HAL
open science

User Intention Estimation for Semi-Autonomous Navigation of a Robotic Wheelchair

Jesús-Arturo Escobedo-Cabello

To cite this version:

Jesús-Arturo Escobedo-Cabello. User Intention Estimation for Semi-Autonomous Navigation of a Robotic Wheelchair. Other [cs.OH]. Université de Grenoble, 2014. English. NNT : 2014GRENM098 . tel-01547229

HAL Id: tel-01547229

https://theses.hal.science/tel-01547229v1

Submitted on 26 Jun 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Informatique**

Arrêté ministériel : 7 août 2006

Présentée par

Jesús-Arturo ESCOBEDO-CABELLO

Thèse dirigée par **Christian Laugier**
et codirigée par **Anne Spalanzani**

préparée au sein du centre de recherche **INRIA Rhône-Alpes du Laboratoire d'Informatique de Grenoble**
dans l'Ecole Doctorale **Mathématiques, Sciences et Technologies de l'Information, Informatique**

User Intention Estimation for Semi-Autonomous Navigation of a Robotic Wheelchair.

Thèse soutenue publiquement le **03 Octobre 2014**,
devant le jury composé de :

James L. CROWLEY
Professeur, INPG, France, Président
Rachid ALAMI
Directeur de Recherche, LAAS-CNRS, France, Rapporteur
Marie BABEL
Maître de Conférences, INSA, Rennes, France, Rapporteur
David DANEY
Chargé de Recherche, Inria, Bordeaux, Examinateur
Christian LAUGIER
Directeur de Recherche, INRIA Rhône-Alpes, France, Directeur de thèse
Anne SPALANZANI
Maître de Conférences, UPMF, Grenoble, France, Co-Directeur de thèse

Acknowledgements

Finishing this thesis represents a big satisfaction both professional and personal to me. It has been a complete life experience and a rich process that provided me great experiences. Even if sometimes I didn't think I would be able to go to till the end, I am so proud of myself for being here today.

Words cannot express how grateful I am to my mother, and father for all of the sacrifices that you've made on my behalf. Your prayer and love for me has always been my sustain and inspiration to keep growing. Thanks for my life and thanks for being amazing parents. Thanks to Laura and Alfredo, my beloved sister and brother for your love and for being my best friends in the whole world. Thanks to my uncles Paty and Carlos, because whenever I think in someone nice and happy you are there, thanks for being my family and sharing with me so many trips, so many beautiful conversations, so many laughs, thanks for allowing me to stay at your home whenever I needed it to continue my studies.

I want to acknowledge to all the people that collaborated in the fulfillment of this project. Thanks to my E-motion team because all of you have been there to support me when I needed some technical or personal advice during my Ph.D. thesis.

My special appreciation and thanks to my advisers Anne Spalanzani and Christian Laugier for being such tremendous mentors for me, thanks for encouraging me to finish and being so patient during all these years.

Thanks to Jorge Rios and Karla Loeza for being an amazing family, your great sense of humour, nice conversations and food helped me to be happy even at the hardest moments. Mathias Perrollaz thanks for being such a crazy friend and adviser. Chiara Troiani for all those smiles, delicious cakes and kind support you gave me all those years.

Jorge, Procopio, and Gregoire, the wheelchair team, thanks for your great work that helped me to present this thesis. Thanks to Stéphanie Lefèvre for reviewing my work and teaching me Bayesian Networks. Christelle Gaidatzis

thanks for being a great sister and for helping me with the french part of this thesis. Thanks to Miseon Kim for helping me with some figures in this manuscript.

I would also like to thank all of my friends who supported me in writing, and pushed me to strive towards my goal Rocío, Wendy, Michelle, Andra, Olga, Arnaud, Nahuel, Will and Gerardo. Thanks to Viviane Gamboa for giving me such a valuable support and personal couching during the most difficult times of my thesis. I would not have finished without your many advices and help.

Finally, thanks to my committee members, for your comments and suggestions that helped me to improve the final version of this manuscript. Thanks to David Daney the manager of the PAL project that allocated the economic and logistic resources to fund this work. Thanks also to the Mexican Council of Science CONACYT for the economic resources dedicated to this thesis.

Abstract

This thesis focuses on semi-autonomous wheelchair navigation. We aim to design a system respecting the following constraints.

- **Safety :** The system must avoid collisions with objects and especially with humans present in the scene.
- **Usability :** People with motor disabilities and elders often have problems using joysticks and other standard control devices. The use of more sophisticated and human-like ways of interacting with the robot must be addressed to improve the acceptance and comfort for the user. It is also considered that the user could just be able to move one finger and so the request of human intervention should be as reduced as possible to accomplish the navigation task.
- **Compliance :** The robot must navigate securely among obstacles while reducing the frustration caused to the user by taking into account his intentions at different levels ; final destination, preferred path, speed etc.
- **Respect of social conventions :** When moving, the robot may considerably disturb people around it, especially when its behavior is perceived as unsocial. It is thus important to produce socially acceptable motion to reduce disturbances. We will also addresses the issue of determining those places where the robot should be placed in order become part of an interacting group.

In this work we propose to estimate the user's intention in order to reduce the number of necessary commands to drive a robotic wheelchair and deal with ambiguous or inaccurate input interfaces. In this way, the wheelchair can be in charge of some part of the navigation task and alleviate the user involvement. The proposed system takes into account the user intention in terms of the final destination and desired speed. At each level, the method tries to favor the most "reasonable" action according to the inferred user intention.

The user intention problem is approached by using a model of the user based on the hypothesis that it is possible to learn typical destinations

(those where the user spends most of his time) and use this information to enhance the estimation of the destination targeted by the user when he is driving the robotic wheelchair.

A probabilistic framework is used to model the existent relationship between the intention of the user and the observed command. The main originality of the approach relies on modeling the user intentions as typical destinations and the use of this estimation to check the reliability of a user's command to decide how much preeminence it should be assigned by the shared controller when managing the robot's speed.

The proposed shared-control navigation system considers the direction of the commands given by the user, the obstacles detected by the robot and the inferred destination to correct the robot's velocity when necessary. This system is based on the dynamic window approach modified to consider the input given by the user, his intention, the obstacles and the wheelchair's dynamic constraints to compute the appropriate velocity command.

All of the results obtained in this thesis have been implemented and validated with experiments, using both real and simulated data. Real data have been obtained on two different scenarios ; one was at INRIA's entry hall and the other at the experimental apartment GERHOME.

Résumé

L'objet de cette thèse est la navigation en fauteuil roulant semi-autonome. Nous cherchons à concevoir un système respectant les contraintes suivantes :

- **Sécurité :** Le système doit éviter les collisions avec d'autres objets et surtout avec toute personne présente sur la scène.
- **Facilité d'utilisation :** les personnes âgées ou avec un handicap moteur ont souvent des difficultés à utiliser une manette ou d'autres dispositifs de contrôles standards. Des moyens d'interactions avec le robot, plus sophistiqués et plus proches des interactions humaines, doivent donc être mis en place afin d'améliorer l'acceptation et le confort de l'usager. On considère également que l'utilisateur ne doit avoir qu'à bouger un doigt : de cette manière, la nécessité d'une intervention humaine sera aussi réduite que possible pour accomplir la tâche de navigation.
- **Acquiescement :** le robot doit naviguer en toute sécurité parmi les obstacles, tout en réduisant la possible frustration causée à l'utilisateur en prenant en compte ses intentions à différents niveaux : destination finale, chemin préféré, vitesse, etc.
- **Respect des Conventions Sociales :** en se déplaçant, le robot peut considérablement déranger les personnes qui se trouvent autour, tout particulièrement lorsque son comportement est perçu comme associable. Il est par conséquent important que les mouvements du robot soient socialement acceptables afin de réduire les possibles gênes. Nous déterminerons également les lieux o doit être placé le robot afin de s'intégrer au mieux à un groupe en interaction.

Dans ce travail, nous proposons d'évaluer les intentions de l'utilisateur afin de réduire le nombre de commandes nécessaires à la conduite d'un fauteuil roulant robotisé Nous souhaitons prendre en compte toutes les interfaces d'entrée ambiguës ou imprécises. De cette manière, le fauteuil roulant peut être responsable d'une partie de la tâche de navigation et alléger l'implication de l'usager. Le système proposé prend en compte l'intention de l'utilisateur en termes de destination finale et de vitesse souhaitées. A chaque

niveau, la méthode cherche à favoriser l'action la plus "raisonnable" en fonction des intentions inférées de l'usager.

Ainsi, à partir du panel des destinations types que l'on apprend de l'usager (c'est à dire les lieux dans lesquels celui-ci passe la plus grande part de son temps), il est possible de déduire sa destination lorsqu'il conduit son fauteuil roulant. Une méthode probabiliste est utilisée pour modéliser la relation entre l'intention de l'usager et la commande observée. L'apport principal de cette méthode consiste à modeler l'intention de l'usager à partir de destinations types. Cette information est utilisée pour valider les commandes de l'utilisateur et assigner une importance plus ou moins grande au système de contrôle partagé qui régule la vitesse du robot.

Le système de contrôle partage prend en compte la direction des commandes données par l'usager, les obstacles détectés par le robot et la destination inférée afin de calculer la vitesse du robot la plus appropriée. Ce système se base sur une adaptation de la méthode dynamic window à la problématique décrite.

Tous les résultats obtenus dans cette thèse ont été appliqués et validés en simulation et en conditions réelles. Les mesures ont été effectuées sur deux scènes : la première située dans le hall d'accueil du laboratoire INRIA et la seconde dans le département expérimental GERHOME.

Contents

Contents vii

List of Figures xi

Nomenclature xviii

1 Introduction 1

1.1 Motivation . . . 1
1.2 Problem Description . . . 2
1.3 System Architecture . . . 3
1.4 Contributions . . . 5
1.5 Thesis outline . . . 6
1.6 Publications . . . 7

2 State of the Art 9

2.1 Introduction . . . 9
2.2 Mechanical Design Classification . . . 11
2.3 Input Devices . . . 11
2.3.1 Continuous Input Devices . . . 13
2.3.2 Discrete Input . . . 14
2.3.3 User-Machine Interfaces Used by Robotic Wheelchairs . . . 15
2.4 Operating Modes . . . 19
2.4.1 Manual Control . . . 20
2.4.2 Shared Control . . . 20
2.4.3 Semi-Autonomous Control . . . 22
2.4.4 Autonomous Control . . . 23
2.5 User Intention Estimation . . . 23
2.6 Human Aware Navigation . . . 24

2.7 Approaching Humans 25
2.8 Discussion and Thesis Positioning 26

3 Tools for Navigation in Human Populated Environments 29
3.1 Introduction 29
3.2 Social Cost Mapping 32
3.2.1 Introduction to Occupancy Cost Mapping 33
3.2.2 Personal Space 36
3.2.3 Interaction Space 37
3.2.4 Representing Social Spaces as Cost Grids 41
3.3 Meeting Points Computing 42
3.4 Navigation System 43
3.4.1 Trajectory Control 43
3.4.2 A* Path Planning 44
3.4.3 Wheelchair's Self Localization 49
3.4.4 Velocity Controller 50
3.5 Simulation Results 52
3.6 Discussion 53

4 User Intention Aware Navigation 57
4.1 Introduction 57
4.2 Destination Inference Subsystem 59
4.2.1 Dynamic Bayesian Network to Infer the User Desired Destination 60
4.2.1.1 Command Model 63
4.2.1.2 Transition Model 64
4.2.1.3 Initial Probability Model 64
4.2.1.4 Goal Selection 64
4.2.2 Simulation Results 65
4.2.3 Limitations of the System. 67
4.2.4 Extension of the System to Consider Dynamic Destinations . . . 69
4.2.5 Discussion 74
4.3 Shared Local Controller 74
4.3.1 The Dynamic Window Approach 75
4.3.1.1 Cost Function 78
4.3.2 User Intention Aware Dynamic Window 80
4.3.2.1 Mapping from directional orders to speed commands . . 81

4.3.2.2 Modified cost function 83
4.3.3 Simulation Results 84
4.3.4 Discussion 87

5 Experiments **91**
5.1 Experimental Setup 91
5.1.1 The Experimental Robotic Wheelchair 91
5.1.2 INRIA-hall 93
5.1.3 Gerhome 95
5.2 User-Robot Interfaces 96
5.2.1 Face Pose Estimation System 96
5.2.1.1 Turning in place 99
5.2.2 Voice Recognition 100
5.2.3 3D Human Tracker 102
5.2.3.1 Arm Gesture Recognition 104
5.2.4 Keyboard 104
5.3 Multimodal Control using Face Pose and Speech Recognition 105
5.3.1 Test 1 : Driving with Destination Inference Method vs Driving without Assistance 105
5.4 Destination Inference with Dynamic Destinations 107
5.5 Caregiver Following 112
5.6 Discussion 113

6 Conclusions and perspectives **115**
6.1 Meeting Points Computing 116
6.2 Multimodal Interface 116
6.3 Respect of Social Conventions 116
6.4 Final Remarks 116

A Appendix **119**
A.1 Introductory Concepts of Probability 119
A.1.1 Logic, Probability and Reasoning under Uncertainty 119
A.1.2 Probability of a Proposition 120
A.1.3 Probability Theorems 121
A.1.3.1 The Product Rule 121
A.1.3.2 The Normalization Rule 121
A.1.3.3 The Sum Rule 121

A.1.4 Theorem of Total Probability 122
A.1.5 The Bayes Rule 122
A.1.6 Conditioning on Other Variables 122
A.1.6.1 Internal Variables 123
A.1.6.2 Conditional Independence of Variables 124
A.1.7 Recursive Calculation 124
A.1.8 State Filtering 124
A.1.9 Bayesian Networks 125

List of Figures

1.1 The navigation system of the robotic wheelchair. Several functional components were developed as part of this thesis (orange items) while the other necessary modules were taken from the open source community (white blocks). In this diagram we used standard flow chart symbols to represent information that is stored on disk (blue cylinders), manual data inputs coming from the user (rectangles with diagonal top), and data from other sources (parallelogram). 4

2.1 The autonomous wheelchair prototype from Arizona State University (1986) was the first project that try to use a mobile robot as a wheelchair to assist people with disabilities. 10

2.2 Robotic wheelchairs built from standard electric wheelchairs with added sensors and computing units. 12

2.3 Wheelchairs built from autonomous robots with added seats. 13

2.4 The joystick is the most common access method or input interface with an electric wheelchair. 14

2.5 In a typical head control the user pushes the head support in the desired direction to control the wheelchair. 14

2.6 These types of input devices can be operated by users who can only move one finger. 15

2.7 Different discrete access methods used in commercial electric wheelchairs. 16

2.8 Brain-computer Interfaces used to drive robotic wheelchairs 17

2.9 Shared control paradigm : Both the robotic wheelchair and the user collaborate to accomplish the navigation task. The robot has a predefined model of both the world and the user that will produce the desired level of autonomy to the user. The user contributes to the controlling task by perceiving the results of the actions taken by the shared controller through his senses and applying a regulatory input signal according to his intentions. 21

3.1 The navigation system of the robotic wheelchair. In this section we will explain the functional modules developed as part of this thesis (orange). The components of the system in gray will be explained as they are important for the overall functioning of the system but they were taken from the open source community or they are previous works developed at our research team. 31

3.2 2D occupancy grid generated by the robotic wheelchair, the yellow dots are laser impacts. The corresponding cells are marked as an obstacle (black squares). The gray cells are non-observed. 34

3.3 Inflation is the process of propagating cost values out from occupied cells (red) that decrease with distance. Cells in the inscribed region are assigned a uniformly high cost (inscribed cost), after which an exponential decay function (Eq. 3.2) is applied that will cause the cost to drop off out to the inflation radius used for the cost map. 35

3.4 Personal space calculated by Social Filter Module 37

3.5 Most typical F-formation types. 38

3.6 Example of the elements in the geometrical o-space model for a vis-a-vis and a L-shape formation. 40

3.7 Interaction space computed by the Social Filter functional module for different formations. The maximum risk of disturbance is located at the o-space center, the disturbance is represented by the height of the Gaussian functions (blue regions). 40

3.8 Threshold applied to the Gaussian functions delivered by the social filter to created the occupancy-cost grid. (b) The inflated area (green) considers the dimensions of the wheelchair. In order to guarantee a path free of collisions the center of the wheelchair should never be inside this area. 42

3.9 Projection of both p-space and o-space on the occupancy grid used by the planning algorithm. The Gaussian in the middle represents the value of the interaction space, which is projected on the occupancy grid as occupied space (blue area) and then inflated to consider the size of the robot (green area). 42
3.10 Meeting points 43
3.11 The resulting meeting points for a vis-vis formation (black markers). If the robot reaches that position, a group of three will be formed. 44
3.12 Path planning general problem description. The green cell P_0 is the starting point, the red one P_N is the desired ending point. Black cells are the obstacles while white/gray cells are free space. 45
3.13 A* algorithm : Initializing the search. 47
3.14 A* algorithm (Main loop example). The process finishes when the target square (red) is added to the closed list (blue boundary) (g), in which case the path has been found, or it fails to find the target square and the open list is empty. In this case, there is no path. 48
3.15 A* algorithm (trace the path). To determine the path, just start at the end node and go backwards moving from one square to its parent, following the arrows. 50
3.16 Advance Montecarlo Localization. Given a map of the environment, the position of the robot is modeled as a set of particles (red arrows) representing each of the possible robot states. The state with the highest likelihood is taken as the current position of the robot (where the wheelchair is actually located). 51
3.17 Differential-drive model showing the important geometrical characteristics of the robot, the reference axes and the different robot velocities. X_R , Y_R and Z_R corresponds to the robot referential system ; v, ω are the robot translational and rotational velocities. $2b$ is the distance between the two wheels and r the radius of each wheel. 52
3.18 Comparison between the standard 2D occupancy grid provided in ROS (a), (b) and the one improved with social spaces (c), (d). 53
3.19 Path planning using the social occupancy grid. The robot is asked to go to the destination marked with a red arrow, it then plan a path avoiding the occluded areas (green). The A* path planner used is an open-source implementation provided as part of the ROS navigation stack (Open-Source-Robotics-Foundation, 2014). 54

4.1 In this section we will present the destination inference and shared local controller subsystems -red blocks- of the robotic wheelchair. Both of blocks consider the input from the user. (In this figure we used general flow-chart symbols as cylinders for data-storage, squares for functional blocks and squares with a diagonal top to depict user entries). 59

4.2 The Bayesian Network used to estimate the current user intended destination D_t. At each time step t, the posterior probability is updated by using the current position X_t and current command C_t. D_t also depends on the value of the last estimation D_{t-1} to take into account the history of given commands. Prior knowledge is expressed as the probability of going from each starting position X_0 to any of the possible destinations D_0. Hidden variables are in blue and observable states are dashed circles. 61

4.3 Command model. The probability value for a given command C_t is proportional to the angle $a^{(i)}$ between the sight-line and each typical destination $d^{(i)}$ in the environment. 63

4.4 The GERHOME simulated environment. (a) Simulation in STAGE. (b) Map of the environment with the typical destinations marked with circles. The size of the circles represents the value of the initial probability distribution for the given starting point $X_0 = d^{(10)}$, here we can see that according to the table the destination with the highest initial probability is $d^{(1)}$ followed by $d^{(9)}$. (c) Initial probability table. In practice, those elements with value equal to 0 are replaced with a very small value ϵ. . 66

4.5 Moving towards a typical destination. In this example the user commands the wheelchair to go from the door $d^{(10)}$ to the table in the left $d^{(1)}$ or the bedroom to his right $d^{(9)}$. In the starting state the destinations have a similar prior probability value ((a), (d)). When the user sends the initial GO signal, the probabilities are updated according to the direction pointed by the user and the best destination is sent to the robot ((b), (e)). The robot moves autonomously towards the goal ((c),(f)). . . 67

4.6 The user is asked to move the wheelchair from the door $d^{(10)}$ to the sink $d^{(7)}$ (which is not a frequent destination according to the user's habits). To deal with this kind of situations the user will have to give more than one command but anyway the number of necessary commands to arrive to a given destination will be reduced. 68

4.7 The Dynamic Bayesian Network used to estimate the current user intended destination D_t. The initial state of the DBN depends on the starting position X_0 and initial vocal command V_0. At each time step t, the posterior probability is updated by using the current position X_t and current command C_t. D_t also depends on the value of the last estimation D_{t-1} to take into account the history of given commands. 70
4.8 In this example ; even if the pointed direction is ambiguous between $d^{(9)}$ and $d^{(11)}$, when the user sends the JOIN initial order, the probabilities are computed in such a way that those destinations defined by a group of people get a higher value. 73
4.9 The GO initial order does not give as much contextual information as the verb JOIN ; therefore, the destination that gets the maximum probability is mainly defined by the direction pointed by the user. 73
4.10 Trajectories produced by the dynamic window algorithm and the corresponding velocity search space. 76
4.11 The cost function $dist(v, \omega)$ considers the minimum distance before colliding with an obstacle over the sampled trajectory. 77
4.12 The heading cost reinforces trajectories that moves directly towards the goal. The angle θ is relative to the robot's heading direction in the last point of the simulated trajectory and the goal. 79
4.13 The cost function proposed by (Marder-Eppstein et al., 2010). It takes into account the distance to the global path, the distance to the goal and distance to obstacles to score a trajectory. The function $pDist$ measures the distance to the global path from the end point of the trajectory in map cells or meters, $gDist$ is the distance to the goal from the endpoint of the trajectory. The $occCost$ function returns the maximum obstacle cost along the trajectory. 81
4.14 The position of the joystick gimball is translated to a velocity command $(v_{user}, \omega_{user})$. 82
4.15 The position of the head is translated to a velocity command $(v_{user}, \omega_{user})$. 83
4.16 The cost term $diff(v, \omega)$ is added to the dynamic window cost function in order to take into account the input from the user. 83
4.17 Avoiding an obstacle 85
4.18 Docking Case. 85
4.19 Thanks to the algorithm the robot will not move until it receives a valid/safe command. 86

4.20 The system only corrects the command given by the user when it is necessary. In the scene, detected obstacles are represented by green zones, the command given by the user is a red arrow and the trajectory followed by the robot is the red line. 87

4.21 Using the shared local controller to follow a person. The target person is shown with a red marker, the generated trajectories are the shown as colored arcs (red=good score, green=bad score, black=not valid). . . . 88

4.22 Possible behaviors when using the shared trajectory controller without the destination inference system. 89

5.1 The INRIA Rhône-Alpes robotic wheelchair. 92

5.2 INRIA hall experimental scenario. (a) The lines in the figure represent some of the trajectories normally followed by people. Typical destinations (end of a trajectory) are marked with circles. (b) Typical destinations at INRIA Rhône-Alpes hall. (c) The map of the scenario used by the navigation system. 93

5.3 Initial probability distribution obtained by observing the human activity at INRIA's entry hall. 94

5.4 The Gerhome platform. The map and the list of sensors distributed in the building is presented. 95

5.5 An example of video sequence and the corresponding events detected by the sensors at Gerhome apartment. 96

5.6 3D point cloud image of the user's face is used to estimate its direction. 97

5.7 Face pose recognizer output angles. Yaw face's angle can be measured for movements in the range $[-45°, 45°]$, $pitch \in [-30°, 30°]$ and $roll \in [-6°, 6°]$ 97

5.8 The face pose recognizer system publishes a transformation between the camera reference frame and the head reference frame. Assuming that the camera will have a fixed position with respect to the wheelchair's reference frame, it can be computed the transformation between the head pose and the wheelchair's pose. 98

5.9 The user receives some indications to look around while the wheelchair is in turning mode. The wheelchair turns around according to the user's face position. It is a comfortable way of acting as long as it does not hit obstacles. 99

5.10 In this test the user drives the wheelchair by showing his desired travel direction with the face. The wheelchair performs obstacle avoidance using the presented shared local controller. Stop and Go commands are performed through the speech recognition system. The full video can be accessed at http ://youtu.be/O3IPISn7waw. 100
5.11 Driving the wheelchair using vocal commands, we see the map showing the wheelchair and the list of vocal commands recognized by the speech recognizer. The video corresponding to this image sequence can be watched at http ://youtu.be/w5VMVrXhzfg. 101
5.12 Driving the wheelchair using vocal commands, we see the map showing the wheelchair and the list of vocal commands recognized by the speech recognizer. The video corresponding to this image sequence can be watched at http ://youtu.be/w5VMVrXhzfg. 102
5.13 The OpenNI skeleton tracker uses a 3D point cloud to estimate the position (referent to the camera) of the tracked user. 103
5.14 The skeleton tracker can be used to drive the wheelchair using gestures. In the scene our experimental robotic wheelchair is driven using hand-gestures. The video corresponding to this image sequence can be watched at http ://youtu.be/w5VMVrXhzfg. 104
5.15 Using face pose to estimate the user's desired destination. (Video at http ://goo.gl/VvtezQ). 107
5.16 Some samples of the logged trajectories are presented. (a) and (c) show the results when using the assistance of the user intention estimation system. (b) and (d) were achieved by driving the wheelchair using the face without any assistance. Here we can observe the oscillations in the trajectory due to involuntary changes in the face direction produced when observing the surroundings. 108
5.17 Experimental Evaluation, the user is asked to go from $d^{(7)}$ to $d^{(1)}$ using the user intentions system. (a) As the destination with the highest prior probability from $d^{(7)}$ is $d^{(5)}$ in the beginning of the test the results are biased to that side that is why $d(3)$ is initially chosen as the most likely goal. (b) As the user keeps looking to the left all the destinations in that direction become more likely. (c) When the wheelchair has enough evidence, it changes the desired destination to $d^{(1)}$. (d) The user arrives to his desired destination, the spot in the middle of the trajectory marks the place were the change of destination was produced. 110

5.18 Example of an experimental scenario where some typical destinations and meeting points have been detected. In the intended application, the user points in the direction of his desired destination and use a vocal command to order the wheelchair to move. 111
5.19 Error handling. When the system detects an error in the inferred destination (it is blocked or there is not enough evidence to decide) (a) it asks for help to the user (b). The user gives a new command (c) and if the new destination is valid it will move towards it. 111
5.20 A scene of the robotic wheelchair following the user. This is an important characteristic that a robotic wheelchair should pursuit to ease the job not only for the user but also for the medical staff an his family. 112

Chapter 1

Introduction

1.1 Motivation

The aging of the world's population is bringing the need to assist the elderly to move when they lose the necessary motor skills, strength or visual acuteness to do it by themselves (Simpson et al., 2008). Mobility limitations are a leading cause of functional disabilities among adults and it is related to depression and anxiety disorders (Pope and Tarlov, 1991).

