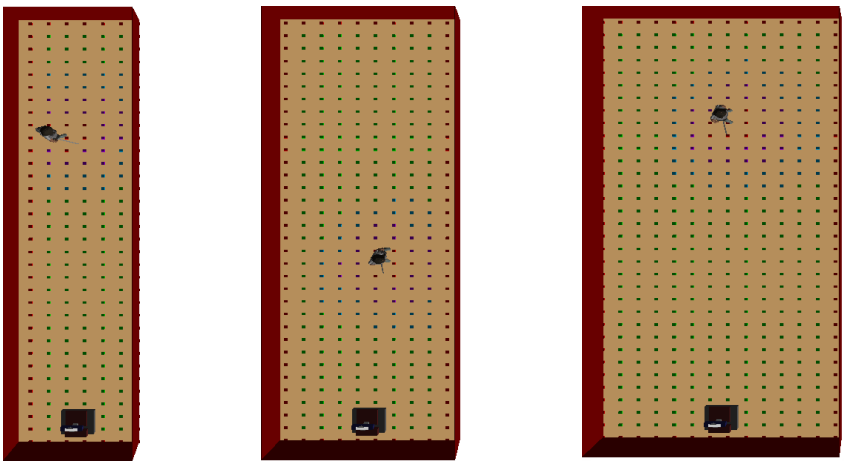
Those results marked with two asterisks (**) indicate that the distance $d_s < d_{min}$.In

these cases the Gaussian curve has not changed and the original σ values have been used.

From the results shown in the tables 6.1, 6.2 and 6.3, it can be concluded that adapting personal space to the environment produces better navigation. In comparison with the results obtained for a fixed person space, $gh(x,y)$, adapting the personal space ensures that the robot is not blocked in reduced environments. For example, with a fixed personal space $gh(x,y)$ the robot cannot navigate when the person is 2.05 m from the wall facing it. However, by adapting the personal space $g'h(x,y)$ the robot can circulate in the previous case even if the person is less than 1.5m from the wall.

With this algorithm the robot always tries to reach its destination, getting close to the person whenever it can. Although this algorithm provides social navigation even in reduced spaces, the robot can get quite close to the person, which can be annoying.

6.2.2 Path planning dependant on Personal Space Evaluation



(a) 2m wide (b) 3m wide (c) 4m wide

Figure 6.2: Scenarios used in the second experiment

To evaluate the performance of the second proposed algorithm to adapt the personal space to the spatial context, several simulations have been performed in three different spaces. The widths of the rooms used were 2, 3 and 4 m, while the height of the rooms

was constant, 10 m. In Fig. A.1 the used scenarios can be seen. A simulated version of the robot Viriato has been used. Viriato has had to navigate from the position $x = 0m$, $y = 0m$ to $x = 8.5m$, $y = 0m$, through those scenarios in which a person was located in random positions. The aim of the experiment is to measure the percentage of time (i.e. the personal space intrusions, *Psi*) that the robot spends in each interaction space defined for the person, as explained in the section 5.4.2. The "softblock" produced when the robot has to cross the personal space of a human has not been considered for this experiment, so the robot will pass through that space without launching the interaction with the human in first place, as explained in the section 5.5.

The results obtained for the simulations of rooms 2, 3 and 4 meters wide can be found in the tables 6.4, 6.5 and 6.6, respectively.

Table 6.4: Navigation results for 2m wide room considering interaction spaces

Parameter	Obtained value (σ)
d_t *(m)*	9.71 (0.56)
τ*(s)*	40.39 (11.83)
CHC	1.49 (0.58)
d_{min} *Person (m)*	1.13 (0.16)
Ψ *(Intimate) (%)*	0.0 (0.0)
Ψ *(Personal) (%)*	4.43 (7.38)
Ψ *(Social) (%)*	16.46 (11.46)
Ψ *(Public) (%)*	79.10 (13.92)

Table 6.5: Navigation results for 3m wide room considering interaction spaces

Parameter	Obtained value (σ)
d_t *(m)*	10.02 (0.49)
τ*(s)*	31.39 (3.64)
CHC	0.88 (0.18)
d_{min} *Person (m)*	1.62 (0.28)
Ψ *(Intimate) (%)*	0.0 (0.0)
Ψ *(Personal) (%)*	0.0 (0.0)
Ψ *(Social) (%)*	3.29 (5.41)
Ψ *(Public) (%)*	96.70 (5.41)