Self-sufficient mobility reduces dependence on caregivers and family members and it has a considerable effect on social inclusion, self-esteem and vocational opportunities for both adults and children (Mortenson et al., 2012). As a result of the reduction in medical dependence, it also plays an essential role in promoting the "aging in place" that aims to support daily activities at home as long as possible.

Mobility can be improved by using typical devices including powered and manual wheelchairs, scooters and walkers. However, there exist a segment of the population who still find it very demanding or even unfeasible to use those normal devices. This population includes patients with spasticity [1], tremors, hemiplegia [2] or cognitive deficits. The development of smart robotic platforms capable of providing an specialized and intelligent assistance can clearly improve the quality of life of this sector of the population.

The PAL project

1. Spasticity is a disease causing stiff or rigid muscles. It may also be called unusual tightness or increased muscle tone. Reflexes (for example, a knee-jerk reflex) are stronger or exaggerated. The condition can interfere with walking, movement, or speech.

2. Hemiplegia is the total or partial paralysis of one side of the body that results from disease or injury to the motor centers of the brain

The work presented in this thesis has been done as part of the so called PAL project. The Personally Assisted Living project (PAL) [1] is an INRIA [2] project grouping together many INRIA research teams. This project proposes to develop technologies and services to improve the autonomy and quality of life for elderly and fragile persons covering a broad spectrum of research axis such as: social interaction, intelligent habitat, mobility assistance, health care and well being.

Fundamental and technological research particularly related to autonomous indoor vehicles is concerned with the development of mobility assistance devices such as robotic wheelchairs and walking aids. The large scale nature of the project leads the development of other technologies to track the daily activities of people at home such as sensor and actuators networks, and human-robot interaction technologies to ease the interaction of the elderly with new technologies.

1.2 Problem Description

Conventional hand-operated wheelchairs provide mobility to persons with physiological impairments but are not very appropriate for someone with a combination of physical, cognitive or perceptual disability. Powered wheelchairs are less physically demanding than hand-operated wheelchairs; however, powered wheelchairs demand cognitive and physical abilities that not all persons possess. For example, a clinical study in a variety of rehabilitation services reported that between 10 and 40 percent of the interviewed patients could not use powered wheelchairs because sensory impairments, motor, or cognitive deficits made driving safely impossible with any of the existent mechanisms (Fehr et al., 2000). Nowadays, those who are considered unable to safely and independently maneuver a manual or powered wheelchair are typically seated in a manual wheelchair and pushed by a caregiver.

When working with a mobility assistance device there are at least two entities to be considered: the user and the assistance device itself. This kind of problem is referred as shared control. Researching on human-robot shared control systems is becoming very important because of the growing number of daily situations where cooperation between robots and humans is necessary to accomplish a certain task.

The purpose of shared control is to combine the strengths of both the human and the machine to reduce their weaknesses. Whereas humans excel in global planning

1. https://pal.inria.fr/
2. French: Institut National de Recherche en Informatique et en Automatique, (Institute for Research in Computer Science and Automation).

and coarse control, local navigation is one of the tasks that is performed better by robots. In this case, the robotic assistant could avoid embarrassing and potentially dangerous situations such as collisions with walls, other objects, and other persons in the environment. On the other hand, wheelchair drivers do not necessarily accept to give up driving control to the robot.

In order to succeed, each of the entities in a shared controlled system needs to have certain knowledge about the intention of the other parts. In a user-centered design it is necessary to reduce the frustration caused by disagreements with the robotic assistant. This is why the robot has to be aware of the intentions of his user at different levels.

To be well accepted, the movement of the wheelchair must be reliable, safe and comfortable. The approach discussed in this work has been designed in order to meet the following requirements:

- **Safety:** The system must avoid collisions with objects and specially with humans present in the scene.
- **Usability:** People with motor disabilities and elders often have problems using joysticks and other standard control devices. The use of more sophisticated and human-like ways of interacting with the robot must be addressed to improve the acceptance and comfort for the user.
- **Compliance:** The robot must navigate securely among obstacles while reducing the frustration caused to the user by taking into account his intentions at different levels; final destination, preferred path, speed etc.
- **Respect of social conventions:** When moving, the robot may considerably disturb people around it, especially when its behavior is perceived as unsocial. It is thus important to produce socially acceptable motion to reduce disturbances.

1.3 System Architecture

In order to meet the requirements mentioned in section 1.2, we created a system with a modular architecture to address each different objective separately, as shown in figure 1.1.

The functional modules developed as part of this thesis are shadowed in orange while the other necessary modules were taken from the open source community (white blocks). In this diagram we used standard flow chart symbols to represent information that is stored on disk (blue cylinders), manual data inputs coming from the user (rectangles with diagonal top), and data from other sources (parallelogram). A brief description of the essential modules in this thesis are now presented.

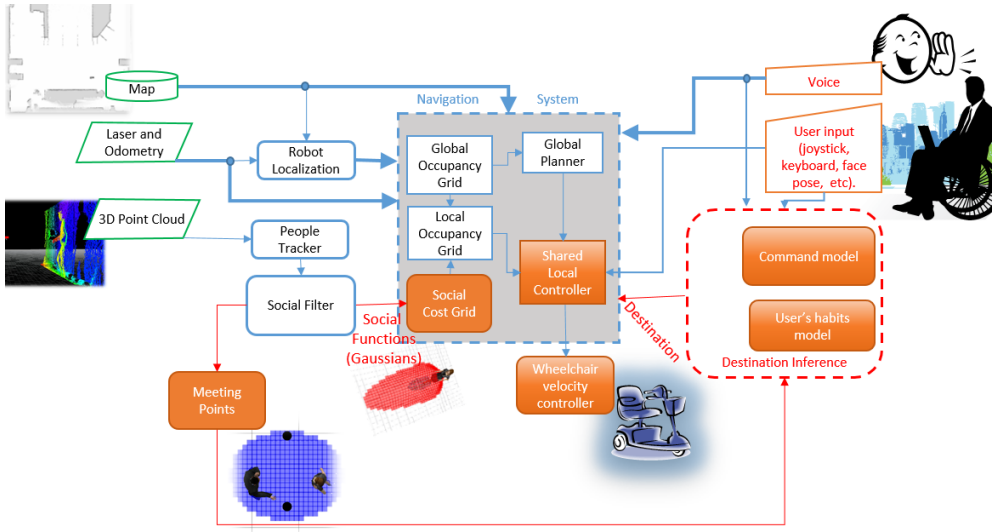


Figure 1.1: The navigation system of the robotic wheelchair. Several functional components were developed as part of this thesis (orange items) while the other necessary modules were taken from the open source community (white blocks). In this diagram we used standard flow chart symbols to represent information that is stored on disk (blue cylinders), manual data inputs coming from the user (rectangles with diagonal top), and data from other sources (parallelogram).

- **Destination Inference:** This functional component estimates the desired destination within the map of the environment among a list of possible predefined locations in the map. Those locations can be selected by an expert caregiver, the user, or learned automatically by the system. For each typical destination, the probability to be the one desired by the user is computed using a Dynamic Bayesian Network that considers the current position of the wheelchair, the direction of user's commands as well as a model of the user's habits. The destination with the highest probability value is selected and sent to the navigation module (Escobedo et al., 2013).
- **Meeting Points Estimator:** It uses a geometrical model to compute good places to approach groups of people in social interaction (meeting points). Those meeting points are a special kind of possible destinations which are not static but depend on complex social relationships governing human interactions. Whenever an interaction is detected by the social filter, the related meeting points are computed and considered by the destination inference as possible options.
- **Navigation:** The navigation component is built upon the ROS [1] navigation

1. Robot Operating System. http://www.ros.org/

software architecture presented in (Marder-Eppstein et al., 2010). It computes the path to the selected/inferred destination and calculates the necessary velocity to follow the path while avoiding obstacles. The modules that have been modified to meet the design requirements explained in the Problem Description section are:

- **Shared Local Controller:** This part of the navigation component is the one in charge of generating the velocity commands to drive the wheelchair and is one of the main contributions of the presented work. It is based on the dynamic window approach DWA (Fox et al., 1997) that performs reactive navigation of autonomous robots. Opposed to the conventional method our system takes the intention of the user into account preferring trajectories in the direction of the user field of view or in the direction of a deliberate order given by the user using a normal input device as a joystick, a keyboard, etc.
- **Social Grid:** Receives information from the "social filter" (Rios-Martinez et al., 2011) that models a social cost or risk of disturbing people in the vicinity of the robot. The social grid fuses the social cost information with the obstacle cost information retrieved by the wheelchair's sensors. Both costs are projected into a 2D grid representation of the space which is used when scoring the navigation function that governs the wheelchair's movement.

1.4 Contributions

The main contributions of the work reported in this thesis are the following:

- We proposed a "destination inference method". This is a probabilistic model that improves the inference of the desired destination by considering the user's habits and data coming from a multi-modal user-machine interface.
- We proposed a "user's intention driven navigation function" implemented using a modified version of the Dynamic Window Approach to consider the user's intention in the wheelchair's local velocity control.
- Computing and using "meeting points as possible destinations" is a very new research subject exploited in this thesis. We used a geometrical model to extract good places to join groups of people detected by the wheelchair and then use our model of user intention to consider those points as possible goals.
- We propose to use a "multi-modal interface" using the user's face position and speech recognition. The safety and usability of the interface is improved by integrating the shared local planner and destination inference to avoid collisions and reduce the necessary focused attention when the user is driving the wheelchair.

– The wheelchair's navigation function was improved by a adding a "social cost" that considers the cost of bothering people by interrupting their interactions or invading their personal space when the wheelchair navigates around.

1.5 Thesis outline

Chapter 2 - State of the Art: This chapter presents the state of the art on robotic wheelchairs. First of all, a review on different types of architectures is presented. Several user-wheelchair interfaces will be introduced giving special emphasis to those that can be improved by means of robotic assistants. An analysis of the different navigation approaches according to the degree of frailty of the user will be presented. Finally, we will examine how our proposed method compares to other similar approaches.

Chapter 3 - Tools for Navigation in Human Populated Environments: In this chapter, we present the control architecture of our robotic wheelchair. Including a description on how obstacle information is managed by the navigation system and our proposal to respect socially defined regions. We also present a method to compute good places to join a groups of people engaged in a conversation. Finally, a discussion about the applications and performance of the system is presented.

Chapter 4 - User Intention Aware Navigation: In the first part of this chapter we will present an introductory discussion about the importance of inferring the user intended destination when driving the robotic wheelchair. Then our probabilistic to infer the most probable destination model will be proposed. In the second part of this chapter we propose a reactive navigation algorithm that considers the input from the user and the sensors of the robot to compute the necessary velocity commands to move without collisions while remaining as close as possible to the orders given by the user.

Chapter 5 - Experiments: In this chapter we present the experiments that were done during this thesis. We describe the experimental wheelchair, the face-pose and vocal interfaces. Then the results of the experiments are discussed and commented.

Chapter 6 - Conclusions and perspectives: In this chapter we will present the conclusions and future work directions to pursuit for the destination inference, shared controller, meeting points computing and social cost-mapping presented in this thesis.

1.6 Publications

A. Escobedo, A. Spalanzani, and C. Laugier, *Using social cues to estimate possible destinations when driving a robotic wheelchair*, in IROS, 2014.

A. Escobedo, A. Spalanzani, and C. Laugier, *Multimodal control of a Robotic Wheelchair: Using contextual information for usability improvement*, in IROS, 2013.

J. Rios-Martinez, A. Escobedo, A. Spalanzani, and C. Laugier, *Intention driven human aware navigation for assisted mobility*, in Workshop on Assistance and Service robotics in a human environment, IROS 2012.

A. Escobedo, J. Rios-Martinez, A. Spalanzani, and C. Laugier, *Context-Based Face Control of a Robotic Wheelchair*, in Workshop on navigation and manipulation assistance for robotic wheelchairs, IROS 2012.

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

A. Escobedo, A. Spalanzani, and C. Laugier, *Experimental Setup for Human Aware Navigation*, in Control Architectures of Robots, 2012.

Chapter 2

State of the Art

This chapter presents the state of the art on robotic wheelchairs. First of all, a review on different types of architectures is presented. Several user-wheelchair interfaces will be introduced emphasising those that can be improved by means of robotic assistants. An analysis of the different navigation approaches according to the user's degree of frailty will be presented. Finally, we will examine how our proposed method compares to other similar approaches.

2.1 Introduction

Research in mobility assistance devices has earned increasing importance in recent years. Wheelchairs have been the most important mobility aid for people with motor disabilities for a long time and there exist a big interest in developing strategies to make those devices available for a larger group of users and enhance their performance to be easier and safer.

A robotic wheelchair is a type of mobile robot intended to be used by people with motor disabilities who are unable to drive a normal electric wheelchair or users who need some assistance when maneuvering in cluttered areas. Many robotic wheelchairs are differential robots consisting of two motorized wheels that share the same axis. In general, at least one free caster wheel is added to improve the stability of the robot. This type of robotic platform is *non-holonomic* which means that it can not drive instantaneously in every direction.

First works on intelligent wheelchairs appeared in the 80's and many of them have been proposed in all around the globe since then. Madarasz et al. (1986) from Arizona State University proposed to use an autonomous robot for the physically disabled.

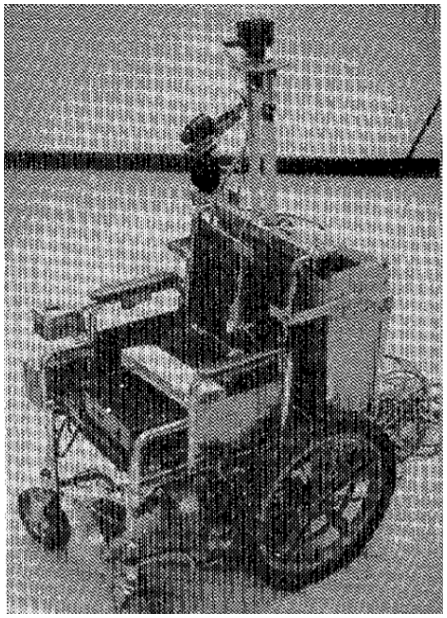


Figure 2.1: The autonomous wheelchair prototype from Arizona State University (1986) was the first project that try to use a mobile robot as a wheelchair to assist people with disabilities.

This robot was equipped with an on-board microcomputer a (128 px x 128 px) digital camera and a ultrasonic range finder mounted on a tilting base. The platform already contemplated -theoretically at least- a planning algorithm to give a list of actions to move between rooms and a visual servoing system that worked at 2Hz to drive the wheelchair in corridors. The limited perception and computing power as well as the absence of battery supply made largely constrained the real use of this system.

Robotic wheelchairs have been classified according to different factors as its mechanical design, the input method, the control software, the operating mode etc Simpson (2005).

- **Mechanical Design**: It relates to the fabrication of the robotic wheelchair. It can be made from an autonomous robot with a supplementary seat, a commercial electric wheelchair with new sensors and computing units, or a manual wheelchair adapted with sensors, a computer and some actuators.
- **Input method**: The interface with the user is a very important part to consider when designing the robotic wheelchair. Different interfaces have been proposed ranging from traditional input methods (joysticks, switches, touch screens, etc.) to more sophisticated devices as voice recognition, face pose control, brain-computer interfaces (BCI), electro-oculography etc.

– **Operating Mode**: Systems with different levels of autonomy have been proposed to assist users with different degrees of frailty. Those system range from full manual control when the user is completely in charge of driving, to fully autonomous robots where the user gives the desired destination and the wheelchair computes the necessary velocity commands to reach it while avoiding obstacles. Most of the proposed solutions rely on shared control architectures or semi-autonomous robots where the control is divided between the robot and the user. In every case the selected operating mode should be based on the user capabilities.

2.2 Mechanical Design Classification

Many researching groups have used technologies that were originally developed for mobile robots to create robotic wheelchairs. According to the way they are built, there are two different types of experimental robotic wheelchairs.

The first type is formed by a standard electric wheelchair connected to a computer and some sensors; e.g. The robotic wheelchair from INRIA-Rennes laboratory (Pasteau et al., 2013), the RobChair from University of Coimbra (Lopes et al., 2012), the third VAHM prototype (Grasse et al., 2010), Sharioto from Leuven University (Vanhooydonck et al., 2010), the SmartChair from University of Pennsylvania (Patel et al., 2002), Rolland from Bremen University (Lankenau et al., 1998) shown at figure 2.2.

The second type consists of a mobile robot base with an added seat. This type of platforms has the advantage to be excellent experimental test prototype allowing fast development and integration with new experimental control and navigation methods; e.g. our robotic wheelchair at INRIA Rhône-Alpes (Escobedo et al., 2013), the MIT autonomous wheelchair (Hemachandra et al., 2011), and the first VAHM prototype (Bourhis and Agostini, 1998) shown at figure 2.3.

2.3 Input Devices

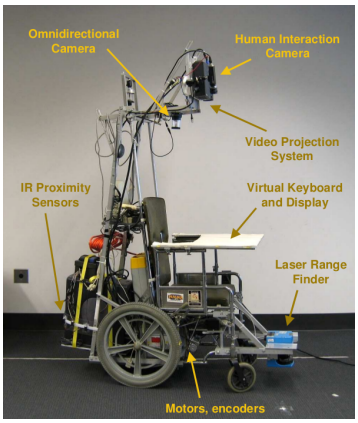
Input or access devices are those instruments used by the user to drive the wheelchair. Typical electric wheelchairs use a joystick as access device, however, there are many different dedicated interfaces to be used by patients with diverse illnesses or disability degree.

Even those users that can drive by means of a joystick could be assisted by a robotic wheelchair to perform some difficult or repetitive maneuvers as corridor following or traversing a door and almost every other access device is also suitable to be improved

(a) INRIA-Rennes wheelchair (France).

(b) VAHM Third generation, University of Metz (France).

(c) Sharioto, Leuven University (Belgium).



(d) SmartChair, University of Pennsylvania, (USA).

(e) ROLLAND, Bremen University (Germany).

Figure 2.2: Robotic wheelchairs built from standard electric wheelchairs with added sensors and computing units.

(a) INRIA Rhône-Alpes robotic wheelchair (France). (b) MIT robotic wheelchair (USA). (c) First generation VAHM, University of Metz (France).

Figure 2.3: Wheelchairs built from autonomous robots with added seats.

by some kind of robotic assistance. In this section we will present some of the most common access devices used by robotic wheelchairs in order to understand how different users could be assisted by the robotic device.

2.3.1 Continuous Input Devices

In this kind of input devices the control signal is continuously measured, interpreted and used to drive the vehicle. The user can make small adjustments to both the speed and direction of the wheelchair, however, he has to do it constantly (in the same way as the driver of a car) to properly control the movement of the device which is sometimes difficult and tiring. Some input devices of this type are the joystick, head control, finger control, touch pads, etc.

- **Joystick:** The joystick is the most common continuous input device. The normal joystick consists of a stick attached to a gimbal and is operated using the hands. Other type of joysticks can be operated by movements of the head, chin, or tongue. This device measures the angular displacement of the gimbal with respect to the repose position and maps it into a desired linear and angular velocities (Fig. 2.4).
- **Head Control:** The user pushes the left side of the head support to go left, the right side to go right and pushes back to go forward. In order to back up the user must activate a switch and then push the head support straight back. The user must activate the switch again to move forward. This can be an important

(a) Hand-operated joystick (b) A joystick operated with the chin.

Figure 2.4: The joystick is the most common access method or input interface with an electric wheelchair.

Figure 2.5: In a typical head control the user pushes the head support in the desired direction to control the wheelchair.

drawback if the user is in a situation where several back and forward movements are needed to get through a doorway or enter an elevator etc (Fig. 2.5).

- **Finger Control:** It consists of a small square box with a hole in the top of it. To drive the chair, the user places one finger through the hole on the top of the box and moves the finger in the direction they want the power wheelchair to move. This system is basically the same principle as a joystick but the user only moves a finger instead of the joystick (Fig. 2.6(a)).
- **Touch-Pad Control:** Allows the user to drive the power wheelchair with only one finger. Because touch pads are also proportional, the user can determine and control the speed of the wheelchair while moving simply by a small movement of the finger (Fig. 2.6(b)).

2.3.2 Discrete Input

This type of interfaces send a signal just at a discrete time, as a result, the interaction with the user is slower than with continuous input devices, and the number of

(a) Finger control. (b) Touchpad control.

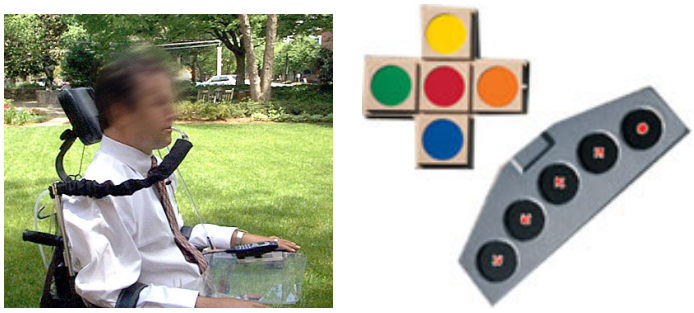
Figure 2.6: These types of input devices can be operated by users who can only move one finger.

possible commands is also significantly reduced. They are normally designed to be used by users with very severe disabilities.

- **Sip and puff:** Are designed for those users who aren't able to use any part of their body to operate a control device on their power wheelchair. Sip and puff systems are non-proportional drives and require quite a bit of practice by the user to get good at driving. Generally, the user will sip a specific number of times to indicate a direction, and puff to confirm the choice and activate the movement of the wheelchair. It is common for an auxiliary display to be used with sip-and-puff to provide feedback to the user (Fig. 2.7(a)).
- **Switches:** An array of switches can be used for the directional input of a wheelchair. The switches are usually mounted visibly on a board and activated by the user's hand. This digital system might be an option for users who have some control of a hand but can't maintain the constant control needed to operate a joystick. Switches are also indicated for individuals who have good control over an anatomic site not usually used to control a wheelchair. An individual with a disability might, for example, have better motor control over a foot rather than a hand. An array of large switches mounted to the footrest could then be used as a direction input (Fig. 2.7(b)).

2.3.3 User-Machine Interfaces Used by Robotic Wheelchairs

In addition to the typical input devices mentioned before, many robotic wheelchair research is based on the development and testing of new input methods with the purpose of helping a larger group of patients or making the driving experience easier or more



(a) Sip and puff. (b) Switches.

Figure 2.7: Different discrete access methods used in commercial electric wheelchairs.

intuitive.

Inaccurate or slow interfaces such as brain-computer, or electro-oculography based systems can be used thanks to the obstacle avoidance and intelligent assistance provided by robotic wheelchairs. This kind of interfaces could be the only form of communication for certain patients as those affected by tetraplegia or other sever motor disabilities therefore there is a huge interest in its development.

New interfaces can also be used to increase the comfort and usability of the wheelchair. Those new kind of human-machine interfaces can allow to interact with the wheelchair in a more natural and human-like way. This is the case of systems that use interfaces as voice control, gesture recognition, face pose recognition or eye trackers.

Brain Control Interfaces (BCI)

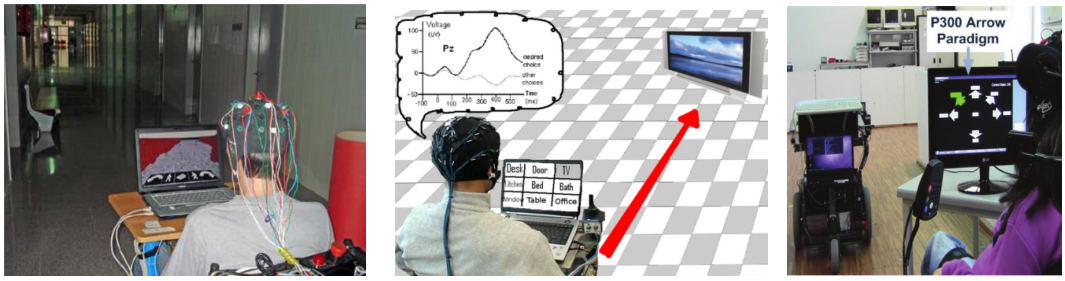
This is one alternative way to control robotic wheelchairs that have been proposed in the literature. Brain Control Interfaces are based on the decoding of the electrical brain activity, to perform the communication between the robot and the user and they are well adapted for paraplegic patients. The recording of brain activity is typically obtained using electroencephalographic systems.