Table 6.6: Navigation results for 4m wide room considering interaction spaces

Parameter	Obtained value (σ)
d_t *(m)*	10.81 (0.55)
τ*(s)*	45.65 (19.24)
CHC	1.27 (0.51)
d_{min} *Person (m)*	1.76 (0.18)
Ψ *(Intimate) (%)*	0.0 (0.0)
Ψ *(Personal) (%)*	0.0 (0.0)
Ψ *(Social) (%)*	2.017 (3.37)
Ψ *(Public) (%)*	97.98 (3.37)

As can be seen in the results obtained in tables 6.4, 6.5 and 6.6, the robot is capable of avoiding human interaction spaces in planning. The robot tries to avoid the different interaction spaces defined for the person. For environments more than 2m wide, the robot does not invade the person's personal space, as Ψ *(Personal)* is equal to 0.0 . It does this without deforming the human space. The main difference with the previous algorithm is that with this one, the robot is always as far away from the person as possible. However, the smaller the room, the more difficult it becomes. For instance, for the 2m wide room, the robot must pass through the human's personal space, as can be seen in Fig. 6.2a. As already explained, this can be annoying for the person. In this case, the Human-Robot interaction would be launched, so that the robot could ask for permission beforehand. The advantage of this algorithm is that used in conjunction with Human-Robot interactions, provides a much more acceptable social navigation, interacting with people only when necessary and without deforming their personal space, getting closer than a person could desire.

Regarding the other metrics used, it can be noted that the values of τ, d_t and *CHC* are quite high. This is due to the presence of humans, which makes the robot's path longer. In a non-social robot navigation it is desired these values to be as low as possible. However, since there are humans in the environment, a compromise appears between them and the comfort of the human being, the latter being the priority.

It should also be noted that the variance of the results obtained is quite high. This is due to the fact that the experiments were carried out in random positions of the person, with the aim of measuring the personal space intrusion, Ψ; therefore the same

navigation results are not obtained for each of the simulations performed.

6.2.3 Comparison of proposed methods for adapting personal space to the environment

Two different methods of adapting personal space to the environment have been presented. Although both of them have shown positive results for the social navigation of the robot, they are very different from each other and each has its advantages and disadvantages. Below, a comparison between both is shown.

- When considering a deformable personal space around the person (first proposed method), the robot will try to adapt the human space in the first place, so it won't have to bother the person if it doesn't have enough space to pass through.
- This algorithm provides social navigation, without invading personal space and without the need to launch HRI interactions in reduced environments.
- The main disadvantage of this algorithm is that the robot is always going to try to reach its destination, even though this means getting very close to the person.
- Deforming a person's space is not always pleasant for that person, so it would be more logical to establish certain spaces that the robot tries to avoid and, if unable to do so, launch an interaction with the person when possible.
- In addition, the first algorithm presents the difficulty of previously having to know the positions of the walls, so this information must first be entered into the symbolic model of the robot.
- By introducing interaction spaces around the person (second proposed method), there are more possibilities for action, such as checking whether the robot needs to go through one of these spaces and act accordingly.
- The second algorithm proposed has the advantage that the robot always tries to navigate as far away from the person as possible.

- The use of this algorithm in conjunction with human-robot interactions provides a much more appropriate social navigation. The robot will respect interaction spaces when possible and, unable to do so, it will launch an interaction with the person to ask permission or collaboration to pass.

For these reasons, the second proposed method has been used for the development of the rest of work, as it can be more useful. It can be concluded that this second method is effective for rooms more than 2 meters wide. Although the first method proposed allowed navigation in smaller environments, as mentioned above, deforming the person's space is not pleasant for the person. In addition, it is preferable to get as far away from the person as possible and ask permission when it is necessary to get close.

6.3 Navigation in interactive scenarios with Space Affordances

To test the performance of the space affordances algorithm, a rectangular simulated environment with a whiteboard on it has been used. It has been considered an experiment in which the robot's trajectory is not blocked by the space affordance.

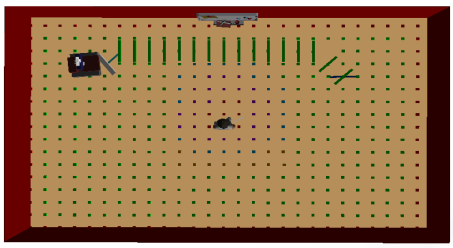
The simulated environment is shown in Fig. 6.3a. The object has been placed in the position $x = 2m$, $y = 4.5m$ with $a_s = 3m$ in order to create a space affordance which the robot has to avoid if people in the environment are interacting with it, what means that the space affordance is being used as an Activity Space.

A single human, placed in front of the object in the position $x = 2m$, $y = 2m$, has been used for this test. The robot has had to navigate from the position $x = -0.8m$, $y = 3m$ to $x = 4.5m$, $y = 3m$, avoiding the space affordance of the object if the person is interacting with it.

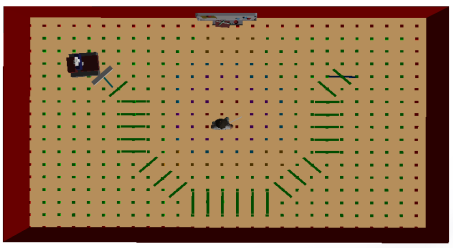
The same test has been carried out with and without space affordances. The comparison between the different paths the robot took can be seen in Fig. 6.3b, and

Fig.6.3c where it is marked the path planned in both situations. It can be noticed that, in the first case, the robot interrupts the human in the performance of its activity.

(a) Simulated interactive scenario



(b) Navigation withouth Space Affordance



(c) Navigation with Space Affordance

Figure 6.3: Interactive scenario described for the test and navigations results with and withouth spaces affordances

Table 6.7 shows the results of navigation with and without space affordances, obtained for each of the metrics used, explained in 6.1: average minimum distance to a human during navigation, d_{min}; distance traveled, d_t; navigation time, τ; cumulative heading changes, *CHC* and personal space intrusions, Ψ. It is also indicated whether the activity performed by the human has been interrupted or not.

Table 6.7: Navigation results with space affordances

Navigation with space affordances		Navigation without space affordances	
Parameter	Value (σ)	Parameter	Value (σ)
d_t *(m)*	8.76m	d_t	5.18m
τ	64.1s	τ	33.84s
CHC	1.47 (0.11)	*CHC*	0.21 (0.05)
d_{min} *Person (m)*	0.78 (0.007)	d_{min} *Person (m)*	1.10 (0.005)
Ψ *(Intimate) (%)*	0.0 (0.0)	Ψ *(Intimate) (%)*	0.0 (0.0)
Ψ *(Personal)(%)*	0.0 (0.0)	Ψ *(Personal)(%)*	0 (0.0)
Ψ *(Social)(%)*	15.46 (0.6)	Ψ *(Social)(%)*	12.54 (0.57)
Ψ *(Public)(%)*	84.53 (0.6)	Ψ *(Public)(%)*	87.44 (0.9)
Interruption (Y/N)	N	*Interruption (Y/N)*	Y

In the table 6.7 the navigation results with and without spaces affordances can be seen. Firstly, it should be noted that, as is obvious, the robot navigates further when considering the space affordance (i.e. d_t is higher). For the same reason, in this situation the robot takes longer to reach its destination (i.e. τ is higher). In addition, the CHC is also higher, since if the space affordance are not considered, the path is practically straight.

It should be noted that, in the experiment in which space affordances are considered, the minimum distance to the person (i.e. d_{min}) is less than when they are not considered. This is due to the shape of the Gaussian curve used to describe human personal space. This curve, as explained in [1], is smaller from the rear than from the heading of the person, therefore, when the robot tries to avoid space affordance it can get closer to the person.

Regarding the personal space intrusion, *Psi*, in both scenarios the interaction spaces are respected. The robot has invaded the human social space when necessary and this has been only in a small part of the path.