Electroencephalography consists on placing electrodes on the scalp with a conductive gel or paste to measure the electrical activity of the brain. The number of electrodes depends on the application and they can be mounted on a cap for convenience of use. Then, the recorded signal is filtered and digitalized to be used by the computer.

There are two general ways of decoding the generated electrical brain signal:

- **Signal Classification:** It uses a classification process that analyses the features in order to find a matching command, which will be used to generate an action in response to the command.
- **Event Related Potential (ERP):** It generates an stimulus in order to elicit

specific response patterns in the human brain activity. This is the case of the P300 evoked potential that is one of the most used BCI signals. It is a natural and involuntary response of the brain to rare or infrequent stimuli. In this paradigm a random sequence of stimuli is presented to the user who is only interested in one of them. Around 300 milliseconds after the target is presented, a positive potential peak is recorded in the EEG signal.



(a) (Iturrate et al., 2009) (b) (Rebsamen et al., 2010) (c) Robchair, University of Coimbra (Portugal).

Figure 2.8: Brain-computer Interfaces used to drive robotic wheelchairs

Iturrate et al. (2009) describes a brain-actuated wheelchair that relies on a P300 based BCI and autonomous navigation. During operation, the subject looks to a screen that shows a real-time virtual reconstruction of the scenario, the user then focuses his attention on the spatial area to reach. A visual stimulation process elicits the neurological phenomenon and the EEG signal processing detects the target area. This target area represents a location that is given to the autonomous navigation system, which drives the wheelchair to the desired place while avoiding collisions with the obstacles detected by a laser scanner. A P300 BCI was also used in (Rebsamen et al., 2010). In this case the user selects a destination amongst a list of predefined locations that is shown to him on a screen. In the example of figure 2.8(b) the user selects the destination (TV) in the menu on the laptop monitor, items are flashed randomly and if the user focuses his attention on an item, the EEG signal will present a peak around 300 ms after the target has been flashed, which determines the destination. Then the wheelchair follows a guiding path to it.

The RobChair wheelchair from University of Coimbra (Lopes et al., 2012) uses also a P300 BCI to drive a robotic wheelchair. The stimulus is created by enlightening a marker displayed to the user on a screen (Fig. 2.8(c)). The user provides the sparse commands such as: stop, go forward, large turn left, small turn left, etc. And they fuse this data with the results of a vector field histogram local planner to perform obstacle

avoidance.

Gaze Direction and Face Pose Recognition

Systems that rely in the position of the face could ease the driving task given that it is a natural human behavior to look where we are going, however, the use of this type of interfaces is difficult due to the existence of involuntary movements and the reduced accuracy of the detection methods. Face direction interfaces normally use image based face trackers where some features are used to measure the direction of the face. Methods that detect the position of the face using 3D point-cloud cameras have been developed in recent years and some of them have reported good results (Murphy-Chutorian and Trivedi, 2009),(Fanelli et al., 2011).

Other methods try to detect the gaze direction by using electro-oculography or image based eye trackers. Electro-oculography detects changes in the potential between the front and back part of the eye to measure the movement and estimate the gazing point. Methods based on image processing track the position of the eye in a given video sequence and estimates the gaze direction given that they know the position of the camera in front of the user.

The WATSON wheelchair from the Nara Institute of Sciences and Technology in Japan (Matsumoto et al., 2001) used an stereo camera to estimate the position of the user's face and gaze. The user is supposed to drive the wheelchair by looking where he wants to go, and can start and stop by shaking his head. The system also estimates whether the user is concentrated on the operation based on the relationship between the head and gaze movement. The wheelchair is operated in full manual mode.

The Intelligent Wheelchair System project from the University of Osaka (Kuno et al., 2003) uses two cameras to control the navigation. The first one is used to perform a facial recognition of the user to determine the direction to follow, and the second one is designed to follow a target while avoiding obstacles detected with a set of proximity sensors. In (Hashimoto et al., 2009) it was presented a method for controlling an electric wheelchair using a hands-free manipulation interface based on the electro-oculography and electromyography biopotential signals. They detect gestures as jaw closure, wrinkled forehead, and look left and right to control the wheelchair without assistance from the robotic wheelchair.

In (Banerjee et al., 2010) a wheelchair controlled by a 2D face pose recognition system is presented. The user is completely in charge of driving the wheelchair by moving his face without any assistance from the robotic controller. In (Berjon et al., 2011) the direction of the movement is controlled by moving the face up, down, left and right to move forward, back, left and right. They do not propose any obstacle

avoidance provided by the robotic controller.

Voice Control

Voice control has long been desired as a control mechanism for wheelchairs but it is very difficult to implement within a standard power wheelchair because of the low bandwidth of the voice signal. Speech recognition is used in works such as (Simpson and Levine, 2002), (Murai et al., 2009), (Ruzaij and Poonguzhali, 2012). In (Simpson and Levine, 2002) the NavChair platform used vocal commands as (go forward, go back, turn left and right), the performance of the system is improved by using obstacle avoidance based on the vector field histogram method, the authors reported good results when comparing against not assisted use of the vocal command. In (Murai et al., 2009) the wheelchair has two possible driving modes: follow a wall and get into an elevator. The user switches between modes using the voice control and drives using commands such as "Turn right 90 degree", "go forward" etc. The method presented in (Ruzaij and Poonguzhali, 2012) employs speech recognition to control the movement of the wheelchair in different directions (back, forward, left, right) without assistance.

In (Berjon et al., 2011) the Google Speech Recognition Service and Microsoft SAPI are used in combination with a 2D face tracking to control a mobile robot. The speech recognition is used to switch between modes. In the first mode the user directs the robot using "execute backwards, forward, left and right" commands, while in the second mode the direction of the movement is controlled by moving the face up, down, left and right similar as a joystick, again no further assistance is provided by the robotic controller. Voice recognition is employed in (Kollar et al., 2010) to select the desired navigation task or destination using high level commands as (follow me, go to the kitchen, etc) and then the robot executes the task autonomously.

2.4 Operating Modes

Systems with different levels of autonomy have been proposed to help users with different degrees of frailty. Those robotic wheelchairs can be classified in four different categories as:

- Manually controlled wheelchairs.
- Shared controlled wheelchairs with collision avoidance.
- Semi-autonomous wheelchairs.
- Robotic wheelchairs with fully autonomous navigation.

2.4.1 Manual Control

This is the simplest way of controlling a mobile robot (from a computing complexity perspective). The user controls directly the robot's translational and rotational velocities, such as driving a powered wheelchair with a joystick. The robot task is thus only to execute the translational and rotational velocities ordered by the user. In manual control, all the responsibility of platform safety, obstacle avoidance, and path planning to the destination is let in charge of the human user.

2.4.2 Shared Control

Even if there is no formal definition to distinguish between shared control and semi-autonomous control; we will assume the following difference them. In shared control the user gives commands continuously so the assisting device should be able to modify them at a similar frequency to avoid obstacles. In semi-autonomous mode the user just gives sparse directional commands whenever he wants to change his trajectory while the robot performs the full navigation task to move in the selected direction until the user gives another directional command.

Shared control is presented in situations in which the assisting device combines the control input coming from the robot and the user in order to accomplish a given task. Those methods do not normally require a pre-built map or any specific alterations to the environment, however, they require more thinking effort and focus from the user.

A shared control system has to consider the human in the loop problem and given the complexity of the human being; it is difficult to be modeled properly. In Fig. 2.4.2, we present the shared control paradigm where both the robotic wheelchair and the user collaborate to accomplish the navigation task. The robot has a predefined model of both the world and the user and the amount of correcting actions taken by the robot will depend on those models. The user contributes to the controlling task by perceiving the results of the actions taken by the shared controller through his senses and applying a regulatory input signal according to his intentions.

The ROLLAND wheelchair from the University of Bremen (Lankenau et al., 1998) uses a neural network based control and a camera. The wheelchair is equipped with a ring of sonar sensors and internal sensors to measure the speed and steering angle. This wheelchair reads the user commands from the joystick and checks them for safety. If there are no close obstacles in that direction the commands are passed to the actuators. If an obstacle comes close, the driving wizard reduces the speed of the wheelchair proportionally to the distance to the closest obstacle.

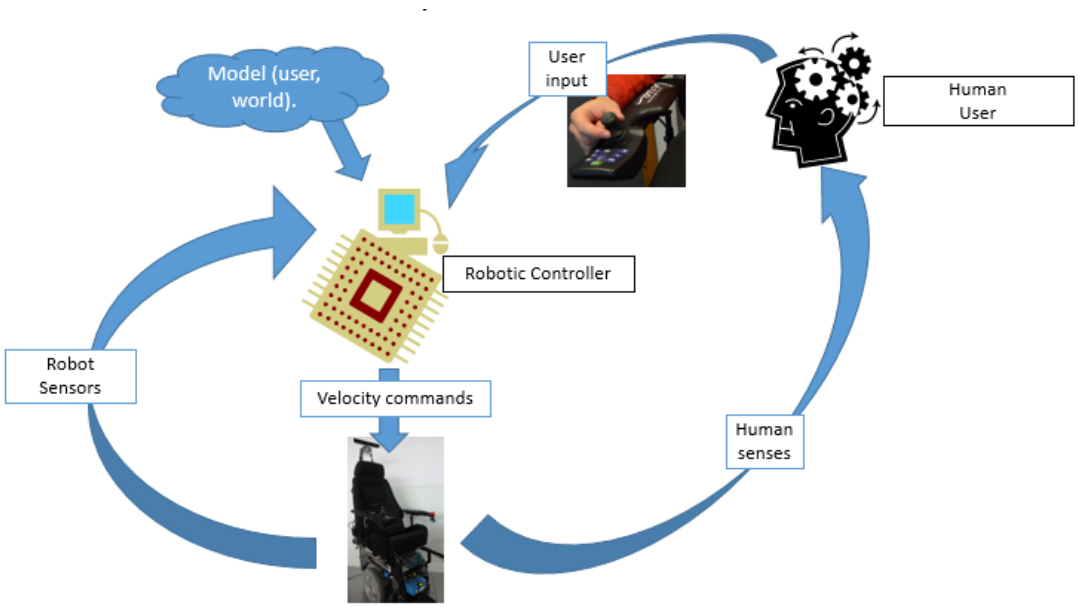


Figure 2.9: Shared control paradigm: Both the robotic wheelchair and the user collaborate to accomplish the navigation task. The robot has a predefined model of both the world and the user that will produce the desired level of autonomy to the user. The user contributes to the controlling task by perceiving the results of the actions taken by the shared controller through his senses and applying a regulatory input signal according to his intentions.

The SmartChair project from the University of Pennsylvania (Patel et al., 2002) uses an omni directional camera and a laser range finder to control the navigation of the wheelchair. The wheelchair is able to move through a doorway and perform obstacle avoidance while the user is in charge of the planning and velocity control most of the time.

The NavChair project developed by the University of Michigan (Simpson, 2005) employs three operating modes: general obstacle avoidance, door passage, and automatic wall following. It is equipped with a joystick and a belt of sonars. They use the vector field histogram method to look for the best direction of travel according to the obstacles and the direction pointed by the joystick.

The wheelchair developed at INRIA-Rennes (France) uses a single camera and visual servoing to follow a corridor without any previous knowledge of the environment. The non-holonomic constraints of the wheelchair are taken into account by the servo controller (Pasteau et al., 2013).

2.4.3 Semi-Autonomous Control

Many works consider semi-autonomous control solutions as those where the user takes care of the high level path planning and the robot of the velocity control. The user just points out to the place he wants the robot to go. This concept refers to the fact that the robot is autonomous on a given short path, requiring then a new input from the user regarding the next goal direction. In these situations, an intermediate controller has to plan a path from the current robot location to the nearby goal and compute the velocity commands.

The RobChair project (Lopes et al., 2010), (Lopes et al., 2012) presents a semi-autonomous approach to drive the wheelchair using discrete human-machine interfaces. The controller is able to share the information provided by the local motion planning level with the commands issued sparsely by the user. The local planner is based on the Vector Field Histogram algorithm. Every time a new obstacle is detected, the local planner is activated. Obstacle detection is carried out using a laser range finder located at the front of the robot. They present a Fuzzy Shared-Controller that takes the user command and the velocities computed by the Vector Field Histogram algorithm to get the final linear and angular velocities sent to the wheelchair.

The Sharioto wheelchair from Leuven University (Vanacker et al., 2007), uses a semi-autonomous system to help the subject to drive with a noninvasive brain interface. The subject's steering intentions are estimated from electroencephalogram signals and passed through to the control system before being sent to the wheelchair motors. To filter out the commands given by the user, the wheelchair detects corridor crossings and assigns a probability value to each possible command (LEFT,RIGHT,BACK and FORWARD) according to the direction of the opening. Obstacle avoidance is performed by the wheelchair between crossings.

The biba robot from ETH Zurich, (Perrin et al., 2010) presents a semi-autonomous navigation strategy designed for low throughput interfaces. A mobile robot (e.g. intelligent wheelchair) proposes the most probable action, as analyzed from the environment, to a human user who can either accept or reject the proposition. In the case of refusal, the robot will propose another action, until both entities agree on what needs to be done. In an unknown environment, the robotic system first extracts features so as to recognize places of interest where a human-robot interaction should take place (e.g. crossings). Based on the local topology, relevant actions are then proposed, the user providing answers by means of a button or a brain-computer interface (BCI).

2.4.4 Autonomous Control

Robotic wheelchairs with fully autonomous navigation are designed so that the user gives a final destination and supervises as the wheelchair is in charge of the complete navigation task. This kind of system requires to have a map of the environment. Furthermore, the robot has to localize itself within the map given the sensory readings. The goals and robot positions are then described in a global coordinate system. Receiving the order to go to a specific location, the robot first plans its path through the environment and then moves.

In the late 90's Bourhis and Pino (1996) proposed an autonomous navigation system able to plan a path in a map and execute the full trajectory to arrive to the destination. A big drawback was that the wheelchair just followed blindly the computed path and it did not perform obstacle avoidance. The user could just stop the execution of the path if he detected any trouble while the wheelchair was executing the trajectory which could be annoying or even dangerous.

Modern autonomous navigation methods where also used in (Wang and Chen, 2011) to solve the obstacle avoidance problem while following the path they used the lane curvature method (Simmons, 1998). The authors claimed to have a good response to avoid obstacles quickly, however the navigation method had some troubles when turning or passing through narrow doorways. The user interacts with the wheelchair by using a touch-screen and selecting his desired destination in the map of the environment that is displayed to him. Again, once the execution of the trajectory is started; it is no possible for the user to make adjustments in the velocity control.

Fully autonomous methods that have been proposed up to the moment does not consider the possibility of an eventual adjustment in the path made by the user. This lack of controllability can be frustrating for the user which is why we think that semi-autonomous or shared controlled navigation systems can be easier to accept in real situations.

2.5 User Intention Estimation

The estimation of the user objective destination is a key point in many shared control and semi-autonomous navigation systems because it allows the automatic controller to adjust its actions to the desire of its user.

Some methods to perform an implicit estimation of the user intention from joystick commands have been proposed in (Demeester et al., 2006), (Demeester et al., 2009).

They model the user intention as a set of possible trajectories. A probability distribution is maintained over the set of trajectories and the most likely one is selected within a Bayesian framework.

In (Taha et al., 2008) a learned Partially Observable Markov Decision Process (POMDP) is used to estimate the user intended destination in a topological map of the environment. Places of interest are selected as those locations in the environment where the user spends comparatively most of his time. The user drives the wheelchair from one spatial location to another while the robotic device avoids obstacles in the middle. This method uses corridor crossings to define sub-goal points that are used to update the POMDP, given a determined sequence of visited sub-goals the system will be more certain about the user's final goal.

In (Perrin et al., 2010) a local reasoning system is used to get an initial probability distribution over the actions the robot could execute (in crossings), so as to engage a dialog with its user. Given the recognized topology, this probability distribution relates each possible action with the type of crossing detected to filter-out unlikely actions.

2.6 Human Aware Navigation

Human aware navigation is receiving an increasing attention in robotics community, this area of research appears once that robots navigate in human environments and safety solutions are not enough; now the main concern is related to produce solutions which also have to be understandable and acceptable by human beings.

A proposal of human aware navigation was presented in (Sisbot et al., 2007), where a motion planner takes explicitly into account its human partners. The authors introduced the criterion of visibility, which is based on the idea that the comfort increases when the robot is in the field of view of a person. Other work, (Tranberg Hansen et al., 2009), introduced an adaptive system which detects whether a person seeks to interact with the robot based on the person's pose and position. That system was presented as a basis for human aware navigation. Their results showed that the system was able to navigate based in past interaction experiences and to adapt to different behaviors.
In (Chung and Huang, 2010) was proposed a Spatial Behavior Cognition Model (SBCM) to describe the spatial effects existing between human-human and human-environment. SBCM was used to learn and predict behaviors of pedestrians in a particular environment and to help a service robot to take navigation decisions. Technique in (Sehestedt et al., 2010) proposed an on-line method to learn generally occurring motion patterns in an office environment with a mobile robot. Navigation is realized by using these

patterns, in form of sampled hidden Markov model (HMM), along with a probabilistic road map based path planning algorithm. Socially acceptable motion is achieved by minimizing social distractions, such as going through someone else working space.

The work presented in (Lam et al., 2011) proposed rules that a single robot should obey in order to achieve not only a safe but also a least disturbance motion in a human populated environment. Rules define sensitive zones for both humans and robots, depending either on their security regions or the psychological feeling of humans.

Personal space, o-space and their relation to comfort were addressed in (Rios-Martinez, 2013), where a risk based navigation was extended to include risk due to discomfort. Human movement is supposed to be known by learning of typical trajectories in a particular environment.

2.7 Approaching Humans

Carton et al. (2012) investigated what abilities robots will need to successfully retrieve missing information from humans. Socially-aware navigation is employed to request help from human passing by. An approach based on Bezier curves is implemented as a nonlinear optimization problem with the objective to find a velocity profile for the Bezier path under constraints enhancing social acceptance. Experiments were realized where the robot approaches a static human at different velocities and angles. In (Yamaoka et al., 2009) and (Yamaoka et al., 2010) different formations were used to control the position of the robot when it presents information to humans. They established a model for information-presenting robots to properly adjust their position. The model consisted in four constraints to establish the interaction space: proximity to a listener, proximity to an object, listeners field of view, and presenters field of view. The model was implemented for a humanoid robot with a motion-capturing system.

In (Althaus et al., 2004) the authors proposed a method for a robot to join a group of people engaged in conversation. The results of the implementation and the experiments conducted with their platform show a human-like behavior as judged by humans. Robot just wants to preserve the formation of the group and doesn't know explicitly where the o-space is located.

Shi et al. (2011) studied natural human interaction at the moment of initiating conversation in a shopkeeper scenario where a salesperson meets a customer. Then they use the observed spatial formation and participation state to model the behavior of initiating a conversation between a robot and a human.

2.8 Discussion and Thesis Positioning

If a truly user-machine integrated system is to be developed, the type of cooperation between user and machine must always consider the possible input coming from the user to navigate, while the wheelchair avoids dangerous obstacles on the ground. To achieve this level of user-machine integration the machine should predict the user intention with only minimal corrective input from the user.

The fundamental component of this kind of systems is intention recognition. Many platforms have already been devised in the past to help people in their daily maneuvering tasks; Bremen Autonomous Wheelchair, Sharioto, RobChair, VAHM, Wheelesley, and Navchair. The developed architectures consist of different algorithms that each realize a specific assistance behavior, such as "avoid an obstacle" , "follow a corridor" or "pass through a door". In many cases, different levels of shared control are defined, ranging from full operator control to full computer control. The operational mode is normally selected by the user manually.

The probabilistic user intention model presented in this thesis is similar to the one presented in (Demeester et al., 2006) where the user intention was modeled as a set of possible trajectories. However, in this thesis the user intention is modeled at a higher level as a set of possible destinations. This difference leads to a considerable reduction in the computation complexity of the Bayesian Network.

The work presented in (Taha et al., 2008) is interesting for us because they proposed a method to infer the intended destination using Partially Observable Markov Decision Process, however, the POMDP has to be computed off-line and the navigation of the wheelchair is performed only following the corridor from one local goal to the next and no obstacle avoidance is performed in between. We do believe, that in order to be practical for the user, the wheelchair has to provide a method to navigate without colliding with obstacles in the environment.

While many of the other presented systems use a vector field histogram method to perform reactive navigation (Simpson, 2005), (Lopes et al., 2010), we propose a modified version of the dynamic window approach which is new and it has the advantage of providing reliable and fast performance and allows to consider the intention of the user at different levels, as the goal, the computed path and the current input command. The navigation of our robotic wheelchair is improved by adding the respect of personal and interaction spaces.

The use of natural interfaces can improve the usability of the robotic wheelchair. Many of the works presented in the literature have tried to use a face pose recognition

system to drive the wheelchair (Banerjee et al., 2010), (Berjon et al., 2011),(Wei and Hu, 2011) but only a few have tried to use it with obstacle avoidance (Simpson and Levine, 2002), (Kuno et al., 2003). We established a semi-autonomous navigation mode where the position of the face is used to detect the user's desired destination while the robot is in charge of the path planning and velocity control. The usability of the face pose interface was extended by using speech recognition. This could be considered as a drawback, because of the use of two different interfaces can be confusing for the user, however, using the voice and face gestures is a natural way of interacting which indeed makes the system easier to use.

Even though, we didn't use a BCI or an electro-oculography interface as those presented in (Tanaka et al., 2005), (Rebsamen et al., 2010), (Lopes et al., 2012) or (Hashimoto et al., 2009) our approach was tested also using an intentionally limited interface (keyboard) to give commands similar to those obtained by typical BCI interfaces as "FORWARD", "BACK", "LEFT" and "RIGHT". The results obtained using this method are encouraging to use the system with this kind of interface in future works.

The socially-aware navigation method presented here, differs to the one presented in (Sisbot et al., 2007) because our assumption of discomfort is, in some way, the opposite of the visibility criterion: the field of view (in front of the human) shows the point of interest of a person then, if the robot enters to it, the activity of the person will be interrupted decreasing the comfort function.

Chapter 3

Tools for Navigation in Human Populated Environments

In this chapter we present the control architecture of the wheelchair navigation system and the related necessary tools to navigate in human populated environments. An introduction to the different functional modules used by the wheelchair as the A path planner to compute the path given a destination (Hart et al., 1968), the social filter (Rios-Martinez, 2013) to avoid disrupting social formations and to compute good places for joining groups of people in the environment. Those tools were integrated and used to perform the autonomous navigation of the robotic wheelchair during this thesis. Finally, the simulation results and a discussion on the performance of the system is presented.*

3.1 Introduction

Mobile robots and specially service robots have to interact with people regularly, therefore, they must follow behavior rules similar to those governing human-human interaction. It is even more important in the case of a robotic wheelchair, because its user may be held responsible for the inappropriate behavior of the robot.

People generally perceive robots as human-like, even when the robots are non-anthropomorphic (Kirby et al., 2009). When such robots does not behave according to what is socially expected, breakdowns in human-robot interaction occur (Mutlu and Forlizzi, 2008). Those situations can be annoying for people around the robot, or even worst, they can produce accidents that could be prevented.

Traditional robot navigation algorithms treat all the detected objects as the same

type of obstacles to be avoided, however, in this thesis we claim that robots should treat nearby people in an special way to be socially accepted. Given that the field of human aware navigation is very extensive, we will address two particular cases of this subject.

- **Respecting personal and interaction spaces:** The first case responds to the idea that people can be upset if they are approached at a distance that is considered to be too close. The discomfort experienced by the person is related to the distance at which he is approached. This idea was formalized as the concept of *personal space*, proposed by Hall (Hall, 1969), which characterizes the space around a human being in terms of comfort to social activity. Similarly, sociologist have studied the space respected by a group of people interacting (such as having a conversation) have been described using the concept of *interaction space*.
- **Approaching humans in a socially acceptable way:** The second case is a new research field that addresses the issue of determining those places where someone should place himself in order to attract the attention and become part of an interacting group -known as *meeting points*.

The system proposed in this thesis (Fig. 3.1) aims to integrate methods that have successfully been used in mobile robotics (white and grey blocks) to enhance the performance of the robotic wheelchair while adding some functional modules (orange blocks) to overcome the previously presented limitations. Now we will explain briefly the function of each of the modules that will be discussed during this chapter.

- **Global and Local Occupancy Grids:** The obstacles perceived by the robotic wheelchair are represented using 2D occupancy grids. An "occupancy grid" is a representation that uses a multidimensional (typically 2D or 3D) division of the space into cells, where each cell stores a probabilistic estimate of its state (Elfes, 1989). One possible classification of the state is as occupied, free or not observed. In this work we use the practical open source software implementation documented in (Marder-Eppstein et al., 2010) where the robot maintains two different occupancy grids. The global occupancy grid is used only when the robot has a pre-built map of the full (global) environment where the robot is supposed to operate while the local occupancy grid is centered on the robot's reference frame and is updated every time that new information is retrieved by the sensors.
- **Path Planner:** Throughout this work we will be referring to the path planner as the functional module that computes a free path connecting the robot's starting location with the desired destination given a map of the environment. Its search

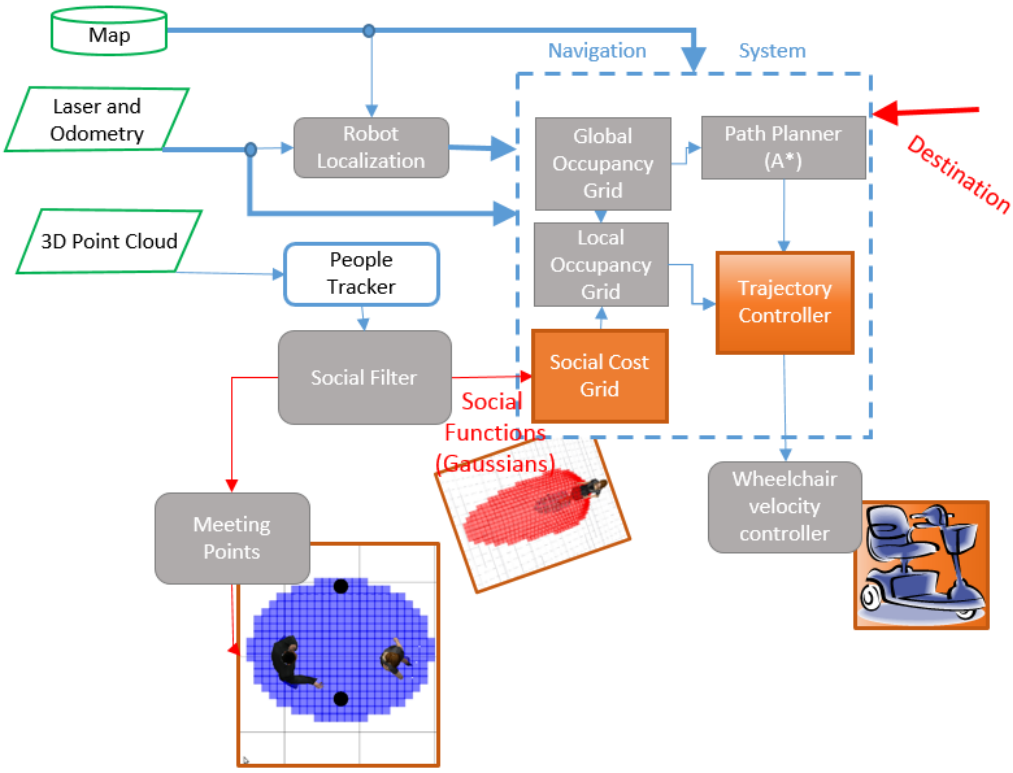


Figure 3.1: The navigation system of the robotic wheelchair. In this section we will explain the functional modules developed as part of this thesis (orange). The components of the system in gray will be explained as they are important for the overall functioning of the system but they were taken from the open source community or they are previous works developed at our research team.

area is discrete and represented by a the global occupancy grid. This module uses the A* search algorithm (Hart et al., 1968) and the software was taken from the open source ROS [1] community and integrated with the wheelchair's navigation system.

– **Trajectory Controller:** Also known as Local Planner, contrary to path planner, where the problem is to find a complete path from the initial position to the goal in a known environment, many real world applications for autonomous navigation in dynamic environment are based on reactive collision avoidance techniques also know as trajectory control or local path planning. This functional module is in charge to compute at each time-step the next appropriate angular and linear velocities (v, ω) to navigate towards the goal while avoiding obstacles. This module

1. ROS: Robot Operating System, http://wiki.ros.org/navfn

will be presented in Sec. 4.3.2.

- **Social Filter:** Integrates constraints inspired by social conventions in order to evaluate the risk of disturbance and take it into account when making the autonomous navigation planning (Rios-Martinez et al., 2011).
- **Social Cost Grid:** This functional module was developed as part of the current thesis, it receives information from the social filter and adequate this data to be used in a similar way as the local and global occupancy grids to be used by the path planner and the trajectory controller.
- **Meeting Points Estimator:** It uses a geometrical model to compute good places to approach groups of people in social interaction (meeting points). Those meeting points are a special kind of possible destinations which are not static but depend on complex social relationships governing human interactions. Whenever an interaction is detected by the social filter, the related meeting points are computed and considered by the destination inference as possible options (Rios-Martinez, 2013).
- **Robot Localization:** This component is used to determine the robot's position with respect to a given fixed frame. Our wheelchair uses the Advance Montecarlo Localization Method that matches data from the laser range finder with a given map to estimate the position of the robot in the map (Thrun et al., 2005).
- **Velocity Controller:** It receives the velocity commands given by the trajectory controller in the format of pairs (v, ω) and computes the necessary speed of the left and right wheels to achieve the required velocity. This controller is based on the model of a differential drive vehicle.

In the following sections we will present in detail how the system deals with personal and interaction spaces and how they are represented as occupancy grids to be used by standard planning algorithms. Then we will present the method to compute adequate meeting points that will be used in the next chapter by the user intentions algorithm. Finally, a presentation about the other required modules integrated in the wheelchair platform as well as simulation results will be presented.

3.2 Social Cost Mapping

Many planning algorithms use occupancy grids as its basic data structure to represent the state of the world. Practical implementations of such algorithms typically used in robot navigation systems (in this work A* and dynamic window) consider occupancy grids with three possible values for every cell: {free, occupied and unknown}

(Marder-Eppstein et al., 2010).

Our method to take the personal and interaction spaces into account when the robot moves is based on the *Social Filter* presented by Rios-Martinez (2013). It implements constraints inspired by social conventions to evaluate the disturbance caused by the robot when it moves near to humans. This measurement can then be considered by the robot's planning algorithm.

In this section, we will first introduce the occupancy cost-map method used to manage obstacles. Then we will present the mathematical models used to represent the personal and interaction spaces.

3.2.1 Introduction to Occupancy Cost Mapping

Sensor processing to detect obstacles around the robot is a very important part of any navigation system. Given that sensor data are not perfect, the obstacle detection can be erroneous and report non-existent obstacles or missing objects that should be detected.

Many current sensor processing methods rely on 2D occupancy cost grids to infer obstacles from geometric information provided by range sensors (lasers, 3D cameras, etc). Laser range-finders are largely used in robotics due to its high accuracy and it is the type of sensors used by the robotic wheelchair used during this thesis.

Definition 1 *The occupancy grid is a space representation that employs a multidimensional (typically 2D or 3D) array of space into cells, where each cell stores a probabilistic estimate of its state (Hart et al., 1968).*

The occupancy grid representation we use is updated whenever new information is sensed. If the sensor detects an object, the corresponding cell in the map are marked as occupied while the cells between the sensor and the obstacle are considered to be free. Unexplored regions, e.g. those behind the obstacle, are marked as unknown as shown in Fig. 3.2.

Occupancy information is used to create safe plans for the navigation system. Therefore each cell has an associated cost value ($OccCos$) that is used for scoring purposes by the path planner and trajectory controllers. The data structure used to keep the values of the occupancy grid is an array of 8 bits integers which allows to have occupancy cost values between 0 and 255. We will now explain each of the possible cost values associated to each cell type.

- **Unknown Space:** (255) Means that the robot does not have information about the state of this cell because it is not in its field of view and is outside the map.

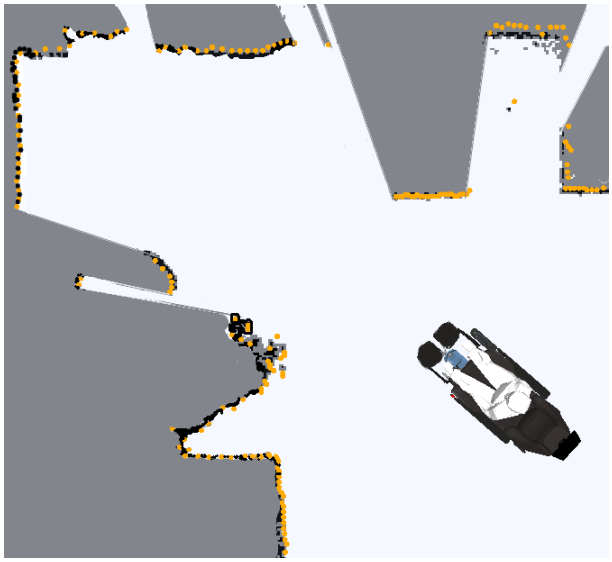
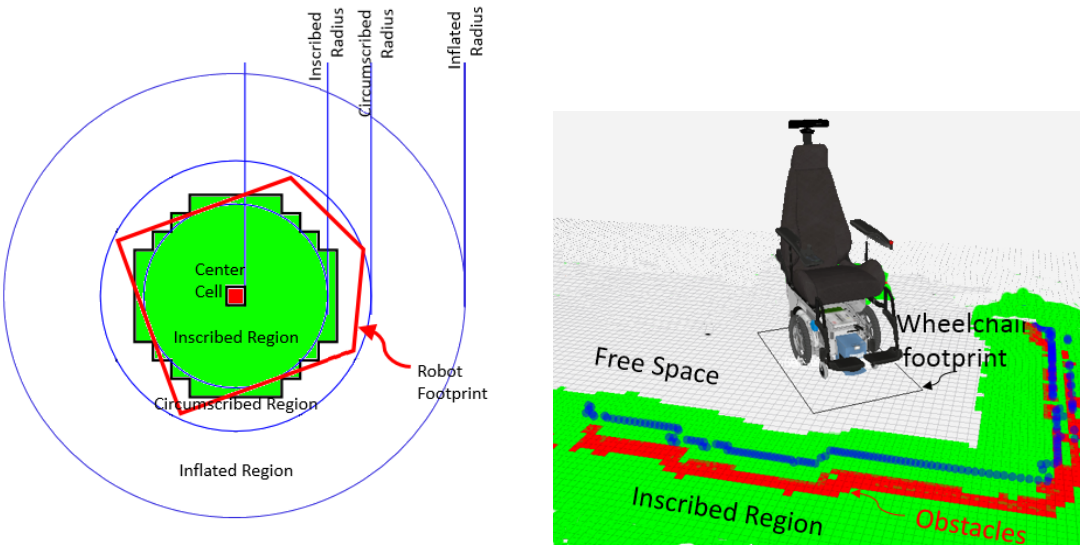


Figure 3.2: 2D occupancy grid generated by the robotic wheelchair, the yellow dots are laser impacts. The corresponding cells are marked as an obstacle (black squares). The gray cells are non-observed.

This value can be used by the path planning for example to avoid generating paths along unexplored regions.

- **Occupied Space:** (254) Means that there is an actual (workspace) obstacle in this cell. If the robot's center were in that cell, the robot would obviously be in collision.
- **Inscribed Space:** (253) Means that a cell is less than the robot's inscribed radius (Fig. 3.3) away from an occupied cell. So the robot is certainly in collision with some obstacle if the robot center is in a cell that is at or above the inscribed cost.
- **Circumscribed Space:** (252 to 1) If the robot center lies in a cell further than the robot's inscribed radius but closer than the circumscribed radius, then the planner should depend on the orientation of the robot to decide whether it collides with an obstacle or not. Cells marked with this value might not be a real obstacle, but some user-preference, to avoid passing too close to a given region.
- **Free Space:** (0) It means that there is nothing that should keep the robot from going across this cell.

All the cells in the grid are assigned a state between free space and inscribed space depending on their distance from an occupied cell. This process is called "inflation" and it is performed in two dimensions to propagate costs from occupied cells to a user-



(a) (b)

Figure 3.3: Inflation is the process of propagating cost values out from occupied cells (red) that decrease with distance. Cells in the inscribed region are assigned a uniformly high cost (inscribed cost), after which an exponential decay function (Eq. 3.2) is applied that will cause the cost to drop off out to the inflation radius used for the cost map.

specified inflation radius. Cells that are less than one inscribed radius of the robot away from an obstacle are assigned a uniformly high cost, after which an exponential decay function (Eq. 3.1) is applied that will cause the cost to drop off out to the user defined inflation radius of the cost map.

$$Cc = (Lc - 1) * \exp^{-Cs*(d-r)} \tag{3.1}$$

Where, (Cc) means circumscribed cost, (Lc) is the lethal cost value. (d) is the distance from the cell to the closest obstacle and (r) is the inscribed radius of the robot. (C_s) Is the cost scale parameter that allows to control how fast the function decays. (r_{inf}) is the inflated radius defined by the user and (r_{insc}) is the robot's inscribed radius as shown in figure 3.3. To summarize, a cell in the occupancy cost grid

can have any of the following cost values.

$$OccCost = \begin{cases} 255 & \text{if Unknown Space} \\ 254 & \text{if Occupied Space} \\ 253 & \text{if Inscribed Space} \\ (Lc-1) * \exp^{-Cs*(d-r)} & \text{if } r_{insc} < d < r_{inf} \\ 0 & \text{if } d > r_{inf} \end{cases} \tag{3.2}$$

3.2.2 Personal Space

Definition 2 *The Personal Space is an area that individual humans actively maintain around themselves and is related to the space in which individuals tend to direct the majority of their actions (Hayduk, 1978). This selective use of space derives from the way human organisms are built. Limbs, for example, can be used effectively only within a highly circumscribed space-bubble stretching mainly in front and to the sides of a person's body.*

The geometrical representation selected for our system was presented in (Rios-Martinez, 2013). The model (Fig. 3.4) is based on the measures of personal and intimate zone defined in (Hall, 1969) and consists of two Gaussian functions as expressed in G_f and G_b (Eq. 3.5) both of them centered in the position of the person. This model is similar tom that presented by (Laga and Amaoka, 2009), but it uses the body orientation as main feature instead of face orientation.

The model reflects the fact that the disturbance for someone is bigger in positions closer to the person than in the borders and it is stronger in front of him than in the back. The values of the Gaussian functions represent the risk of disturbance associated to each point in the space around the pedestrian.

The discomfort caused around a person due to the invasion of his personal space is modeled by evaluating a two-dimensional Gaussian function of covariance matrix Σ and centered in the local reference frame of the person, for each point P around the person:

$$G(P) = e^{-\frac{1}{2}P^t\Sigma^{-1}P} \tag{3.3}$$

Where, P is a point in $\mathbb{R}^2$ and Σ is a diagonal covariance matrix defined as:

$$\Sigma = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix} \tag{3.4}$$

The appropriate shape of the personal space is obtained by selecting the same values for σ_x in both G_f and G_b but different values for σ_y, being the one of G_f the double of the value for G_b. The points C and P are defined in the person's reference frame, where the x axis is in the direction of the person's sightline and the y direction respects the right hand rule. The traversability cost of a given point around the person $P = (x_{local}, y_{local})$ will be expressed as:

$$psCost(P) = \begin{cases} G_f(P) & \text{if } x_{local} \geq 0 \\ G_b(P) & \text{if } x_{local} < 0 \end{cases} \tag{3.5}$$

Where x_{local} is the x component of the P point with respect to the person body. Fig. 3.4 shows an example of personal space for a human, the approximated shape is shown in the left where the Gaussian values are projected in the plane of the ground, observe in the right the high risk of disturbance at the center and front. The disturbance has a maximum value of one in the position of the person and a minimum value of zero in his public zone.

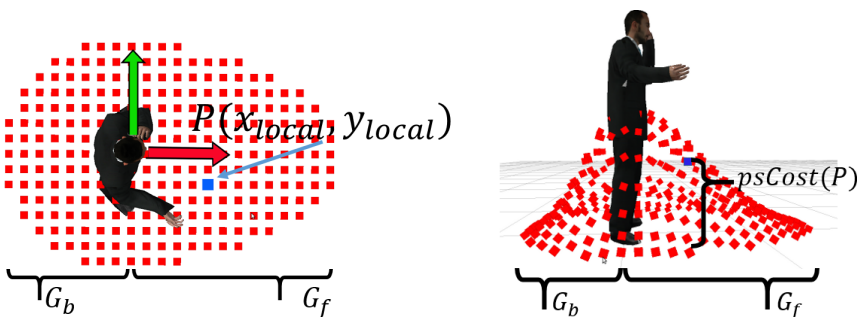


Figure 3.4: Personal space calculated by Social Filter Module. Left: top view, right: side view. The height of the Gaussian means Risk of disturbance then maximum disturbance is located at human position.

3.2.3 Interaction Space

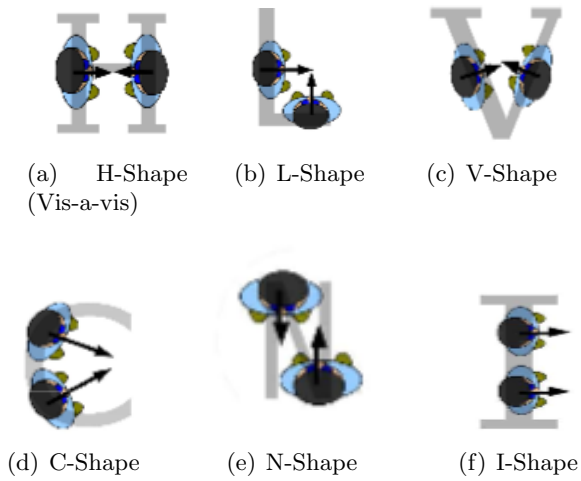
Definition 3 *The interaction space is the joint area reserved by a group of two or more people sharing the same focus of cognitive and visual attention during a conversation. Only the members of the group are allowed to access this space and other people tend to respect it (Kendon, 2010).*

The interaction space was called o-space by Kendon (1976) because in many cases

its shape is similar to that letter. It varies depending on body size, posture, position and orientation of participants during the joint activity.

Definition 4 *A F-formation is a term used to designate the system of spatial orientation, arrangement and postural behavior that people create and maintain in order to keep their interaction space. F-formations are distinctive of people who come conjointly to accomplish a joint activity; those formations are typically associated with the occurrence of small conversational gatherings. (Ciolek and Kendon, 1980).*

For the specific case of two people interactions, some particular F-formations have been identified as being peculiarly frequent; Ciolek and Kendon (1980) suggested that almost 98% of two-people stationary encounters take one of those forms. Those formations are called according to the shape described by the participant's bodies as: H-formation, L-formation, V-formation, C-formation, N-formation, I-formation (Fig. 3.5).



(a) H-Shape (Vis-a-vis) (b) L-Shape (c) V-Shape

(d) C-Shape (e) N-Shape (f) I-Shape

Figure 3.5: Most typical F-formation types.

Four of the mentioned formations were implemented in the "social filter" [1]: Vis-a-vis, L-Shape, C-Shape and V-Shape. This geometrical model considers the participant's body position and orientation, the model is different for each type of formation.

Given the positions of pedestrians $H_1 = (x_1, y_1)$ and $H_2 = (x_2, y_2)$ in the global reference frame and their respective orientations ϕ_1 and ϕ_2 around the normal to that

1. **Social Filter is a functional module used in this thesis that integrates constraints inspired by social conventions in order to evaluate the risk of disturbance and take it into account when making the autonomous navigation planning (Rios-Martinez et al., 2011).** model (Rios-Martinez, 2013)

plane 3.6 the interaction space is represented by a two-dimensional Gaussian function with covariance matrix S and centered in $C(x_c, y_c)$, then for each point $P = (x, y)$ around the center we have:

$$intCost(P) = e^{-\frac{1}{2}(P-C)^t S^{-1}(P-C)} \tag{3.6}$$

Where S is a diagonal covariance matrix defined as:

$$S = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix} \tag{3.7}$$

Rios presented a selection of σ_x and σ_y values to modulate the shape of the interaction space according to the type of detected formation. In this work we use those values, but we will not explain how they were derived.

Formation	σ_x	σ_y
Vis-a-vis	$D_H/3$	$2DH/3$
L-Shape	$D_H/2$	D_i
C-Shape	$D_H/4$	$D_i/3$
V-Shape	$D_H/3$	$2D_i$

D_H is the euclidean distance between the two persons, D_i is the orthogonal distance from the line joining H_1 and H_2 and the intersection of the sight lines of both persons. [1]

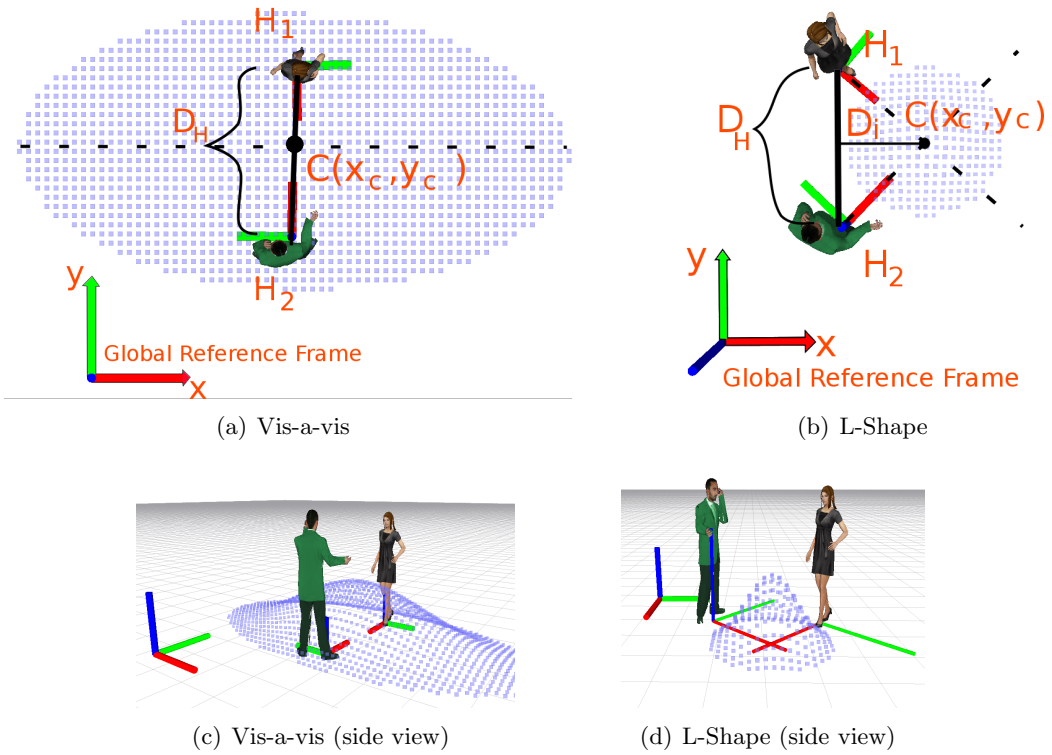
Groups of More Than Two People When more than two people are in a conversation they exhibit a formation with a circular shape with center coinciding with that of the inner space of the group. This form is more obvious when the number of participants grows. The corresponding geometric model of this interaction space is known as o-space because of its circular shape. When a group of more than two people is detected in the scene, the interaction space's center is located at a point C. This point is calculated based on the positions of the group members using:

$$C = \frac{1}{n}\sum_{1}^{n} P_i \tag{3.8}$$

Where, n is the number of people in the group and P_i is the position of the i_{th} person.

As in the other cases a two-dimensional Gaussian is used to estimate the risk of disturbance around the group. The Gaussian is centered in C and the standard deviation for both dimensions is $(\sigma_x = s_d, \sigma_y = s_d)$. The value s_d is defined as the minimum

1. Those terms will be explained in detail in section 3.3.



(a) Vis-a-vis

(b) L-Shape

(c) Vis-a-vis (side view)

(d) L-Shape (side view)

Figure 3.6: Example of the elements in the geometrical o-space model for a vis-a-vis and a L-shape formation.

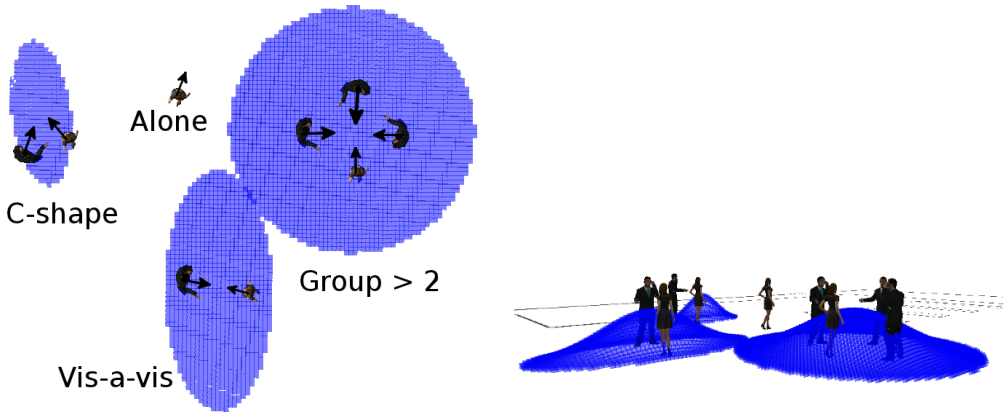


Figure 3.7: Interaction space computed by the Social Filter functional module for different formations. The maximum risk of disturbance is located at the o-space center, the disturbance is represented by the height of the Gaussian functions (blue regions).

distance from P^i to C. Therefore, for each point $P = (x, y)$ around the center we have a social cost called *intCost* defined as:

$$intCost(P) = e^{-\frac{1}{2}(P-C)^t S^{-1}(P-C)} \tag{3.9}$$

where, S is a diagonal covariance matrix defined in Eq. 3.7.