6.4 Human-Robot Interaction Experiment

A set of simulated scenarios were used to validate the results of the proposed navigation planning domain. The algorithms have been developed in C++ and the tests have been performed in a PC with an Intel Core i5 processor with 4Gb of DDR3 RAM and Ubuntu GNU/Linux 16.10. Quantitative and qualitative experimental results have been evaluated, including social navigation metrics and the results of a Likert-scale satisfaction questionnaire. A simulated version of the robot Viriato, a social robot equipped with an RGBD camera and laser range sensor has been used.

The simulated scenario is a $65m^2$ two-room apartment equipped with a kitchen and a living room, where two different tests are described[1]: First, a human blocks the path in the corridor; and second, two people talking blocking the robot's path. The robot Viriato navigates through this apartment to several positions. Fig. 6.4 summarises the tests in six steps: In Fig. 6.4. 1 the robot starts its route. Its first target is located in the corridor. In Fig. 6.4.2 the robot plans its path and navigates to the human. After asking for collaboration, the robot navigates to the first target (Fig. 6.4.3). In the second test (Fig. 6.4.4), the robot has the target in the second room, plans its path and initiates a conversation with people (Fig. 6.4.5). Finally, once the robot asks for permission to pass, it navigates to its target position.

[1] A video of the experiments is located on goo.gl/KdGYBN

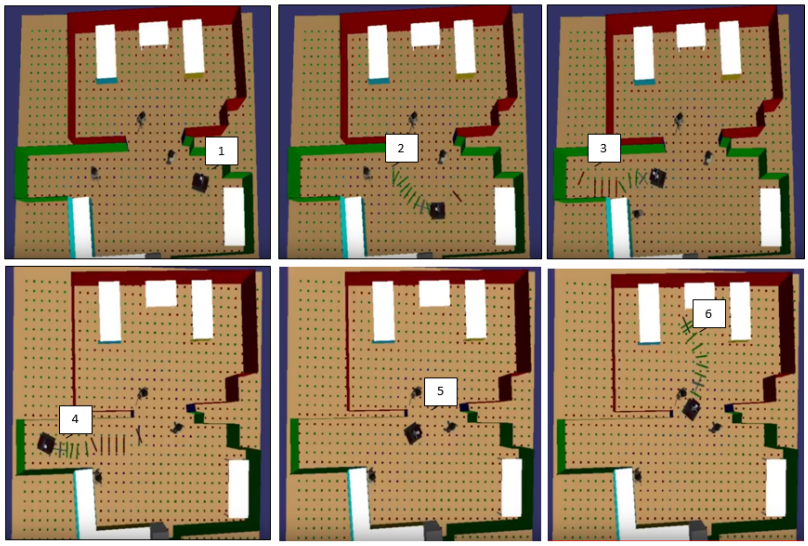


Figure 6.4: Steps of the test used in the experiment

In order to assess the effectiveness of the proposed navigation approach, the methodology has been evaluated accordingly to the metrics explained in 6.1: average minimum distance to a human during navigation, d_{min}; distance traveled, d_t; navigation time, τ; cumulative heading changes, *CHC* and personal space intrusions, Ψ. Results are summarised in Table 6.8 and Table 6.9.

Table 6.8: Results of the first HRI experiment

Parameter	Obtained values (σ)
d_t *(m)*	7.44 (0.15)
τ*(s)*	46.06 (14.75)
CHC	0.76 (0.15)
d_{min} *Person 1 (m)*	1.68 (0.03)
d_{min} *Person 2 (m)*	1.67 (0.03)
d_{min} *Person 3 (m)*	1.47 (0.04)
Ψ *(Intimate) (%)*	0.0 (0.0)
Ψ *(Personal)(%)*	19.36 (1.05)
Ψ *(Social)(%)*	13.03 (2.48)
Ψ *(Public)(%)*	67.6 (2.34)

Table 6.9: Results of the second HRI experiment

Parameter	Obtained values (σ)
d_t*(m)*	8.16 (0.35)
τ *(s)*	44.8 (12.52)
CHC	1.37 (0.18)
d_{min} *Person 1 (m)*	1.69 (0.15)
d_{min} *Person 2 (m)*	1.27 (0.09)
d_{min} *Person 3 (m)*	1.00 (0.06)
Ψ *(Intimate) (%)*	0.0 (0.0)
Ψ *(Personal)(%)*	26.86 (3.10)
Ψ *(Social)(%)*	17.33 (4.92)
Ψ *(Public) (%)*	55.8 (4.26)

To assess the satisfaction of the humans regarding the robot's behavior and HRI

abilities, a Likert scale-based questionnaire was provided to a total of 34 participants. The results of the questionnaire, including the questions are shown in table 6.10.

Table 6.10: Satisfaction questionnaire.

Question	avg. answer (σ)
Robot's behavior is socially appropriate in exp. 1	4.41 (0.54)
Robot's behavior is socially appropriate in exp. 2	4.47 (0.40)
Robot's behavior is friendly in exp. 1	3.79 (0.60)
Robot's behavior is friendly in exp. 2	4.05 (0.52)
The robot understands the social context and the interaction in exp. 1	4.32 (0.62)
The robot understands the social context and the interaction in exp. 2	4.37 (0.71)

In Fig. 6.4 the social behavior of the robot has had during the experiment can be observed. When it detects that the path is blocked, it approaches to the human or group of humans and asks for permission or collaboration to pass.

From the results obtained in the tables 6.8 and 6.9 it should be noted that in both experiments the robot invades the personal space of the human. However, it does so with prior permission to pass, so it does not imply non-social behaviour. In spite of this, most of the way the robot tries to avoid the different zones of social interaction defined for the person.

The other metrics calculated do not provide much information to the evaluation of the interaction between Human and robot.

The video of the experiment has been viewed by a large number of people who have expressed their opinion on the social behaviour of the robot. These results, shown in the table 6.10, can be considered positive, as an average of more than 4 positve responses have been obtained.

Chapter 7

Conclusions and future work

Despite the increasing use of assistance robots in many different areas and applications, the integration of these in a social context is still a problem to solve. This requires to research and develop techniques that allow these robots to act in a socially acceptable way.

This work is a continuation of the previous work presented in [1]. It presents a social navigation approach considering real environments in crowded environments. Several improvements to the previous navigation algorithm are proposed. Among them, the detection and tracking of humans in real time; the inclusion of interactive spaces that take into account the interactions that humans usually perform with objects; the adaptation of human personal space to the environment in which it is located; and the definition of a human-robot interaction planning domain.

From the main results of this work it can be concluded that the new methodology improves the previous social navigation algorithm. First, a reliable detection and tracking, at real time, of people in the environment has been obtained. In addition, successful navigation results have been obtained in which the robot has socially navigated in the environment, taking into account the interaction spaces defined for humans, even in reduced scenarios (e.g., corridors). Finally, the proposed system has been achieved, through human-robot interactions, that the robot asks permission or collaboration to people in cases where the path is blocked.

A hypothesis was initially raised in this work, which said that it is possible to improve the human experience with an assistance robot in real environments with a robust social navigation system and a human-robot interaction planner for human-centered navigation. It can be concluded that this hypothesis is correct, since the results obtained in this work are proof of this.

Although in this work several ideas and improvements are presented for social navigation algorithms, the world of robotics is so broad that there is always a lot of problem to solve. Below are some possibilities that could be implemented in the future in the topic of social navigation for autonomous robots:

- Human information can come from several sensors, so it would be necessary to implement a way of fusing the information from all of them.
- In addition, other features, such as action recognition, emotions or facial recognition can be included.
- In this work only motionless humans are considered, that is, only static scenarios. However the speed of moving humans could be taken into account when planning the trajectory.
- In terms of interactive scenarios, it could be taken into account that the probability of interacting with certain objects varies throughout the day (for example, a person is more likely to interact with a refrigerator at lunchtime). This could be taken into account by varying the weights of the graph according to the time of day.
- In this work only three types of interaction between human and robot are considered. For proper social behavior, it would be necessary to increase the number of interactions. A possible example would be for the robot to approach the person and offer to exercise when it detects that the person has spent a lot of time at rest.

- In addition, certain actions such as guiding the person to a certain place, accompanying the person or following the person could be added to improve the social behaviour of the robot.

7.1 Contributions

The main contributions of this work have been publishes in international journals and conferences. As summary of these contributions is described bellow.