An example situation with a detected group of four people is shown at Fig. 3.7 ($Group > 2$). Also a C-shape and a Vis-a-vis formations were detected in the scene.

3.2.4 Representing Social Spaces as Cost Grids

The social cost data computed previously is projected on the 2D occupancy grid used by the path planner and the local planner in the following way.

Two different thresholds are applied to project the values of both personal space and interaction space as shown in Fig. 3.8. Cells in the occupancy grid with personal or interaction space values higher than the threshold are marked as occupied zones so the planners will avoid passing through those regions, while regions with lower values are marked as free space.

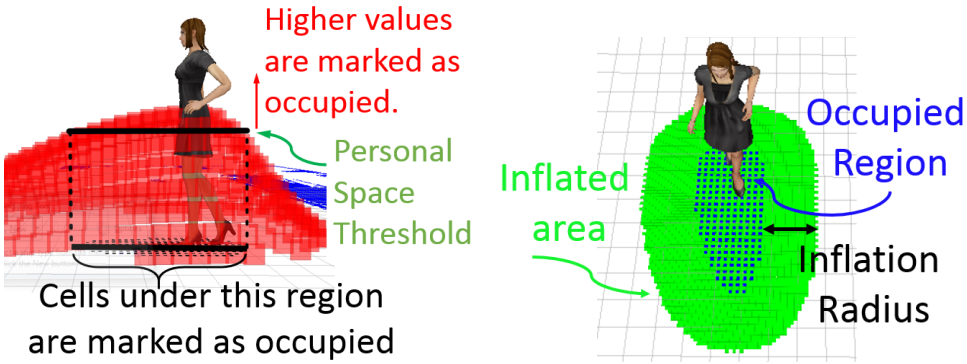
$$socialCost(P) = \begin{cases} Occupied & \text{if} \quad psCost(P) > T_{PS} \ \vee \ intCost(P) > T_{IS} \\ Free & \text{otherwise} \end{cases} \tag{3.10}$$

Where T_{PS} is the threshold applied to the personal space cost and T_{IS} is the threshold applied to the interaction space cost. The value of those thresholds was set by experimentation in the developed software. The final grid is created by computing the maximum value between the occupancy cost and the social cost. [1]

$$gridCost(P) = \max[socialCost(P), occCost(P)] \tag{3.11}$$

Occupied regions computed by the social filter are then combined with the occupancy grid created with the robot's laser range finder and the map of the environment. In order to consider the size of the wheelchair, regions marked as occupied are inflated by the size of an *inflation radius* proportional to the size of the robotic wheelchair (Fig. 3.8(b)). In order to avoid collisions or invading the personal space, the center of the robot should never get inside this inflated area.

1. In the implemented software we used the numerical value of 254 as occupied while 0 represents free space.



(a) (b)

Figure 3.8: Threshold applied to the Gaussian functions delivered by the social filter to created the occupancy-cost grid. (b) The inflated area (green) considers the dimensions of the wheelchair. In order to guarantee a path free of collisions the center of the wheelchair should never be inside this area.

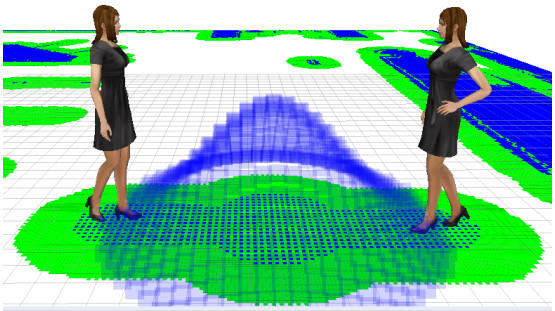
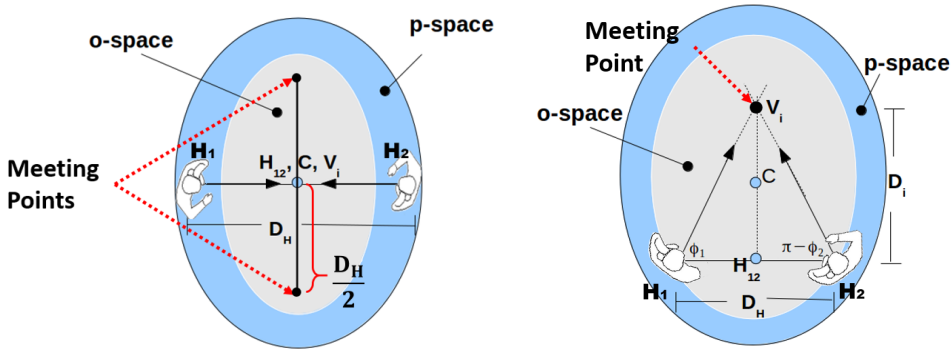


Figure 3.9: Projection of both p-space and o-space on the occupancy grid used by the planning algorithm. The Gaussian in the middle represents the value of the interaction space, which is projected on the occupancy grid as occupied space (blue area) and then inflated to consider the size of the robot (green area).

3.3 Meeting Points Computing

Meeting points are defined as those places where the robotic wheelchair should be placed to attract the attention and/or become part of an interacting group. According to the theory presented by (Rios-Martinez, 2013) those points are located in an area, roughly coincident with the o-space of an interaction where the robot can share the space in an equitable way with the other humans already present.

Fig. 3.10 shows different types of social formations and the positions where the meeting points should be located. Fig. 3.10(a) represents a vis-a-vis or frontal formation



(a) For a vis-vis formation.

(b) For a v-shape formation.

Figure 3.10: Geometrical placement of meeting points in different social formations. The location of those meeting points depends on the focus of attention defined by the orientation of the bodies ϕ.

where two humans H_1 and H_2 are talking face to face. In this case two meeting points are placed on the line perpendicular to the H_{12} line that joins H_1 and H_2. the points are located at a distance $D_H/2$ from the center of the o-space. D_H is the distance between the two persons.

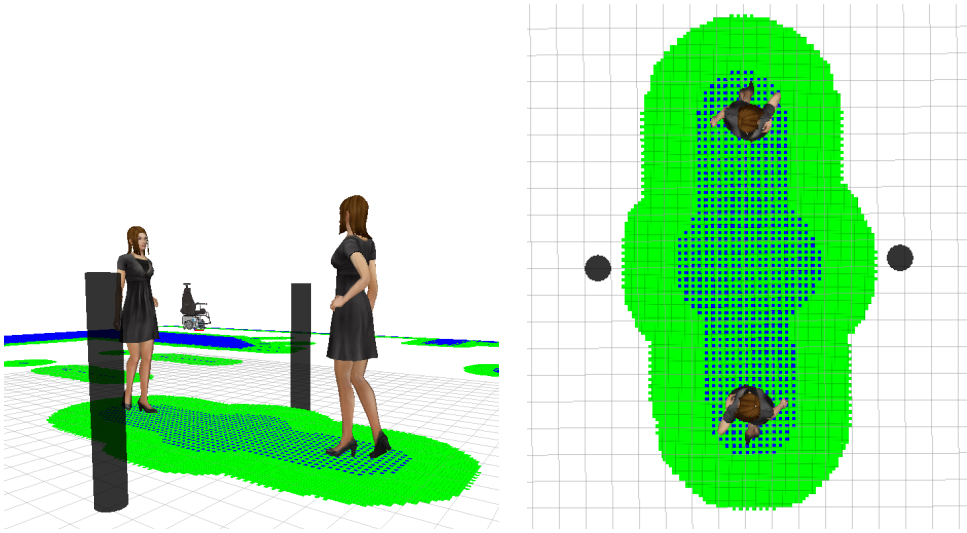
In the case of a v-shape formation Fig. 3.10(b) a meeting point is placed on the line formed by V_i and H_{12}. V_i is the intersection of the two visual axis of both persons in the interaction. The computed meeting points are used by the inference algorithm as possible destinations as it will be described later.

3.4 Navigation System

The wheelchair navigation system takes data from sensors, odometry, a navigation goal and the input from the user and outputs velocity commands that are sent to the mobile base. The low-level architecture of this system, consists of many components that must work together. The major components of the navigation system are presented next.

3.4.1 Trajectory Control

In dynamic environments it is not possible to have a perfect estimation of the future world's state and consequently the results of path planning get poor in the distant future. Therefore, the navigation system needs to include a functional module



(a) Lateral view. (b) Top view.

Figure 3.11: The resulting meeting points for a vis-vis formation (black markers). If the robot reaches that position, a group of three will be formed.

that reacts to changes in the environment. Those systems are usually called reactive approaches.

The problem addressed by reactive approaches is the computation of only the next command velocity of the robot (v, ω), where v is the velocity's linear component and ω is the angular part. The approach is based on the definition of a function to specify a "cost" for the possible controls or movements in terms of clearance with the surrounding obstacles and direction of the desired goal.

At each time step, the algorithm looks for the minimum cost control to be executed. To make the computation fast enough and allow the robot to be reactive to the changes in the environment, only the local information during a short period of time is considered [1]. The trajectory controller used by our robotic wheelchair will be further described in Sec. 4.3.

3.4.2 A* Path Planning

The path planning problem consists in finding the minimum cost path that connects two points given a map of the environment. A simple example is shown in Fig. 3.12: The starting point is called P_0, the final one is P_N and obstacles are represented by

1. This is the reason why it is also called local planner

black squares. The search area is discrete and represented by a grid in this example. However, different discretization alternatives can be used.

When computing the path, each square on the grid is related to an item of a list with status: free or occupied. The path is formed by those cells that allows to move from the starting point to the final one without hitting obstacles in the middle.

Once the path is found, the moving entity moves from the center of one square to the center of the next one until the target is reached. These center points are normally called *nodes*.

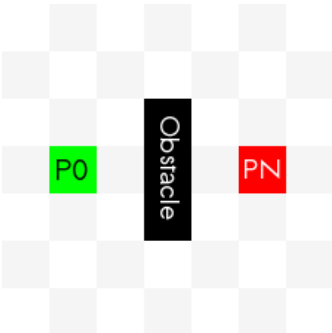


Figure 3.12: Path planning general problem description. The green cell P_0 is the starting point, the red one P_N is the desired ending point. Black cells are the obstacles while white/gray cells are free space.

The autonomous navigation system of the robotic wheelchair presented in this thesis relies on the A* path planning algorithm. Therefore an introduction to this algorithm will be presented in this section.

The A* method solves the path planning problem in an optimum way. It was first presented in (Hart et al., 1968) and is the most widely-known form of best-first search algorithms (Russell and Norvig, 2010). It has been successfully implemented as part of many autonomous navigation systems (Marder-Eppstein et al., 2010), (Konolige et al., 2011).

This planner works well for fully observable problems where the states of the world (the map) is well known, discrete and static. This is a very unlikely situation in real world conditions, that is why this technique needs to be combined with other methods (i.e. the dynamic methods listed in section 4.3) to work in dynamic environments. The method has three main subroutines:

- Search initialization (Algorithm 1).
- Main Loop (Algorithm 2).

– Trace the path

The method uses the following cost function f to score the cells in the map:

$$f(P_i) = g(P_i) + h(P_i) \tag{3.12}$$

where:

– P_i : Is the cell that needs to be scored.

– g : Is the cost to move from the starting point P_0 to a given cell P_i on the grid following the path generated to arrive there. The g cost is computed along a specific path to a given cell and is the sum of the g value of its parent cell plus the cost of moving from the parent to the current cell.

– h : Is the value of the used *heuristic* to estimate the cost of moving from the given cell P_i to the final destination, point P_N. The reason why it is called like that is because we really do not know the actual distance until we find the path. The h value can be estimated in a variety of ways as the direct distance between two points or the Manhattan distance (which are some of the more used heuristics). The closer our estimate h is to the actual remaining distance, the faster the algorithm will converge to the goal. If this distance is overestimated, however, there is no guarantee to find the shortest path.

Algorithm 1 A* Search Initialization

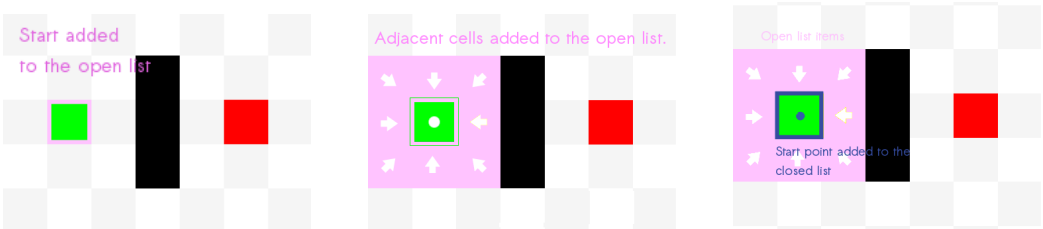
```
OpenList[0] ← P_0
for P_i ∈ AdjacentNodes(P_0) do
    if P_i is Free then
        P_i.parent = P_0
        P_i.g = 0
        P_i.f = f(P_i)
        OpenList[i] ← P_i
    end if
end for
ClosedList[0] ← OpenList[0]
MainLoop()
```

Definition 5 *The open list contains cells that could fall along the path, but maybe not, and they need to be checked out.*

Definition 6 *The closed list contains cells that will be certainly part of the path.*

The A* searching begins at the starting point P_0, then P_0 is added to the "*open list*" Fig. 3.13(a). The cells adjacent to the starting point (ignoring those with obstacles)

are also added to the open list. For each of these cells, the point P_0 is saved as the "*parent*" in the list (represented by white arrows in Fig. 3.13(b)). This is important when we want to trace back the resulting path (explained later). Drop the starting square P_0 from the open list and add it to the "*closed list*" Fig. 3.13(c).



(a) The green cell in the center is the starting point it is surrounded by a pink border to represent that it is already in the closed list.

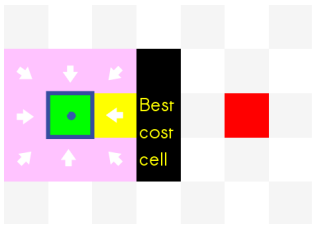
(b) All of the adjacent cells (pink) are now in the open list of cells to be checked. Each cell has a white arrow that points back to its parent node, which in this case is the starting cell.

(c) Starting point is added to the closed list (blue border)

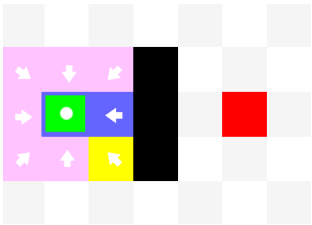
Figure 3.13: A* algorithm: Initializing the search.

- From the remaining cells in the open list select the one with the lowest total cost f to continue the search. The selected cell will be the "parent" of the new cells added to the open list Fig. 3.14(a).
- Drop the new parent from the open list and add it to the closed list Fig. 3.14(b).
- Check all of the adjacent squares. Ignoring those that are on the closed list or occupied by obstacles, add squares to the open list if they are not on the open list already.
- If an adjacent cell is already on the open list, check to see if the new parent gives a better path score g Eq. 3.12 to that cell. In other words, check to see if the g score for that cell is lower if we use the current cell to get there. If the g cost of the new path is lower, change the parent of the adjacent cell. If not, don't do anything.
- If a change was done, recalculate both the f and g scores of the modified cell.
- The process is repeated until the target square is added to the closed list Fig. 3.14(g), in which case the path has been found, or it fails to find the target square and the open list is empty. In this case, there is no path.

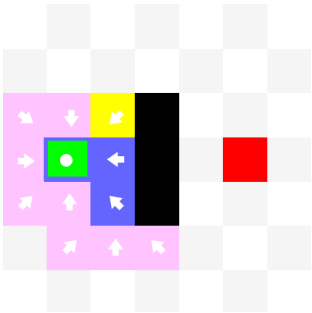
To finally determine the path, just start at the end node and go backwards moving from one square to its parent, following the arrows. This will eventually arrive to the starting square. Moving from the starting square to the destination is simply a matter



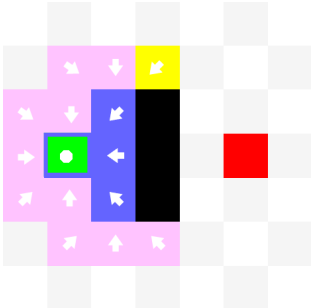
(a) The node with the best cost f (yellow) is selected to be the parent of the next nodes of the open list.



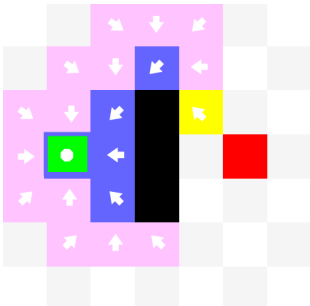
(b) Add the cell to the closed list (blue) and move to the next best scored node.



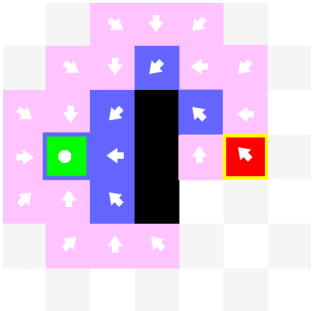
(c) The neighbors of the node with the best score (yellow) are added to the open list (pink)



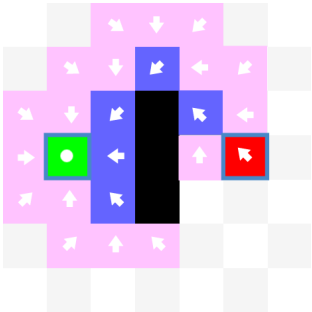
(d) The search continues following always the node with the next best score.



(e) The search continues.



(f) Even if the goal is already in the open list, the algorithm will continue the search.



(g) The target square is added to the closed list (Finish).

Figure 3.14: A* algorithm (Main loop example). The process finishes when the target square (red) is added to the closed list (blue boundary) (g), in which case the path has been found, or it fails to find the target square and the open list is empty. In this case, there is no path.

Algorithm 2 A* Main Loop

```
while OpenList is not empty do
    for P ∈ OpenList do
        P_best = P node with the lowest f value in OpenList
        ClosedList ← P_best
        Remove P_best from OpenList
        for Padj ∈ AdjacentNodes(P_best) do
            if (Padj ∉ ClosedList) ∨ (Padj.status = Free) then
                OpenList ← Padj
            end if
            if g(Padj) < Padj.g then
                Padj.parent = P_best,   Padj.g = g(Padj),   Padj.f = f(Padj)
            end if
        end for
        if P_best = Goal then TracePath()
        end if
    end for
    Error, the path could not be found
end while
```

of moving from the center of each square (the node) to the center of the next square on the path, until reaching the target.

3.4.3 Wheelchair's Self Localization

This system is used to answer the so called *localization problem* that consists on determining the robot's position with respect to a given fixed frame or with respect to known locations in the environment in order to navigate effectively and achieve destinations.

The most common and basic method for performing localization is through dead-reckoning that integrates the velocity history of the robot over time to determine the change in position from the starting location. Unfortunately, pure dead-reckoning methods are prone to errors that grow without bound over time, so some additional method is necessary to periodically correct the robot position.

Our wheelchair uses the Advance Montecarlo Localization Method (Thrun et al., 2001) that matches data from the laser range finder with a given map to estimate the position of the robot. The additional localization information is combined with dead-reckoning to probabilistically update the robot position.

The Advanced Monte Carlo Localization (AMCL) is a specialization of the Monte

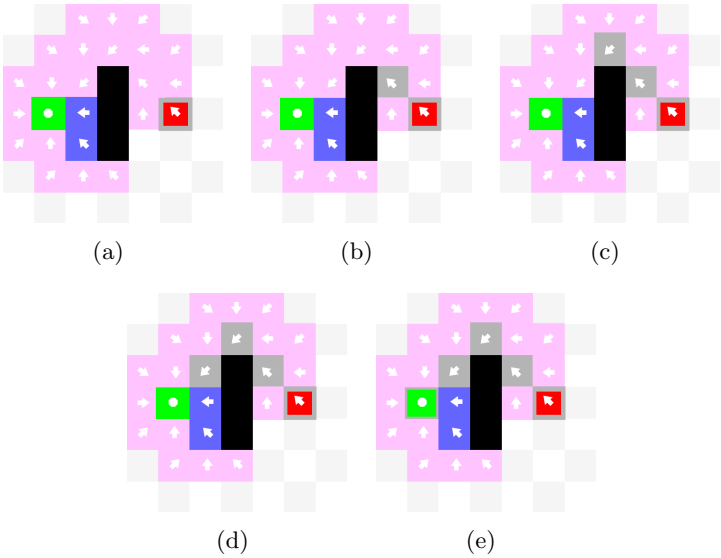


Figure 3.15: A* algorithm (trace the path). To determine the path, just start at the end node and go backwards moving from one square to its parent, following the arrows.

Carlo Localization algorithm that was introduced by Thrun (Thrun et al., 2001). This algorithm uses a particle filter to track the pose of a robot against a known map. given a map of the environment. It estimates the position and orientation of the robot as it moves and senses the environment. A particle filter is used to represent the distribution of likely states, with each particle representing a hypothesis of where the robot is located.

The algorithm typically starts with a uniform random distribution of particles over the configuration space, meaning the robot has no information about where it is and assumes it is equally likely to be at any point in space. Whenever the robot moves, it shifts the particles to predict its new state after the movement. Whenever the robot senses something, the particles are re-sampled based on recursive Bayesian estimation, according to how well the actual sensed data correlate with the predicted state. Ultimately, the particles should converge towards the actual pose of the robot. For further information please refer to (Thrun et al., 2001) and (Thrun et al., 2005).

3.4.4 Velocity Controller

For the experiments we conducted during this thesis, a differential-drive mobile robot was used (see section 5.1.1 for details). Therefore we will focus our overview on this type of robotic platform.

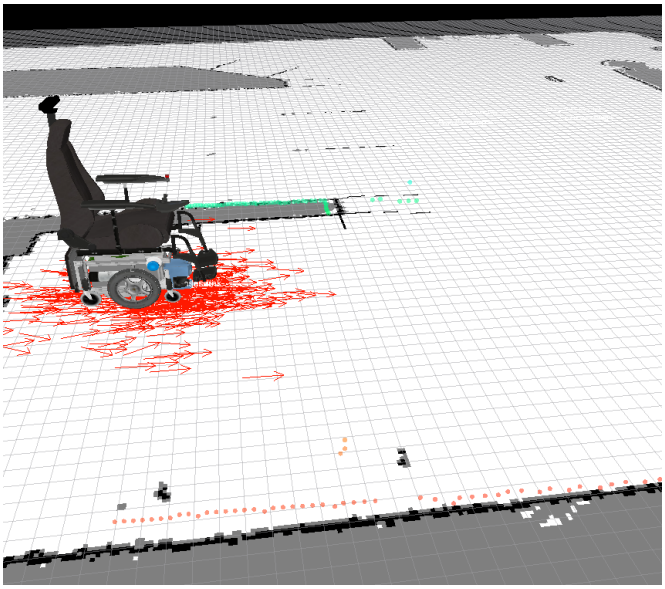


Figure 3.16: Advance Montecarlo Localization. Given a map of the environment, the position of the robot is modeled as a set of particles (red arrows) representing each of the possible robot states. The state with the highest likelihood is taken as the current position of the robot (where the wheelchair is actually located).

Definition 7 *A* differential-drive robot *consists of two motorized wheels that share the same axis. The robot's referential center lies on the middle point between the two wheels Fig. 3.17. Such a robotic platform has some constraint concerning the motion, as it can not drive instantaneously in every direction (non-holonomic motion: it can not move laterally).*

The velocity of any mobile robot can be expressed by a combination of its *linear velocity* v and *angular velocity* ω which combination is commonly known as a *twist*.

Given a differential-drive mobile robot as depicted in Fig 3.17, the wheel speeds (v_R and v_L) are related to the linear (v) and rotational (ω) velocities as described in equation 3.13.

$$v = \frac{r(v_R + v_L)}{2}, \tag{3.13a}$$

$$\omega = \frac{r(v_R - v_L)}{2b}, \tag{3.13b}$$

where r is the wheel radius and $2b$ the distance between the wheels.