- *A. Vega, L. J. Manso Argüelles, R. Cintas Peña, and P. Núñez Trujillo, "Planning Human-Robot Interaction for Social Navigation in Crowded Environments: Proceedings of the 19th International Workshop of Physical Agents (WAF 2018)." , 2018, pp. 195-208.*

- *A. Vega, L. J. Manso Argüelles, P. Bustos García, and P. Núñez Trujillo, "A Flexible and Adaptive Spatial Density Model for Context-Aware Social Mapping: Towards a More Realistic Social Navigation," in Proceedings of the 15th International Conference on Control, Automation, Robotics and Vision, United States, 2018.*

- *A. Vega, L. Manso Argüelles, D. G. Macharet, P. Bustos García, and P. Núñez Trujillo, "Socially aware robot navigation system in human-populated and interactive environments based on an adaptive spatial density function and space affordances," Pattern Recognition Letters, 2018.*

Appendices

Appendix A

Personal Space calculation and clustering of people

This appendix explains how is calculated the personal space of humans and how the people clustering is performed in the social navigational algorithm presented in [1].

A.1 Personal Space Modelling

Let $S \in R^2$ be the space of the Global Map. An individual i is represented by its position $q_i = [x_i\ y_i]^T$ in S and it orientation $\theta_i \in [0, 2\pi]$. The personal space is modelled by a asymmetric 2-dimensional Gaussian function [12], which associates the distance between a point $p = [x\ y]^T \in S$ and the person's position with a real value $g_i \in [0, 1]$. The expression for the Gaussian function is shown in the eq. A.1:

$$g_i(x,y) = exp(-(a(x-x_i)^2 + b(x-x_i)(y-y_i) + c(y-y_i)^2)), \tag{A.1}$$

Where the coefficients a, b and c are used to take into account the orientation θ_i, and are defined by the relations:

$$a(\theta_i) = \frac{cos(\theta_i)^2}{2\sigma^2} + \frac{sin(\theta_i)^2}{2\sigma_s^2},$$

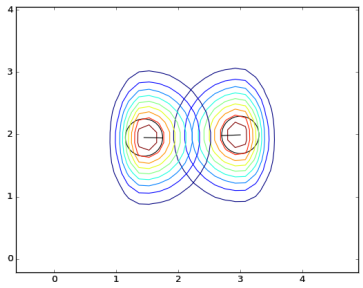
$$b(\theta_i) = \frac{sin(2\theta_i)}{4\sigma^2} - \frac{sin(2\theta_i)}{4\sigma_s^2},$$

$$c(\theta_i) = \frac{sin(\theta_i)^2}{2\sigma^2} + \frac{cos(\theta_i)^2}{2\sigma_s^2},$$

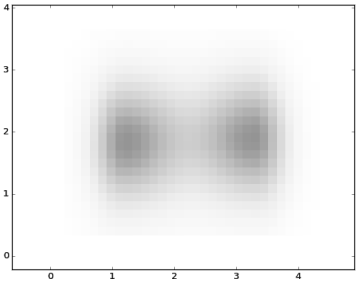
where σ_s is the variance to the sides ($\theta_i \pm \pi/2$ direction) and σ represents the variance along the θ_i direction (σ_h) or the variance to the rear (σ_r) [12]. Figure A.1b illustrates the personal space model of the two humans shown in A.1a.

The decision to use Gaussian functions to modelling personal spaces is very useful, since it allows for many established techniques to be used in this context. In this case, it will be used as the input of a global density function that clusters the individuals, as the next section explains.

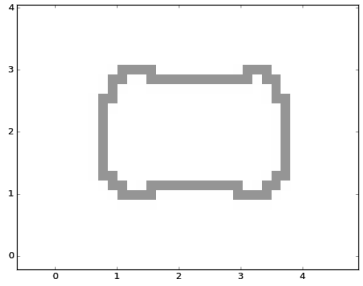
(a) example scenario



(b) Asymmetric Gaussian function



(c) Result of applying the density function



(d) Group contour

Figure A.1: Graphical representation of the results obtained applying the social navigation algorithm presented in [1].