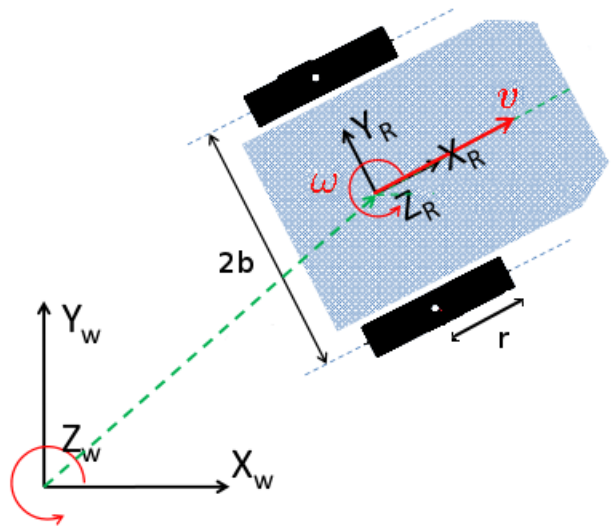


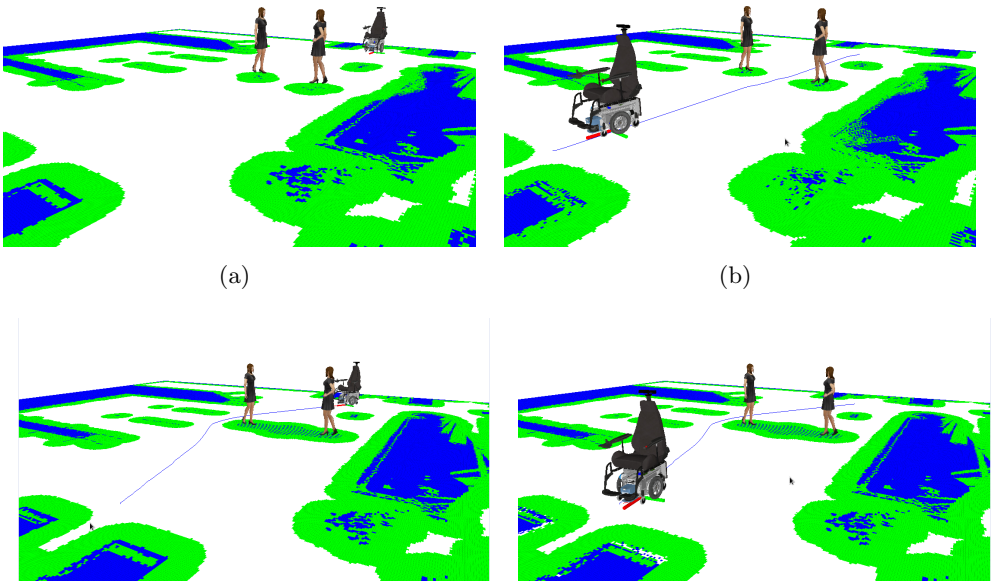
Figure 3.17: Differential-drive model showing the important geometrical characteristics of the robot, the reference axes and the different robot velocities. X_R , Y_R and Z_R corresponds to the robot referential system; v, ω are the robot translational and rotational velocities. $2b$ is the distance between the two wheels and r the radius of each wheel.

3.5 Simulation Results

The navigation system was tested in a simulated environment based on the INRIA's hall environment described in section 5.1.2. The map of the environment was previously generated from laser data and position and orientation of people in the scene is provided by the simulator. In the example scenario there are two persons and our simulated robotic wheelchair. The wheelchair moves using previously described navigation system using the developed social grid.

In Fig. 3.18 we present a comparison between the behavior of the wheelchair including the social grid and without it. The detected occupied regions or social spaces are marked in blue while the resulting inflated areas are shown in green. In the scene, there are two persons in the middle of the map in a interaction (facing each other).

The first sequence (Figs.3.18(a) and 3.18(b)) shows the resulting path computed by the wheelchair's path planner. The path passes in between the persons because they are considered as any other obstacle and the available free space between them is big enough for the wheelchair to pass. However, the same path planner can produce socially acceptable paths when including the social grid plug-in developed as part of



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

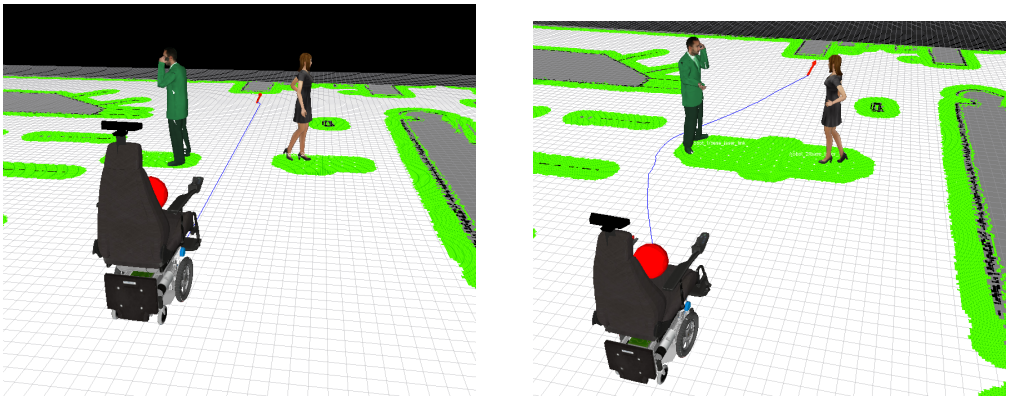
Figure 3.18: Comparison between the standard 2D occupancy grid provided in ROS (a), (b) and the one improved with social spaces (c), (d).

this thesis. This is shown in the second sequence (Figs. 3.18(c) and 3.18(d)) where the wheelchair is shown following a path that respects the interaction space between the persons.

In Fig. 3.19 we present two cases using the social grid. In the first example (Fig. 3.19(a)) the persons are looking front to front so the interaction zone between them is marked as occupied and the computed path avoids the invasion of the interaction space as described before. In the second case the simulated persons are located at the same distance but they are looking back to back which frees the space in between so the robot now computes an straight path because there is no social interaction to be avoided.

3.6 Discussion

Autonomous navigation techniques for robots moving among humans must be aware of social aspects of human interactions. A common expectation when humans and robots share the same physical spaces, is that they must follow the same social conventions when managing the space around. For example, robots must respect proximity constraints but also respect people interacting -as humans do. The work presented in



(a) When no interaction is detected the space between the two persons is free.

(b) When the persons are in a vis a vis formation, the interaction space between them is respected.

Figure 3.19: Path planning using the social occupancy grid. The robot is asked to go to the destination marked with a red arrow, it then plan a path avoiding the occluded areas (green). The A* path planner used is an open-source implementation provided as part of the ROS navigation stack (Open-Source-Robotics-Foundation, 2014).

this chapter addressed that issue in two general directions; avoid disrupting interactions and joining groups in a socially acceptable way.

The first case was based on the minimization of the estimated discomfort generated by paths followed by the robot, so the robot avoids disrupting conversations. Although the method used in this thesis is helpful to prevent possible embarrassing and even dangerous situations; this model of social spaces is static so it will be important to integrate methods that consider the movements of the persons when detecting a possible interaction, one idea for future implementations should be to use dynamically adapted representations as the one presented in (Papadakis et al., 2013).

The second case consisted in computing where the robot should place itself when approaching humans. The geometrical model presented here, gave us some first insights in the understanding of people's reaction when the robotic wheelchair approaches them, however it could be further extended to consider the dynamic evolution of the group once the robot becomes a part of the group. The estimation of meeting points is an important part of our system because it is used to infer the desired destination of the wheelchair user in section 4.2.4.

The social interactions were successfully implemented as a cost-grid that can be used with the popular open source ROS navigation software which makes it easier to be tested by other people interested in reproducing the experiments of this thesis. The

modular architecture of the wheelchair navigation system allowed us to use the path planner provided in this navigation software without modifications.

Although we have just presented how the wheelchair's autonomous navigation system was improved by adding the respect of social conventions it has not been presented how it considers the input and intention of the user while driving. This problem will be addressed in the following section.

3. TOOLS FOR NAVIGATION IN HUMAN POPULATED ENVIRONMENTS

Chapter 4

User Intention Aware Navigation

This chapter we will present and discuss the central part of this thesis. In the first part of this chapter -section 4.1- we will present an introductory discussion about the importance of inferring the user intended destination when driving the robotic wheelchair. In section 4.2 a probabilistic inference model is proposed and an autonomous navigation system is used to drive the robot towards the inferred destination. Some simulation examples are provided and the results are discussed at the end of the section. In section 4.3 we propose a reactive navigation algorithm that considers the input from the user and the sensors of the robot to compute the necessary velocity commands to move without collisions. Some simulation results are then presented and finally, the advantages and limitations of the method are explained.

4.1 Introduction

People with motor disabilities often have difficulties using joysticks and other standard input devices. The assistance provided by a robotic wheelchair can clearly improve the user experience when performing complicated driving tasks. In order to have an appropriate user-robot interaction it is necessary that each of them have some knowledge about the intentions of the other.

Intention recognition has been distinguished as one of the grand challenges facing intelligent human-robot interaction (Burke et al., 2004). This problem arises in multi-agent systems when two or more entities cooperate to achieve a certain task as the problem of driving a robotic wheelchair.

Humans usually perform complex intention recognition tasks to ease the way they interact with each other. For example, when humans are having a conversation they

try to predict how it will evolve in the future and which can be the reaction of the other person. Another example is people walking in groups who try to coordinate their movements by predicting the intention of each other. Intention recognition is a way to avoid or diminish the need for reliable and extensive communication in situations where coordination and cooperation is necessary (Tahboub, 2006).

In this work we propose to recognize the intention of the user in order to reduce the number of necessary commands to drive a robotic wheelchair and improve its response when the input given by user is ambiguous or inaccurate. In this way, the wheelchair can be in charge of some part of the navigation task and reduce the effort applied by the user.

To improve the user experience and diminish frustration, the robot should not only navigate securely among obstacles but also take into account the user intention at different levels as the final destination, the preferred path, the speed etc. At each level, the wheelchair should favor the most "reasonable" action according to the inferred intention.

In this chapter, we present two different approaches to enhance the navigation of the wheelchair and reduce the effort invested by the user. We will first present our model to infer the user desired destination by considering the direction of his commands and his habits. After inferring the destination the robot moves autonomously towards it. We will then describe a shared local controller that computes appropriate motor commands in combination with human input.

Both the user destination inference and the shared local controller are integrated to the wheelchair's platform as depicted in figure 4.1.

- **Destination Inference:** This functional module estimates the desired destination within the map of the environment among a list of potential choices. The list of possible destinations can be previously selected by an expert caregiver, the user, or learned automatically by the system. We propose a method to perform the inference of the desired destination using a Dynamic Bayesian Network that considers the current position of the wheelchair and direction of the commands given by the user. The destination with the highest probability value is selected and sent to the navigation module which is then fully in charge to drive the robotic wheelchair to the destination.
- **Shared Local Controller:** This part of the navigation subsystem is the one in charge of generating the appropriate velocity control to drive the wheelchair to the desired destination. It is based on a reactive navigation method called dynamic window (Fox et al., 1997). Unlike other conventional reactive navigation systems,

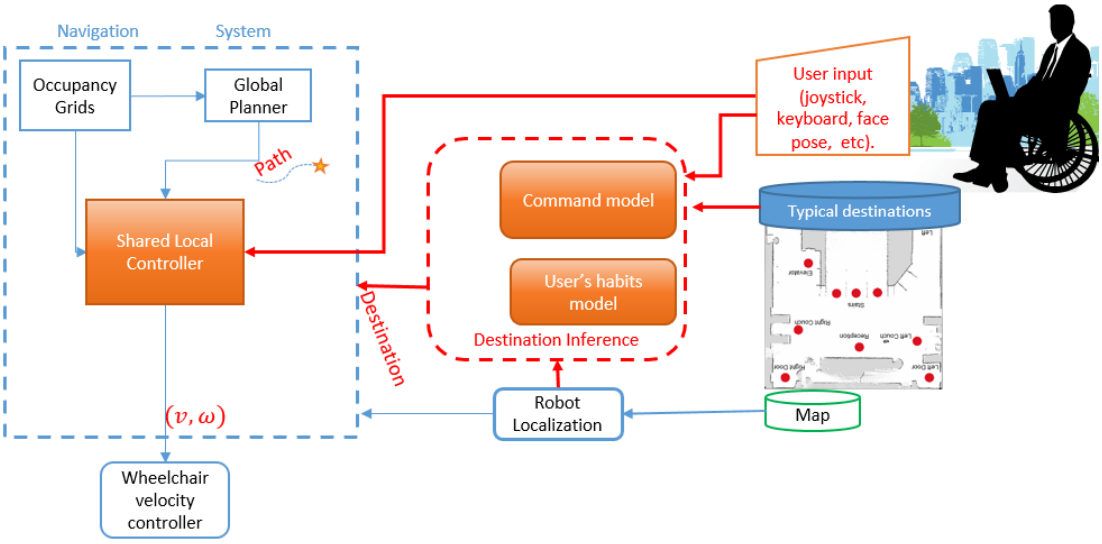


Figure 4.1: In this section we will present the destination inference and shared local controller subsystems -red blocks- of the robotic wheelchair. Both of blocks consider the input from the user. (In this figure we used general flow-chart symbols as cylinders for data-storage, squares for functional blocks and squares with a diagonal top to depict user entries).

our method considers the intention of the user and it will favor the generation of trajectories in the direction of the commands given by the user using the input device (joystick, a keyboard, etc).

4.2 Destination Inference Subsystem

In this section we present our method for inferring the user desired destination while driving the robotic wheelchair. The method is proposed to work in a known indoor environment as the user's office or home. The automated wheelchair receives the direction of the command given by the user, estimates the most probable destination among a predefined list and navigates autonomously to that place meaning that the robot is in charge of planning a path and compute the necessary speed to move towards the goal.

It is considered that the user could just be able to move one finger and so the destination should be inferred even when the user gives a simple command as pressing a key in a keyboard. Human intervention should be reduced to accomplish the navigation task but safety and comfort should always be guaranteed. In chapter 5 we will show

results using other interfaces as voice-command and face-pose recognition.

To deal with the uncertainty inherent to the intention recognition problem we use probabilistic reasoning. The model selected to represent the intention of the user is a Dynamic Bayesian Network that considers the set of possible destinations and takes as evidence the commands given by the user trough the input device and the current position of the robot to infer the intended destination. The inference is improved by adding some knowledge about the habits of the user in the form of a probability table that encodes how frequently the user visits each of the destinations.

Definition 8 *Possible destinations are the set of places where the user could desire to go in a given setting; it will be defined by the union of typical destinations, points of interest and meeting points present in the environment.*

Definition 9 *Typical destination are places where the user spends most of his time during the day. They can be learned automatically or set by the user or an expert caregiver.*

Definition 10 *A point of interest is a specific location or facility that someone may find useful or interesting in a given place such as a building, or any other public place.*

Definition 11 *Meeting points are those socially acceptable locations to join a group of people engaged in a conversation. These points are computed using a geometrical model based on measurements done in social sciences (described in chapter 3).*

Definition 12 *A high-level input is an order given by the user that does not control directly the velocity of the robotic wheelchair, but rather, gives information at a higher level of abstraction (e.g. the destination, a desired operation mode, the travel direction etc). This kind of commands can be given at a much lower rate than low-level input.*

Definition 13 *A low-level input commands is a signal sent by the user to the robot through any standard interface (presented in section 2.3). It gives explicitly the linear and angular velocities* (v, ω) *necessary to drive the wheelchair. When the user is in charge of the complete navigation task (manual mode) he will be constantly giving low-level input commands to the robot.*

4.2.1 Dynamic Bayesian Network to Infer the User Desired Destination

The Dynamic Bayesian Network (DBN), illustrated in Fig. 4.2, is used to infer the user desired destination; a random variable D_t is used to represent the set of

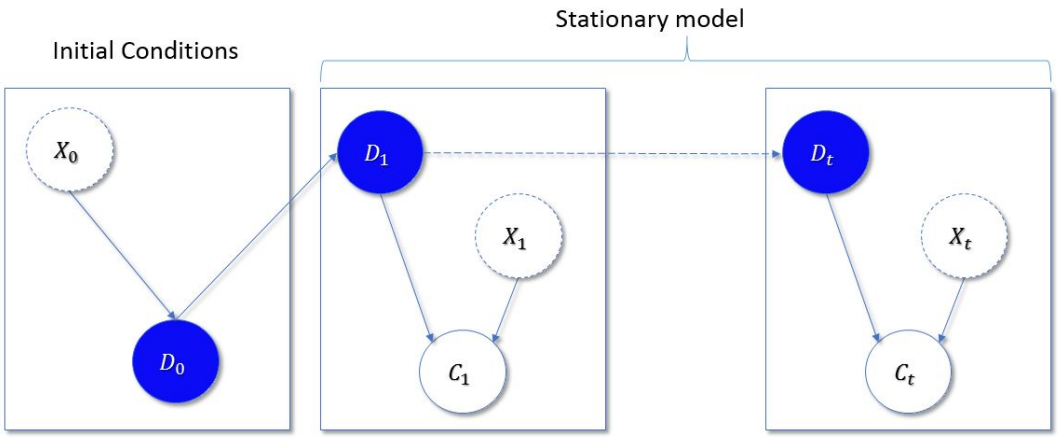


Figure 4.2: The Bayesian Network used to estimate the current user intended destination D_t. At each time step t, the posterior probability is updated by using the current position X_t and current command C_t. D_t also depends on the value of the last estimation D_{t-1} to take into account the history of given commands. Prior knowledge is expressed as the probability of going from each starting position X_0 to any of the possible destinations D_0. Hidden variables are in blue and observable states are dashed circles.

possible destinations. We use the notation D_t, C_t, X_t to indicate the value of the random variables D, C, X at discrete time t.

Whenever the user gives a new input command C_t, the posterior probability is updated taking the current position X_t and current command C_t as evidence values. The link between D_t and D_{t-1} is used to model the dependency of the current estimation with the previous history of input commands. The initial state of the system is represented in the DBN as the probability of going from a starting position X_0 to any of the possible destinations D_0. This is the part of the model that considers the user's habits.

The joint probability distribution that corresponds to the graph (Fig. 4.2) is:

$$P(D_{0:t}, C_{1:t}, X_{0:t}) = P(X_0)\ P(D_0|X_0) \prod_{k=1:t} [P(D_k|D_{k-1})P(C_k|D_k, X_k)\ P(X_k)] \quad (4.1)$$

Where,

– D: This is a random variable that represents the set of possible destinations in a given environment $D \in \{d^{(1)}, d^{(2)}, ..., d^{(i)}, ..., d^{(N)}\}$. where $d^{(i)}$ is the position (x, y) of each typical destination in the current scenario (as it is shown in Fig. 4.3).

- C: This variable represents the command given by the user to express his desired travel direction. This direction is centered on the user's local frame.
- X: This is the position of the wheelchair in a 2D plane with respect to the reference frame of the map. This position is discrete and its domain is defined by a grid whose size is according to the map of the environment.

At each time step t we want to infer which is the probability that the user intended destination at time t is $d^{(i)}$ knowing the history of commands and positions. This is expressed as the inference question $P(D_t = d^{(i)}|C_{1:t}, X_{0:t})$ which can be recursively computed using Bayes Rule as:

$$\begin{aligned} P(D_t^{(i)}|C_{1:t}, X_{0:t}) =& \eta P(X_t)P(C_t|X_t, D_t^{(i)})\cdot \\ & \sum_j [P(D_t^{(i)}|D_{t-1}^{(j)})P(D_{t-1}^{(j)}|C_{1:t-1}, X_{0:t-1})]; \quad \text{for } t > 1 \end{aligned} \tag{4.2}$$

Here, we are using the notation $D_t^{(i)}$ to indicate the argument $D_t = d^{(i)}$ defined before. η is a normalizing constant used to make probabilities sum up to 1.

$$\eta = \frac{1}{\sum_i P(D_t^{(i)}|C_{1:t}, X_{0:t})} \tag{4.3}$$

The term $P(D_{t-1}^{(j)}|C_{1:t-1}, X_{0:t-1})$ is obtained from the result of the previous iteration. In the particular case when $t = 1$ it takes its value from the initial probability table:

$$P(D_{t-1}^{(j)}|C_{1:t-1}, X_{0:t-1}) = P(D_0^{(j)}|X_0); \quad for \quad t = 1 \tag{4.4}$$

It is assumed that the position of the wheelchair X_t (Eq. 4.2) is a fully observable measurement given by the localization system. Therefore the probability distribution is a Dirac function centered on the measurement value leading to $P(X_t) = 1$. Therefore, this term has no effect on the final result and the equation can be simplified as follows:

$$\begin{aligned} P(D_t^{(i)}|C_{1:t}, X_{0:t}) =& \eta\ P(C_t|X_t, D_t^{(i)})\cdot \\ & \sum_j [P(D_t^{(i)}|D_{t-1}^{(j)})P(D_{t-1}^{(j)}|C_{1:t-1}, X_{0:t-1})]; \quad \text{for } t > 1 \end{aligned} \tag{4.5}$$

The term $P(C_t|X_t, D_t^{(i)})$ will be referred to as the **command model**, $P(D_t^{(i)}|D_{t-1}^{(j)})$

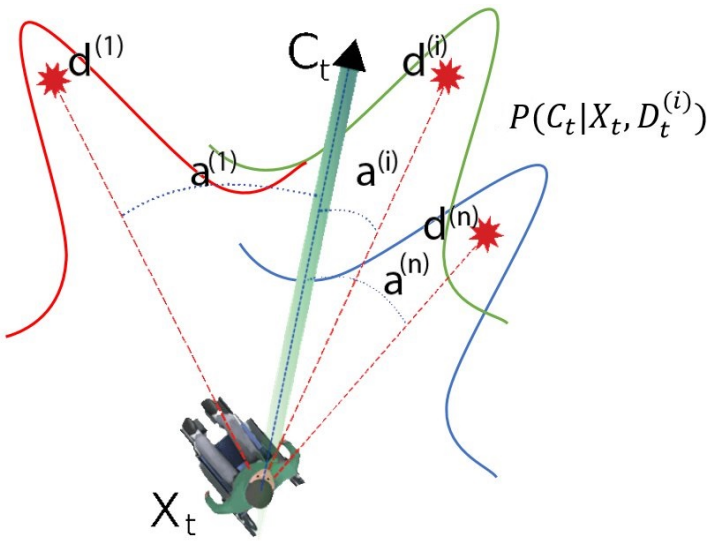


Figure 4.3: Command model. The probability value for a given command C_t is proportional to the angle $a^{(i)}$ between the sight-line and each typical destination $d^{(i)}$ in the environment.

is the **transition model** and $P(D_0|X_0)$ is the **initial probability model**. We will now explain each of those terms.

4.2.1.1 Command Model

The command model estimates where the user will point to, assuming that we know which is his destination. The probability value for a given command C_t is proportional to the angle $a^{(i)}$ between the line in the direction of the command and each typical destination $d^{(i)}$ in the environment as depicted in Fig 4.3.

$P(C_t|X_t, D_t^{(i)})$ represents the probability that a command C_t will be sent by the user when he is located at position X_t and his destination is $d^{(i)}$ at current time t.

Under the assumption that the user will try to give commands directed straight forward to the destination position as shown in Fig. 4.3. $P(C_t|X_t, D_t^{(i)})$ is computed as a normal distribution.

$$P(C_t|X_t, D_t^{(i)}) = \frac{1}{\sigma\sqrt{2\pi}} \exp^{-\frac{1}{2}(\frac{a_i}{\sigma})^2} \tag{4.6}$$

The a_i term is the angle between the command and the destination, the value of the standard deviation σ depends on the type of interface.

4.2.1.2 Transition Model

$P(D_t^{(i)}|D_{t-1}^{(j)})$ expresses the probability that the user intended destination changes over the time t. This term will bring some robustness against involuntary commands, therefore, it is very important when using the face pose interface presented in section 5.2.1. Large values of this term will lead to slow responses whenever the user changes his intended destination.

$$P(D_t^{(i)}|D_{t-1}^{(j)}) = \begin{cases} (gain-1)/(N+gain-1) & \text{if } D_t^{(i)} = D_{t-1}^{(j)} \\ 1/(N+gain-1) & \text{otherwise} \end{cases} \tag{4.7}$$

where:

- N: This is the number of possible goals in the environment.
- $gain$: This term expresses how important the previous estimation will be with respect to the new data. This term has to be adjusted depending on the input device.

4.2.1.3 Initial Probability Model

The initial probability $P(D_0|X_0)$ uses prior information about the user habits. If there is no previous knowledge, a uniform distribution will be assumed. However, if the user is in a known environment, the most frequented places can be extracted by observation so that this probability distribution will be learned from the environment.

In the presented approach we defined those typical destinations from previous observations of the experimental scenario. The probability $P(D_0|X_0)$ is a probability table set by counting the number of times that a destination is visited departing from any of the other important places in the environment. This value could also be set manually by the user or the caregiver. This table will be different for each scenario, one example is presented in figure 4.4 that leads to the probability table shown at figure 4.4(c).

4.2.1.4 Goal Selection

Finally, the destination inference module selects the goal with the maximum posterior probability. It performs exact inference using the equation:

$$d_{sel} = \arg\max_{d^{(i)}} P(D_t = d^{(i)}|C_t, X_t) \tag{4.8}$$

The selected destination is then sent to the autonomous navigation system if its probability value is bigger than a given threshold ϕ, which was fixed by experimentation.

$$d_{wheelchair} = \begin{cases} d_{sel} & \text{if } P(D_t = d_{sel}|C_t, X_t) >= \phi \\ null & \text{otherwise} \end{cases} \tag{4.9}$$

4.2.2 Simulation Results

The user intentions model was integrated in the architecture of our simulated wheelchair as illustrated in figure 4.1. The simulation environment was constructed using real data collected on the GERHOME experimental apartment. Real elderly people, aged from 64 to 85 years old, have been observed in the experimental site during an average period of time of 4 hours. The collected data include the 36 video streams and data provided by 24 environmental sensors can be accessed on the web-page of the Gerhome project. [1]

The map of the apartment was built using laser data and it is shown in figure 4.4. White zones in the map of the environment represent free space, black are occupied regions and gray zones are not known regions.