A.2 Individuals Clustering

When considering groups of humans, it is needed to define how to associate the various personal spaces of each individual. This association is accomplished by performing

a Gaussian Mixture. The personal space function $g_i(p)$ of each individual i in the environment is summed and it arrives at a Global Space function $G(p)$. The proposal defines the Global Space Function:

$$G(x,y) = \sum_{i \in P} g_i(x,y). \tag{A.2}$$

Having performed the association and calculated the value of $G(p)$, the next step is to separate the individuals in groups. The method described discriminate the group contour to which each individual belongs, so it can define regions of forbidden navigation. This is accomplished by using the method described in [24].

This method was originally employed for grouping points in a point cloud to categorize them as to whether they belong to the same object. In essence, this method takes advantage of the property that, if for each point in a point cloud we associate a Gaussian function centered around it, then the closer two or more points are, the larger the sum of their respective Gaussian will be. This same line of reasoning can be used to group people into clusters which a robot can use to reason about space.

The method chooses the Ω parameter as the *smallest euclidean distance between two people $p_i, p_j \in P$ such that those two are neighbours*. This value is given by the insights of proxemics. If p_i, p_j are neighbours, then $\|p_i, p_j\| \leq \Omega$, and the density contribution δ between them is

$$\delta = g_i(p_j). \tag{A.3}$$

Since $g_i(q_i) = 1$ for each $q_i \in P$, then if q_i has k neighbours then $G(q_i) \geq 1 + k\delta$. Therefore, in order to group individuals who have at least k neighbours, the method can adjust a density threshold ϕ given by

$$\phi = 1 + k\delta, \tag{A.4}$$

and it can compare the value of the Global Function for each point in S and determine whether that point belongs to the personal space of a group of individuals. The set of

such points is denoted by J and given by the expression

$$J = \{p \in S \mid G(p) \geq \phi\}. \tag{A.5}$$

By manipulating the value of ϕ either by setting it directly or by manipulating the value of δ, it is able to control how near or far the border of J is in relation to each human in the cluster. In the method described the threshold has been fixed to $\phi = 0.4$, so that the border of the forbidden region is not immediately adjacent to an individual.

Finally, the contours of these forbidden regions are defined by a set of k polygonal chain (*i.e.*, polyline) $L_k = \{l_1, ..., l_k\}$, where k is the number of regions detected by the algorithm. The curve l_i is described as $l_i = \{a_1, ..., a_m\}$, being $a_i = (x, y)_i$ the vertices of the curve, which are located in the contour of the region J. The number of vertices, m, is dynamically adjusted by the algorithm, being the Euclidean distance between two consecutive vertices, $d(a_i, a_j)$, less than 10 cm. The results of applying the density function are shown in Fig. A.1c and the final polyline obtained is shown in Fig,.A.1d.

Bibliography

[1] Araceli Vega Magro. Desarrollo de un algoritmo de navegación social para robots autónomos móviles. In *Trabajo Fin de Grado, Universidad de Extremadura*, 2017.

[2] E. T. Hall. *The Hidden Dimension: Man's Use of Space in Public and Private.* The Bodley Head Ltd, 1966.

[3] Konstantinos Charalampous, Ioannis Kostavelis, and Antonios Gasteratos. Robot navigation in large-scale social maps: An action recognition approach. *Expert Systems with Applications*, 66:261–273, 2016.

[4] Jorge Alberto Rios-Martinez. Socially-Aware Robot Navigation: combining Risk Assessment and Social Conventions. *Hal.Inria.Fr*, 2013.

[5] Thibault Kruse, Amit Kumar Pandey, Rachid Alami, and Alexandra Kirsch. Human-aware robot navigation: A survey. *Robotics and Autonomous Systems*, 61(12):1726–1743, 2013.

[6] Jorge Rios-Martinez, Anne Spalanzani, and Christian Laugier. From proxemics theory to socially-aware navigation: A survey. *International Journal of Social Robotics*, 7(2):137–153, 2015.

[7] Konstantinos Charalampous, Ioannis Kostavelis, and Antonios Gasteratos. Recent trends in social aware robot navigation: A survey. *Robotics and Autonomous Systems*, 93:85–104, 2017.

[8] Jonathan Mumm and Bilge Mutlu. Human-robot proxemics: physical and psychological distancing in human-robot interaction. pages 331–338, 2011.