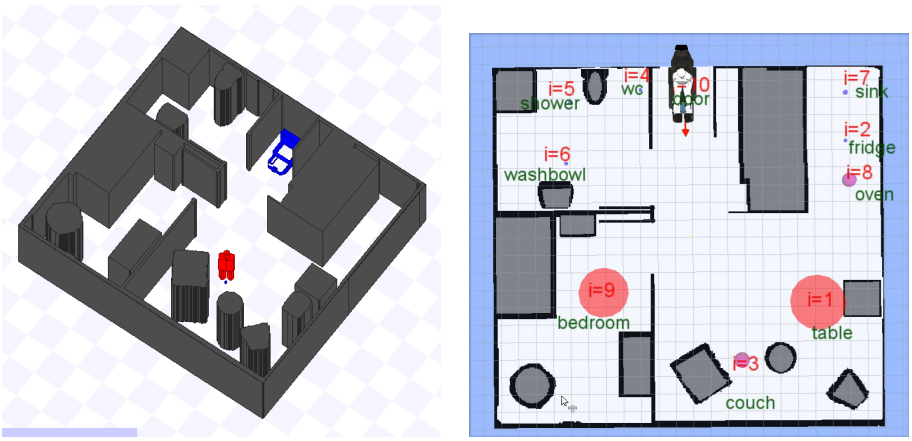
Possible destinations were extracted from real data by tracking people in the scene and marking places where they spent most of his time. In the figure 4.4 the points corresponding to possible destinations are marked with circles. Each destination has an associated probability value $P(D_0|X_0)$ shown in table 4.4(c). In practice, those elements with value equal to 0 are replaced with a very small value ϵ.

In the example situation, the user is asked to control the wheelchair only by pointing (for simulations we use a keyboard) in the direction of his intended destination (the direction is marked with a red arrow). The user can stop the wheelchair at any time by sending a "STOP" signal. The possible destinations are marked with red spheres with a size proportional to the computed posterior probability.

In the first example (Fig. 4.5(a)) the user is asked to drive the wheelchair from the door $d^{(10)}$ to the left table $d^{(1)}$, while in the second example (Fig. 4.5(d)) he has to go to the bedroom on the right $d^{(9)}$. Notice that for the given starting point $X_0 = d^{(10)}$ both $d^{(1)}$ and $d^{(9)}$ have similarly high initial probability values.

In the first case, when the user points to the left and sends the initial command (GO), the probabilities get updated with the new information. As the system is more confident that the user wants to go to the left table, thus the wheelchair takes it as the desired destination. The same situation happens when the user points to the right and, in this case, the wheelchair starts moving towards the right bedroom.

1. Data available at www.sop.inria.fr/members/Francois.Bremond/topicsText/gerhomeProject.html.

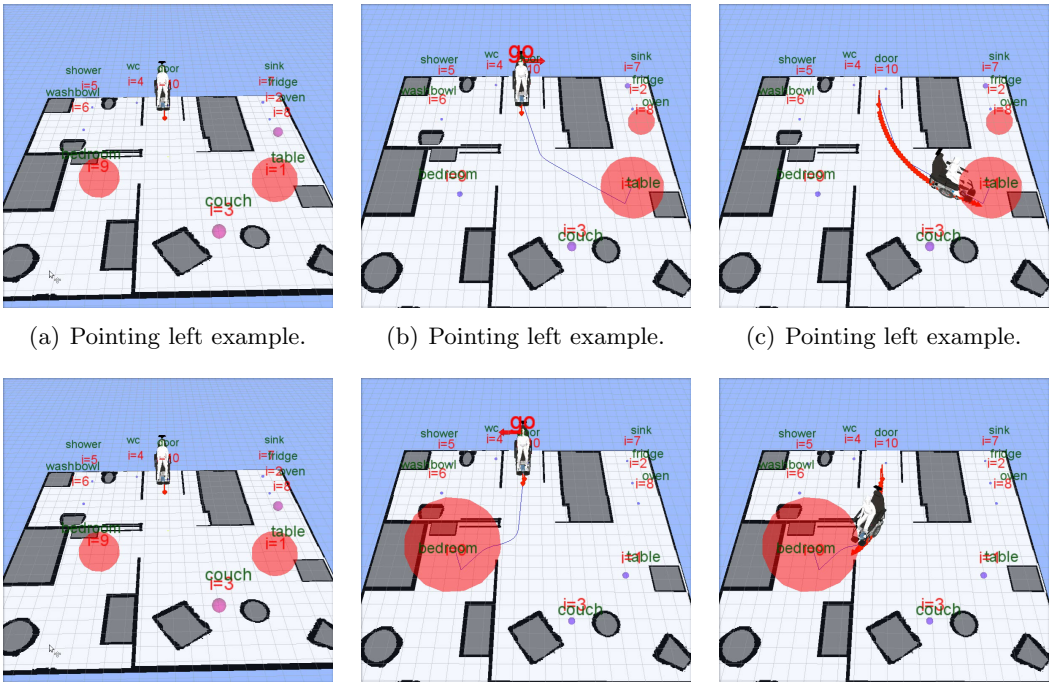


(a) (b)

	$d^{(1)}$	$d^{(2)}$	$d^{(3)}$	$d^{(4)}$	$d^{(5)}$	$d^{(6)}$	$d^{(7)}$	$d^{(8)}$	$d^{(9)}$	$d^{(10)}$
$x_0=d^{(1)}$	0.000	0.330	0.182	0.028	0.001	0.057	0.267	0.063	0.044	0.028
$x_0=d^{(2)}$	0.099	0.000	0.016	0.013	0.005	0.007	0.633	0.223	0.002	0.002
$x_0=d^{(3)}$	0.525	0.169	0.000	0.034	0.017	0.034	0.136	0.085	0.000	0.000
$x_0=d^{(4)}$	0.161	0.129	0.000	0.000	0.000	0.352	0.323	0.032	0.000	0.003
$x_0=d^{(5)}$	0.500	0.182	0.000	0.000	0.000	0.182	0.133	0.000	0.000	0.003
$x_0=d^{(6)}$	0.267	0.133	0.111	0.089	0.089	0.000	0.289	0.000	0.021	0.001
$x_0=d^{(7)}$	0.094	0.712	0.014	0.025	0.010	0.020	0.000	0.117	0.002	0.006
$x_0=d^{(8)}$	0.065	0.620	0.025	0.005	0.010	0.004	0.270	0.000	0.000	0.001
$x_0=d^{(9)}$	0.301	0.077	0.077	0.000	0.154	0.154	0.154	0.007	0.000	0.076
$x_0=d^{(10)}$	0.420	0.000	0.100	0.000	0.000	0.000	0.100	0.090	0.380	0.000

(c)

Figure 4.4: The GERHOME simulated environment. (a) Simulation in STAGE. (b) Map of the environment with the typical destinations marked with circles. The size of the circles represents the value of the initial probability distribution for the given starting point $X_0 = d^{(10)}$, here we can see that according to the table the destination with the highest initial probability is $d^{(1)}$ followed by $d^{(9)}$. (c) Initial probability table. In practice, those elements with value equal to 0 are replaced with a very small value ϵ.



(a) Pointing left example. (b) Pointing left example. (c) Pointing left example.

(d) Pointing right example. (e) Pointing right example. (f) Pointing right example.

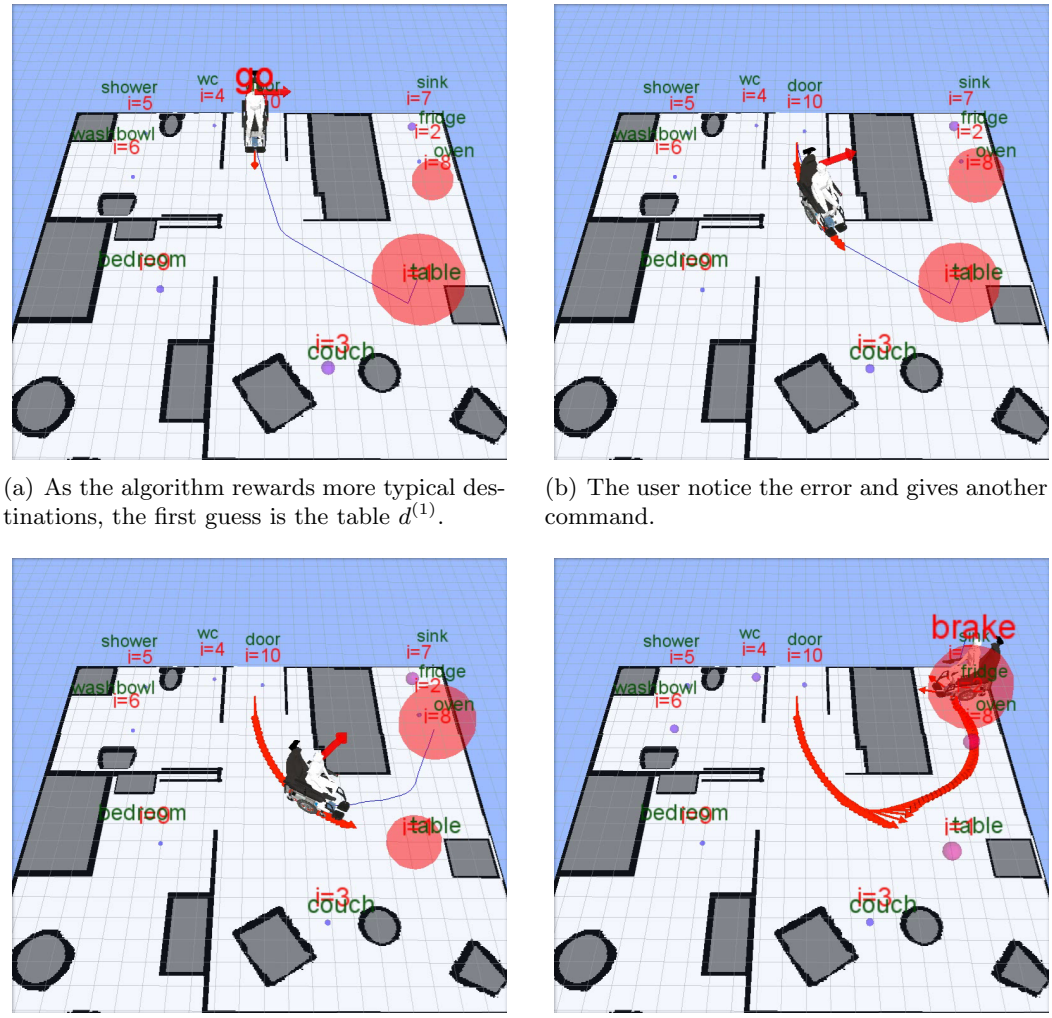
Figure 4.5: Moving towards a typical destination. In this example the user commands the wheelchair to go from the door $d^{(10)}$ to the table in the left $d^{(1)}$ or the bedroom to his right $d^{(9)}$. In the starting state the destinations have a similar prior probability value ((a), (d)). When the user sends the initial GO signal, the probabilities are updated according to the direction pointed by the user and the best destination is sent to the robot ((b), (e)). The robot moves autonomously towards the goal ((c),(f)).

The navigation subsystem is in charge of planning the path to the desired destination and avoiding obstacles (see Fig. 4.1).

4.2.3 Limitations of the System.

When the user moves in a trajectory that does not correspond to his normal habits; for example, moving from the door $d^{(10)}$ to the sink $d^{(7)}$ in the GERHOME scenario. In this case, the initial probability is low.

In the example test shown in figure 4.6 the wheelchair is initially located at the door $d^{(10)}$, the user points to the left and sends the initial order (GO). Due to the high prior probability to go to the table $d^{(1)}$, the wheelchair takes it as the most likely solution. This first guess turns out to be an error, in response, the user sends a new command which again is erroneous so the user has to give more commands (in this case up to 6)



(a) As the algorithm rewards more typical destinations, the first guess is the table $d^{(1)}$.

(b) The user notice the error and gives another command.

(c) The probabilities are updated and a new destination is selected.

(d) Arriving to the destination

Figure 4.6: The user is asked to move the wheelchair from the door $d^{(10)}$ to the sink $d^{(7)}$ (which is not a frequent destination according to the user's habits). To deal with this kind of situations the user will have to give more than one command but anyway the number of necessary commands to arrive to a given destination will be reduced.

before the wheelchair arrives to the desired destination.

This situation can be frustrating for the user, however, the idea of the proposed approach is to help the user accurately most of the times and even if the will have to correct a bad inference by giving more than one command. In the end the number of necessary commands to arrive to a destination will be reduced.

As the wheelchair is in charge of the full planning task, the user is relieved of this

workload of driving himself. However, it also means that the user can not make small adjustments in the trajectory when moving towards the goal. That can be a severe limitation of the system that we will try to correct in section 4.3.2.

4.2.4 Extension of the System to Consider Dynamic Destinations

We have just described how to infer the user's intended destination among a list of static goals, now we will explain our proposal to extend the system to consider new destinations that appear in the scene. The application described here consists in joining groups of people engaged in a conversation by sending the wheelchair to the detected meeting points.

The meeting points computed in section 3.3 are added to the list of possible destinations and they are considered when inferring the user intention. This system is expected to be used with a multi-modal interface to enrich the available data to improve the initial guess. The vocal command is used to indicate the start of a navigation task. It will be similar to the initial command -GO- used in previous section, however, now we will consider two possible orders *GO* and *JOIN*. The GO command is modeled to be more ambiguous and it adds less information to the inference which in turn will depend on the user's habits and pointed direction as explained in section 4.2. However, the JOIN verb expresses more explicitly the user desire to join a group, therefore it adds new contextual information that will be considered by the inference system.

A new random variable V_0 is added to the network as shown in figure 4.7. This variable represents the possible value of the initial vocal command.

The variable D now represents both the static destinations and the meeting points. The domain of the D variable (denoted as Ω_D) is defined as the union between the set of static destinations Ω_S and the set of meeting points Ω_M.

$$\Omega_D = \Omega_S \vee \Omega_M \tag{4.10}$$

Where,

- Ω_D is the domain of the D variable.
- $\Omega_S = \{d^{(1)}, d^{(2)}, ..., d^{(N)}\}$ is the set of static destinations.
- $\Omega_M = \{d^{(N+1)}, d^{(N+2)}, ..., d^{(N+M)}\}$ is the set of M meeting points.

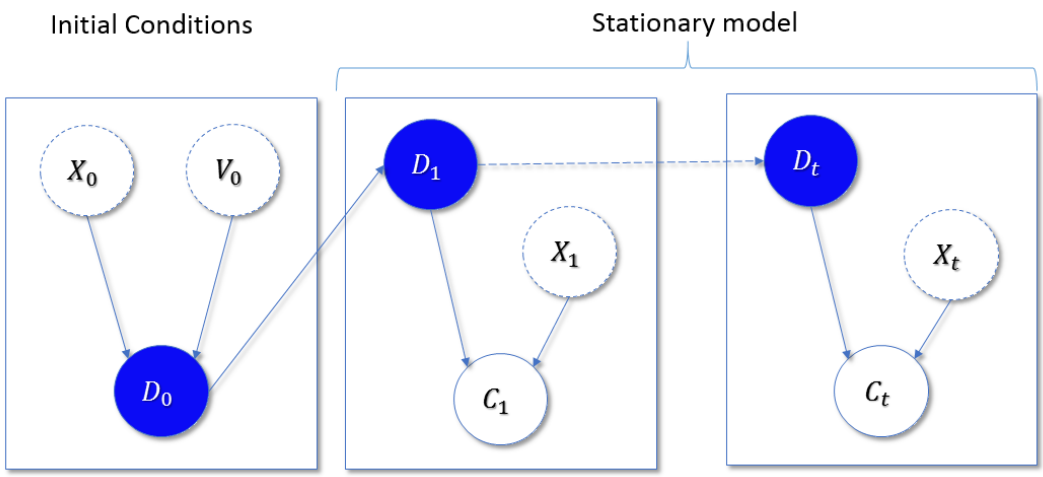


Figure 4.7: The Dynamic Bayesian Network used to estimate the current user intended destination D_t. The initial state of the DBN depends on the starting position X_0 and initial vocal command V_0. At each time step t, the posterior probability is updated by using the current position X_t and current command C_t. D_t also depends on the value of the last estimation D_{t-1} to take into account the history of given commands.

The joint probability distribution that corresponds to the graph (Fig. 4.7) is:

$$P(D_{0:t}, C_{1:t}, X_{0:t}, V_0) = P(X_0)\, P(V_0)\, P(D_0|X_0, V_0) \prod_{k=1:t} [P(D_k|D_{k-1}) P(C_k|D_k, X_k)\, P(X_k)] \tag{4.11}$$

Where,

- X_t is the position of the wheelchair in a 2D plane with respect to the map reference frame. This position is discrete and its domain is defined by a grid whose size is according to the map of the environment. It is assumed that the position of the wheelchair X_t is a fully observable measurement given by the localization system.
- V_0 is the starting vocal command. This variable can take two possible values $\{GO, JOIN\}$.
- D_t is a random variable that represents the user desired destination at time t. This is a discrete variable whose domain is defined as the set of possible destinations in a given environment.
- C_t represents the direction signaled by the user at time t. This direction is centered on the user's local frame.

At each time step t we want to infer $P(D_t = d^{(i)}|C_{1:t}, X_{0:t}, V_0)$. Using Bayes rule we find that it can be recursively computed as:

$$P(D_t^{(i)}|C_{1:t}, X_{0:t}, V_0) = \eta P(X_t)\ P(C_t|X_t, D_t^{(i)}) * \sum_j [P(D_t^{(i)}|D_{t-1}^{(j)}) P(D_{t-1}^{(j)}|C_{1:t-1}, X_{0:t-1}, V_0)] \quad (4.12)$$

Once again the notation $D_t^{(i)}$ is used to express $D_t = d^{(i)}$ and η is a normalizing constant used to make probabilities sum up to 1.

$$\eta = \frac{1}{\sum_i P(D_t^{(i)}|C_{1:t}, X_{0:t}, V_0)} \quad (4.13)$$

The term $P(D_{t-1}^{(j)}|C_{1:t-1}, X_{0:t-1}, V_0)$ is obtained from the result of the previous iteration. In the particular case when $t = 1$ it takes its value from the initial probability model.

$$P(D_{t-1}^{(j)}|C_{1:t-1}, X_{0:t-1}, V_0) = P(D_0^{(j)}|X_0, V_0); \quad for \quad t = 1 \quad (4.14)$$

It is assumed that the position of the wheelchair X_t and the vocal command V_0 are fully observable measurements, therefore, their probability distributions are Dirac functions centered on the measurement value leading to $P(X_t) = 1$, $P(V_0) = 1$, therefore, those terms do not affect the final result and the equation can be simplified as follows:

$$P(D_t^{(i)}|C_{1:t}, X_{0:t}, V_0) = \eta P(C_t|X_t, D_t^{(i)}) * \sum_j [P(D_t^{(i)}|D_{t-1}^{(j)}) P(D_{t-1}^{(j)}|C_{1:t-1}, X_{0:t-1}, V_0)] \quad (4.15)$$

The "command model" $P(C_t|X_t, D_t^{(i)})$ and the "transition model" $P(D_t^{(i)}|D_{t-1}^{(j)})$ are exactly the same as explained in sections 4.2.1.1 and 4.2.1.2 respectively, so they will not longer be explained.

The "initial probability model" $P(D_0|X_0, V_0)$ now has two different modalities according to the values of the initial vocal command. When $V_0 = GO$ it is the same as

the term $P(D_0|X_0)$ term presented in section 4.2.1.3.

$$P(D_0^{(i)}|X_0, V_0 = GO) \equiv P(D_0^{(i)}|X_0) \tag{4.16}$$

When $V_0 = JOIN$ the meeting points get an evenly distributed high probability while all the others get low values. the non-normalized distribution is expressed in Eq. 4.17.

$$P(D_0^{(i)}|X_0, V_0 = JOIN) = \begin{cases} \alpha/M & \text{if } d^{(i)} \in \Omega_m \\ (1-\alpha)/N & \text{otherwise} \end{cases} \tag{4.17}$$

In our tests we used $\alpha = 0.9$.

The proposed approach was designed considering the scenario shown in Fig. 4.4. We use again the prior probability table learned for our simulated scenario (Fig. 4.4(c)).

The user can start the test at any location of the experimental scenario, he drives the wheelchair by pointing towards his desired destination and giving the *JOIN* or *GO* command he starts the navigation task. In the example situation illustrated in figure 4.8, the user is pointing in the direction of the arrow (blue) and there are two people in the middle of the simulated INRIA-hall. The task of the wheelchair is to navigate towards the goal that gets the highest posterior probability.

When the user gives the signal to start the movement (GO or JOIN) the prior probabilities are loaded and the posterior probability is computed considering the direction of his face, the probability for each goal is depicted in figures 4.8 and 4.9 as the size of the sphere. The goal with the highest posterior probability is sent to the navigation module, the wheelchair plans the path and computes the necessary velocity to reach the destination while avoiding obstacles.

The destination inference method considers both the meeting points defined by the two persons in vis-a-vis formation. In the first case 4.8 the user gives a JOIN order. Given the strong context related to this verb the probabilities are computed in such a way that the meeting points get a higher probability value than all the other possible destinations. The information acquired between the pointed direction and the initial order JOIN is enough for the system to infer where to go.

In the second case 4.9 the user gives a GO order which does not give as much contextual information as the JOIN command, so the probability distribution is more evenly distributed, therefore, the destination that gets the maximum probability value is mainly defined by the direction in which the user is looking.

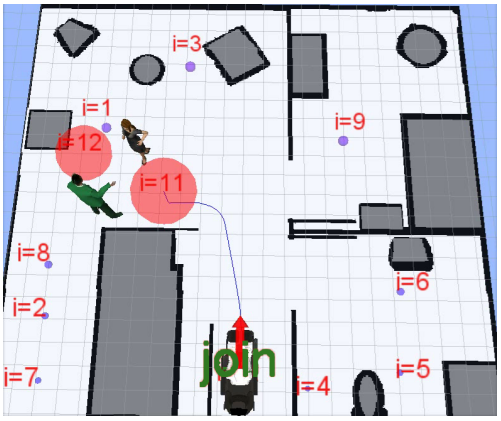


Figure 4.8: In this example; even if the pointed direction is ambiguous between $d^{(9)}$ and $d^{(11)}$, when the user sends the JOIN initial order, the probabilities are computed in such a way that those destinations defined by a group of people get a higher value.

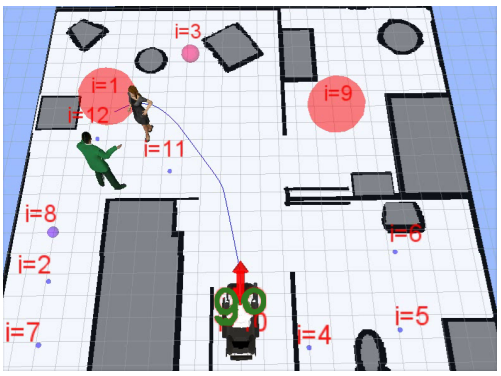


Figure 4.9: The GO initial order does not give as much contextual information as the verb JOIN; therefore, the destination that gets the maximum probability is mainly defined by the direction pointed by the user.

4.2.5 Discussion

In this section we presented two methods to infer the desired destination of the user and how it was integrated with the autonomous navigation system of the wheelchair. Both methods were designed to improve both the usability and the compliance of the wheelchair.

The first method considered only static destinations and we presented the idea of inferring the desired destination considering the habits of the user and the direction of the commands given by the user. This method considers that the set of possible destinations where a user is likely to go during the day is reduced and this can be used to simplify the problem of inferring the intention of the user.

This probabilistic reasoning method allows to consider the inherent uncertainty in the estimation task, even if the system is not certain about the intention of the user it can still favor the most "reasonable" actions when presented with an ambiguous command.

The main contribution of the second approach is to consider moving destinations defined by meeting points. This extension uses information from different inputs - voice and face position-, to reduce ambiguities in the inference model.

Real tests will be presented in chapter 5.

4.3 Shared Local Controller

We just presented a way to infer the most probable destination and then the wheelchair moves autonomously towards that destination. In this section we will present a method to consider the intention of the user also in the computation of the adequate velocity commands to move to the destination.

The intention of the user has to be considered because otherwise it is very awkward for him when the trajectories planned by the robot differs from what he expected. This is a quite frequent situation when using a fully autonomous navigation system, since human interpretation of the environment often differs from that of the robot. This undesirable behavior may be so annoying that with time the user could stop using the system.

To increase the compliance and acceptability of the system, the navigation strategy should be shared between the robot and the human. The shared controller presented in this section takes as input directional commands from the user and is based on the *dynamic window approach* to avoid obstacles while moving closely to the last user

command. We selected this method due to its reliability, low computational cost and its capability of producing a soft movement.