[9] Michael L Walters, Mohammedreza A Oskoei, Dag Sverre Syrdal, and Kerstin Dautenhahn. A long-term human-robot proxemic study. In *RO-MAN, 2011 IEEE*, pages 137–142. IEEE, 2011.

[10] Panagiotis Papadakis, Patrick Rives, and Anne Spalanzani. Adaptive spacing in human-robot interactions. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 2627–2632. IEEE, 2014.

[11] Panagiotis Papadakis, Anne Spalanzani, and Christian Laugier. Social mapping of human-populated environments by implicit function learning. In *Intelligent robots and systems (IROS), 2013 IEEE/RSJ international conference on*, pages 1701–1706. IEEE, 2013.

[12] Rachel Kirby. *Social robot navigation*. PhD thesis, Carnegie Mellon University, The Robotics Institute, 2010.

[13] Gian Diego Tipaldi and Kai Oliver Arras. Planning problems for social robots., 2011.

[14] Gian Diego and Tipaldi Kai O Arras. Please do not disturb! minimum interference coverage for social robots, 2011.

[15] Ioannis Kostavelis, Andreas Kargakos, Dimitrios Giakoumis, and Dimitrios Tzovaras. Robot's workspace enhancement with dynamic human presence for socially-aware navigation. In *International Conference on Computer Vision Systems*, pages 279–288. Springer, 2017.

[16] Ioannis Kostavelis. Robot Behavioral Mapping: A Representation that Consolidates the Human-robot Coexistence". *Robotics and Automation Engineering*, 1:1–3, 2017.

[17] Harmish Khambhaita and Rachid Alami. Assessing the social criteria for human-robot collaborative navigation: A comparison of human-aware navigation planners. In *International Symposium on Robot and Human Interactive Communication (RO-MAN) 2017 26th IEEE*, pages 1140–1145, 2017.

[18] Leia Stirling Vaibhav Unhelkar, Claudia Pérez-D'Arpino and Julie A Shah. Human-robot co-navigation using anticipatory indicators of human walking motion. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6183–6190, 2015.

[19] Frank Broz. Planning for human-robot interaction: representing time and human intention. Carnegie Mellon University, 2008.

[20] Claudia Esteves, Gustavo Arechavaleta, and J-P. Laumond. Motion planning for human-robot interaction in manipulation tasks. In *IEEE International Conference on Mechatronics and Automation*, volume 4, pages 1766–1771, 2005.

[21] Spruiell M Henning, M. *Distributed Programming with Ice*. 2009.

[22] Luis Vicente Calderita. *Deep State Representation: a Unified Internal Representation for the Robotics Cognitive Architecture CORTEX, PhD Thesis, University of Extremadura*. PhD thesis, 2015.

[23] Lorenz Mösenlechner Michael Beetz, Dominik Jain and Moritz Tenorth. "towards performing everyday manipulation activities". 58(9):1085–1095, 2010.

[24] Antonio W Vieira, Paulo LJ Drews, and Mario FM Campos. Spatial density patterns for efficient change detection in 3d environment for autonomous surveillance robots. *Ieee Transactions on Automation Science and Engineering*, 11(3):766–774, 2014.

[25] E. Olson, J. Leonard, and S. Teller. Fast iterative alignment of pose graphs with poor initial estimates. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 2262–2269, 2006.

[26] Steven M. LaValle. *Planning algorithms*. 2006.

[27] M. Haut, L. Manso, D. Gallego, M. Paoletti, P. Bustos, and A. Bandera, A. Romero. A navigation agent for mobile manipulators. In *Proceedings of the Robot 2015: Second Iberian Robotics Conference*, pages 745–756, July 2016.

[28] Orbbec astra website, `https://orbbec3d.com/product-astra/`, last accessed 22/01/2019.

[29] Opencv website, `https://opencv.org/`, last accessed 23/01/2019.

[30] Pablo Bustos García, Luis Jesús Manso Argüelles, Pilar Bachiller Burgos, and Pedro Núñez Trujillo. Navigation among people with cortex. In *REACTS workshop at the International Conference on Computer Analysis and Patterns, CAIP*, Ystad Saltsjöbad, 2017.

[31] Billy Okal and Kai O Arras. Learning socially normative robot navigation behaviors with bayesian inverse reinforcement learning. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 2889–2895. IEEE, 2016.