A brief introduction about the dynamic window controller will be presented, then a modified version of the algorithm considering the input from the user is described. Finally, some simulation results and the performance of the proposed idea will be discussed.

4.3.1 The Dynamic Window Approach

The *dynamic window controller* (Fox et al., 1997) presents a way to select appropriate velocity commands (v, ω) to navigate safely among obstacles. The instantaneous linear v and angular ω velocities of the robot are directly chosen in the velocity space bounded by some constraints imposed by the obstacles in the environment and the physical characteristics of the robot.

Different constraints over the robot's velocities are used to reduce the size of the search space; it considers that the robot moves in circular trajectories, the robot's physical constraints define the maximum possible velocities (V_p), reachable velocities or dynamic window (V_d) are those velocities that can be reached within a short time interval given the limits in linear and angular accelerations and the admissible velocities (V_a) consider only safe trajectories.

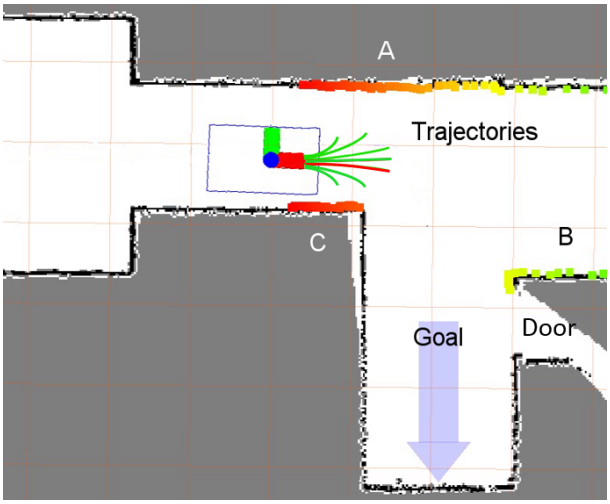
The resulting search space (Vr) is defined as:

$$V_r = V_p \cap V_a \cap V_d \tag{4.18}$$

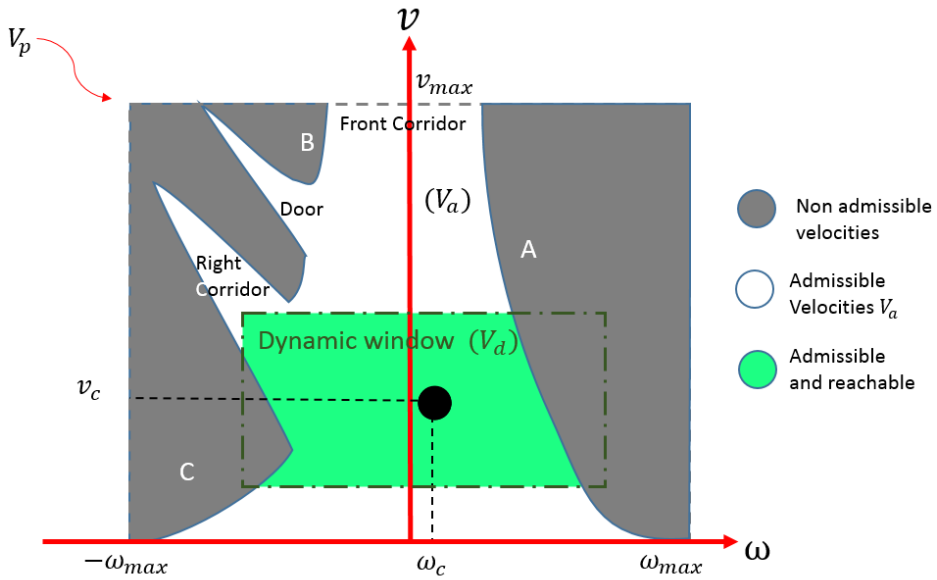
In Fig. 4.10(a) we show a situation of a robot moving in a corridor, the footprint of the robot is represented by the rectangle while obstacles are marked as gray areas. The trajectories generated by the dynamic window method are the constant curvature green arcs and for this example the trajectory with the best value of the cost function (explained later) is shown in red. The terms of equation 4.18 will be now further explained.

Circular Trajectories Constraint:

The robot is considered to move with constant linear and angular velocities (v, ω) during each control loop which results in a 2D search space. This assumption is valid when the period of the control loop is small so that the robot's acceleration is consider to be null during the full cycle. As a result the robot is modeled following circular trajectories with a constant curvature $c = \omega/v$ as those in the example situation of figure 4.10(a).



(a) The blue rectangle is the footprint of the robot, green arcs are possible trajectories and the red one is the one with the highest value of the objective function for the given situation.



(b) The velocity search space for the presented scenario. The dynamic window V_d is centered on the current velocity (v_c, ω_c) and its size is regulated by the characteristics of the robot as the maximum acceleration and obstacles in the environments (gray regions). Turning with high angular velocities would lead to collisions with the walls so those speeds are excluded from the search space. Trajectories with low angular velocity (going straight to the front) are accepted as the opening of the corridor is in front of the robot.

Figure 4.10: Trajectories produced by the dynamic window algorithm and the corresponding velocity search space.

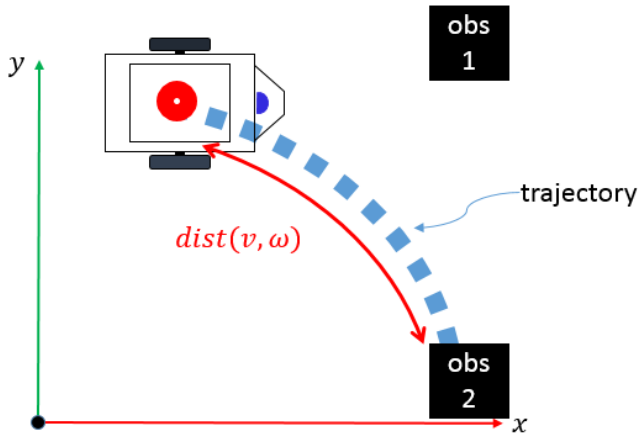


Figure 4.11: The cost function $dist(v, \omega)$ considers the minimum distance before colliding with an obstacle over the sampled trajectory.

Possible Velocities Constraint (V_p) This set is defined by the robot's physical constraints. It is defined as the maximum linear and angular velocities that the robot can exert. There is no need to consider velocities out of this set because the robot will not be able to execute them anyway.

$$V_p = \{(v, \omega) | v \in [v_{min}, vmax] \wedge \omega in[-\omega_{max}, \omega_{max}]\} \tag{4.19}$$

Admissible Velocities Constraint (V_a):

The number of possible admissible velocities are constrained by the obstacles near the robot. The set of velocities (v, ω) which allow the robot to stop before colliding with an obstacle is denoted V_a. The maximum admissible velocity, over a given curvature, depends on the distance $dist(v, \omega)$ to the nearest obstacle over that curvature as shown in Fig. 4.11. The set of admissible velocities (V_a) is then defined by:

$$V_a = \left\{ (v, \omega) | v \leq \sqrt{2 * dist(v, \omega) * \dot{v}_{max}} \quad \text{and} \quad \omega \leq \sqrt{2 * dist(v, \omega) * \frac{\dot{\omega}_{max}}{c}} \right\} \tag{4.20}$$

where, $\dot{v}_{max}$ and $\dot{\omega}_{max}$ are the maximum translational and rotational accelerations respectively; c is the curvature of the trajectory as it was previously described.

In the example figure 4.10(b), it can be observed that turning with high angular

velocities could lead to collisions with the walls, so those speeds are excluded from the search space (gray areas C and B. On the other hand, straight trajectories (with low angular velocity and large linear speed) are accepted inside the search space (green region) given that the opening of the corridor is in front of the robot.

Reachable Velocities Constraint (Dynamic Window) (V_d):

This set defines the *dynamic window* that constraints the search space to those velocities that the robot can achieve during the next control loop. The dynamic window is centered around the current velocity and its size depends on the accelerations that can be applied. It is expressed as:

$$V_d = \{(v,\omega)|v \in [v_c - \dot{v}_{max} * \Delta t, v_c + \dot{v}_{max} * \Delta t] \quad \wedge \\ \omega \in [\omega_c - \dot{\omega}_{max} * \Delta t, \omega_c + \dot{\omega}_{max} * \Delta t]\} \quad (4.21)$$

where, V_d is the dynamic window, v_c and ω_c are the current translational and rotational velocities, $\dot{v}_{max}$ and $\dot{\omega}_{max}$ are the maximum linear and angular accelerations and Δt is the duration of the control loop.

4.3.1.1 Cost Function

The objective function includes a measurement that quantifies the progress towards the goal, the forward velocity of the robot and the distance to the next obstacle on the trajectory. By combining them, we obtain a trade-off between how fast the robot will reach the goal and how safe the drive will be.

The objective function is defined as:

$$G(v,\omega) = \sigma * (\alpha * heading(v,\omega) + \beta * dist(v,\omega) + \gamma * vel(v,\omega)) \quad (4.22)$$

where,

- *heading* is a measure of progress towards the goal location. It is maximal if the robot moves directly towards the target:

$$heading = 180 - \theta \quad (4.23)$$

  θ is the angle of the target point relative to the robot's heading direction as shown in figure 4.12.
- *dist* is the distance to the closest obstacle on the trajectory (curvature) as it is

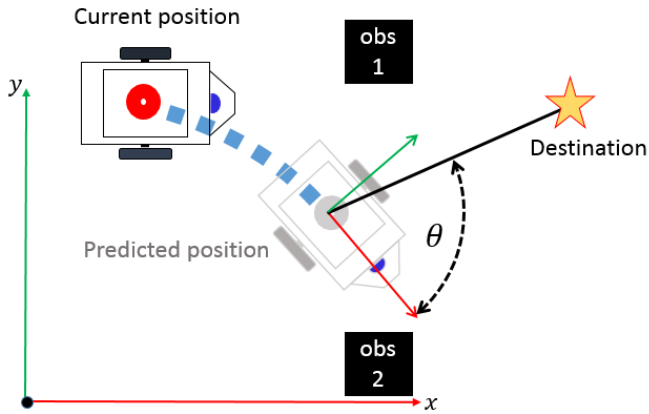


Figure 4.12: The heading cost reinforces trajectories that moves directly towards the goal. The angle θ is relative to the robot's heading direction in the last point of the simulated trajectory and the goal.

shown in Figure 4.11. If no obstacle is on the curvature this value is set to a large constant.

– *vel* is the forward velocity of the robot normalized on the maximum velocity v_{max}. This term reinforces fast movements.

$$vel(v, \omega) = v/v_{max} \tag{4.24}$$

– α, β, γ are weighting factors. To normalize all three components of the objective function to $[0, 1]$.

– σ smooths the weighted sum of the three components and results in more side-clearance from obstacles

The combination of translational and rotational velocity is chosen within the dynamic window by maximizing the objective function as follows:

$$(v_{best}, \omega_{best}) = \arg\max_{(v,\omega)} G(v, \omega) \tag{4.25}$$

In a more recent work Marder-Eppstein et al. (2010) modified the dynamic window cost function by replacing the *heading* and *vel* terms with two navigation functions defined as the distance to the shortest unobstructed path to the goal $pDist(v, \omega)$ and the distance to the goal $gDist(v, \omega)$. Thus they were able to eliminate the local minima problems present in many obstacle avoidance schemes. The obstacles were represented in a grid which assigns a occupancy cost to each cell in the grid as it is presented in

section 3.2.1.

$$G_r(v,\omega) = \alpha * pDist(v,\omega) + \beta * gDist(v,\omega) + \gamma * occCost(v,\omega) \tag{4.26}$$

where,

- $gDist(v,\omega)$ is the distance to the goal from the endpoint of the trajectory (Fig. 4.13).
- $pDist(v,\omega)$ is the distance to path from the endpoint of the trajectory, when no path is available this term will be equal to 0 (Fig. 4.13).
- $occCost(v,\omega)$ is the maximum obstacle cost along the trajectory. This value is taken from the occupancy grid used to represent obstacles around the robot as presented in section 3.2.1 (Fig. 4.13).
- α is a weighting factor that controls how close the robot should stay to the path.
- β is the weighting factor for how much the controller should attempt to reach the goal. It also controls speed.
- γ is the weighting factor to specify how much the controller should attempt to avoid obstacles.

Using this modified cost function the best trajectory selected at each control cycle is the one with the lowest value (Eq. 4.27). Note that this equation is different to equation 4.25 because the best trajectory is the one with the minimum value.

$$(v_{best}, \omega_{best}) = \arg\min_{(v,\omega)} G_r(v,\omega) \tag{4.27}$$

The dynamic window approach has been used to perform autonomous navigation in indoor environments with good and consistent practical results (Fox et al., 1997). Marder-Eppstein et al. (2010) allowed a robot to complete 26.2 miles of autonomous navigation in a real office environment when using the dynamic window approach together with an A* path planning.

4.3.2 User Intention Aware Dynamic Window

Unlike a normal mobile robot, a robotic wheelchair will always be ridden by a human. It makes highly important to consider the orders given by the user when selecting the appropriate velocity commands, otherwise, the loss of control can not only cause frustration to the user but also be dangerous since it avoids to correct possible errors made by the robot.

One limitation of the dynamic window controller previously presented is that it does not take into account the possible input from the user. In our proposed solution, the

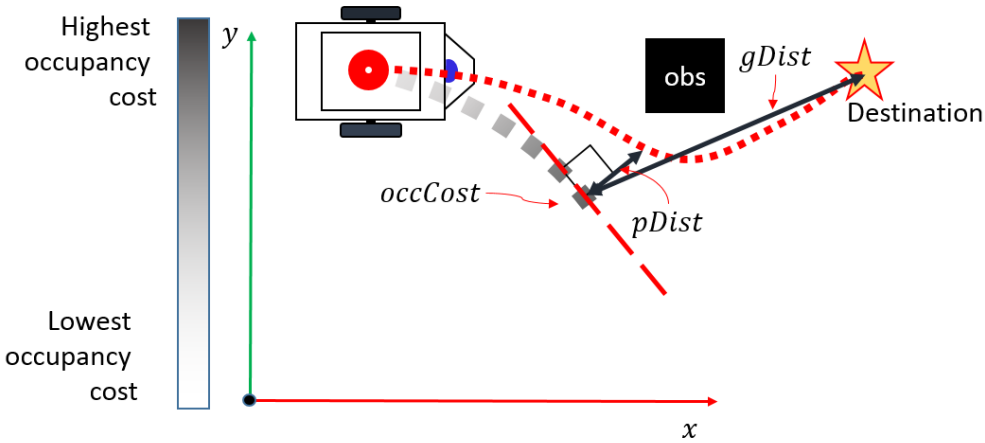


Figure 4.13: The cost function proposed by (Marder-Eppstein et al., 2010). It takes into account the distance to the global path, the distance to the goal and distance to obstacles to score a trajectory. The function $pDist$ measures the distance to the global path from the end point of the trajectory in map cells or meters, $gDist$ is the distance to the goal from the endpoint of the trajectory. The $occCost$ function returns the maximum obstacle cost along the trajectory.

user will indicate his desired direction of travel by using the input device at discrete time. Then the wheelchair performs the low level control (computing the necessary velocity commands) to avoid obstacles or to keep the wheelchair moving in the desired direction.

4.3.2.1 Mapping from directional orders to speed commands

In manual driving, when the user sends an order through the input device there is always a direct mapping between the magnitude of the input signal and the resulting velocity applied to the wheelchair.

As it was introduced in section 2.3 wheelchair drivers can use many different type of interfaces. However, the input received from all of them is always mapped into a corresponding value of the instantaneous linear and angular velocity.

In this work we implemented a mapping function that translates the direction pointed by the user to a $(v_{user}, \omega_{user})$ velocity command that is then used by the local planner.

First consider the example of a joystick (Fig.4.14) in which the deflection of the handle translates directly to some linear and angular velocities. The joystick handle position is represented in a Cartesian coordinate system, with two axis, x and y, which

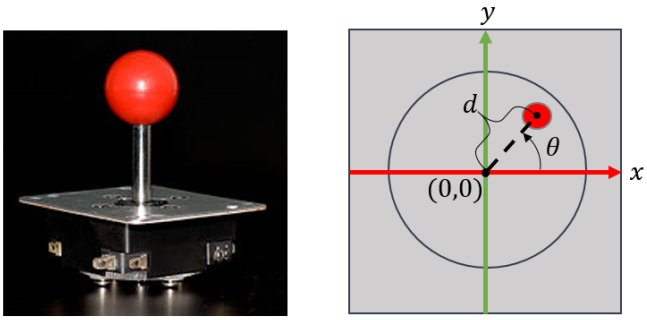


Figure 4.14: The position of the joystick gimball is translated to a velocity command $(v_{user}, \omega_{user})$.

vary between -1 and 1. These (x, y) coordinates can be used to determine the distance (d) of the handle to the central (resting) position of the joystick $(0, 0)$ and an angle θ relating to the $x - axis$ of the reference frame. Then the angular and linear velocities indicated by the user $(v_{user}, \omega_{user})$ are computed as:

$$\omega_{user} = k_1 \ \theta \tag{4.28}$$

$$v_{user} = k_2 \ d \tag{4.29}$$

Where, the angle θ is measured in relation to the x-axis; when θ is undefined , e.g. when $(x, y) = (0, 0)$, the wheelchair stops $(v_{user} = 0, \omega_{user} = 0)$. Otherwise, the θ angle is mapped to a given value of the angular speed ω. The distance d is measured from the center of the reference frame to the (x, y) position of the joystick handle. The constants $k1$ and $k2$ are gain values.

When using the position of the head to drive the wheelchair (Fig.4.15), the linear velocity will have a constant value v_c, while the user can decide the angular velocity according to the position of his face θ.

$$\omega_{user} = k_1 \ \theta \tag{4.30}$$

$$v_{user} = v_c \tag{4.31}$$

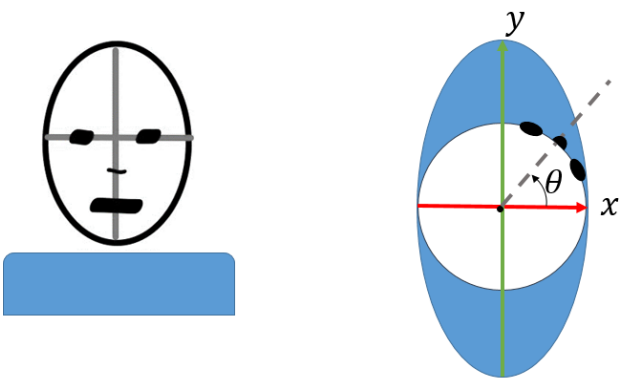


Figure 4.15: The position of the head is translated to a velocity command $(v_{user}, \omega_{user})$.

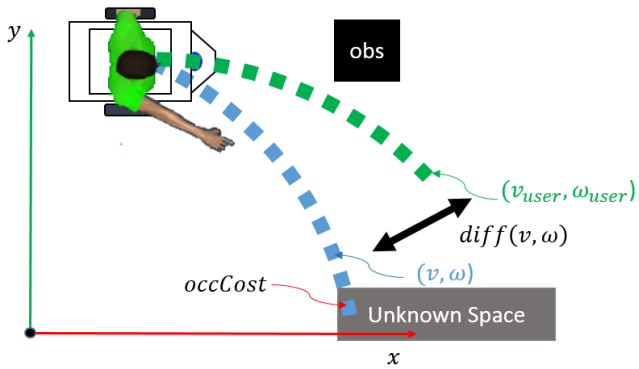


Figure 4.16: The cost term $diff(v, \omega)$ is added to the dynamic window cost function in order to take into account the input from the user.

4.3.2.2 Modified cost function

The input from the user is considered as a term in the dynamic window cost function.

$$G_s(v, \omega) = \eta_u * diff(v, \omega) + \eta_r * G_r(v, \omega) \tag{4.32}$$

The constants η_u and η_r modulate the priority given to the user input and the robot input on the final result. G_r was defined in Eq. 4.26 and the cost function $diff(v, \omega)$ measures the difference between the command given by the user and the sampled v, ω values. It is defined by the following square difference:

$$diff(v, \omega) = \sqrt{(v_{user} - v)^2 + (\omega_{user} - \omega)^2} \tag{4.33}$$

The value of the cost function (Eq. 4.32) needs to be minimized (so the semantics

are that a trajectory with a lower score is preferable to one with a higher one).

$$(v_{best}, \omega_{best}) = \arg\min_{(v,\omega)} G_s(v, \omega) \tag{4.34}$$

When the system does not receive a goal, the system will rely completely on the user to perform the planning but it can still provide some obstacle avoidance, in this case the objective function of the shared controller will be reduced to:

$$G_s(v, \omega) = \eta_u * diff(v, \omega) + \eta_r * occCost(v, \omega) \tag{4.35}$$

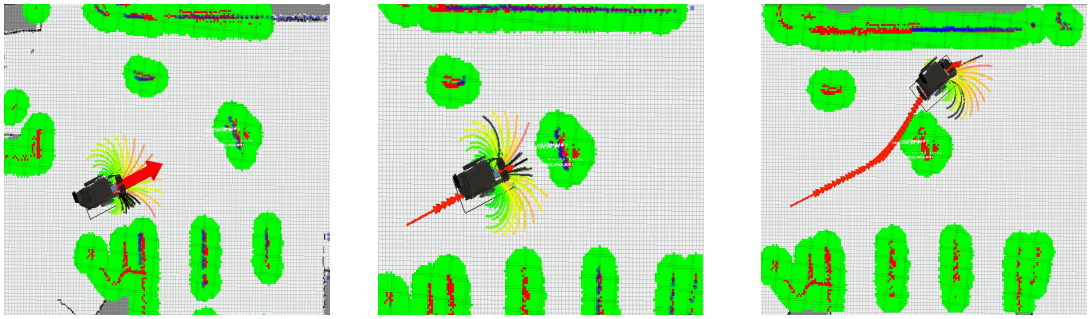
Where the term $occCost(v, \omega)$ is the maximum obstacle cost along the simulated trajectory. This value is taken from the occupancy grid used to represent obstacles around the robot as presented in section 3.2.1 (Fig. 4.16)

4.3.3 Simulation Results

In order to test the algorithm we used the same simulated scenario presented in section 4.2.2. In Fig. 4.17 the blue rectangle represents the footprint of the wheelchair, the green areas represent the obstacles detected using the on-board laser and the red arrow represents when the user sends an input command. For this simulation the only possible commands are forward, backward, left and right, given that many wheelchair interfaces give just this reduced set of possible commands. The idea is to complement the ability of the user to move with a reduced set of instructions. The simulated trajectories are marked as circular or straight lines.

The first example represents the case where the user gives a "FORWARD" command (pressing a key, or with vocal commands), so the simulated wheelchair starts moving in that direction until it detects the two green obstacles in the middle. Then as the input given by the user gets invalidated, the wheelchair starts searching for another safe trajectory (arcs in Fig. 4.17(b)). The trajectory with the best score is selected so the wheelchair is able to avoid the collision without any intervention of the user once the wheelchair avoids the obstacle it continues moving with the velocity selected by the user.

When the user wants to dock in front of a table or another piece of furniture, it is desired that the wheelchair starts decreasing its speed autonomously. This is presented in the simulation (Figure 4.18) so the user gives a straight forward command to the couches in the simulated scenario and the wheelchair moves accordingly. When the couch is detected by the laser the wheelchair starts searching a safe velocity command



(a) The user sends a "go forward" command.

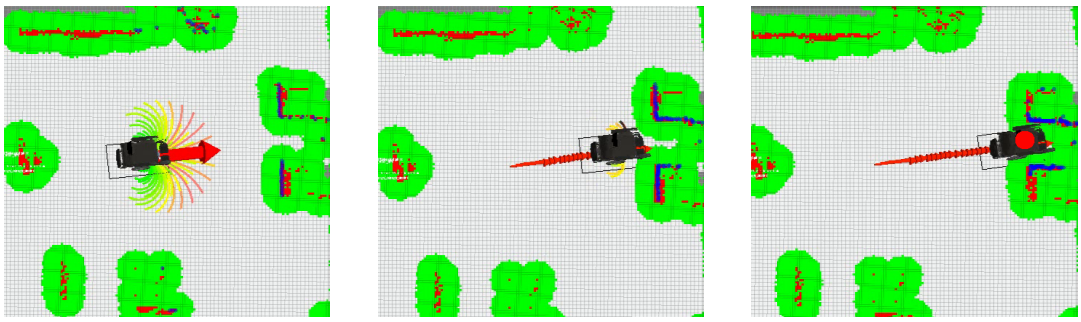
(b) the robot detects an obstacle and starts looking for a trajectory to avoid it while remaining as close as possible to the user command.

(c) Once the obstacle was avoided the wheelchair will keep moving in according to the last user command.

Figure 4.17: Avoiding an obstacle

which in this case results in reducing the speed. As the wheelchair gets closer to the couches it slows down more and more until it finally stops in front the obstacle.

In this situation the wheelchair will keep trying to move until the user actually sends a stop command (red circle in Figure 4.18(c)).



(a) The robot detects the obstacles in front looks for a way to avoid it.

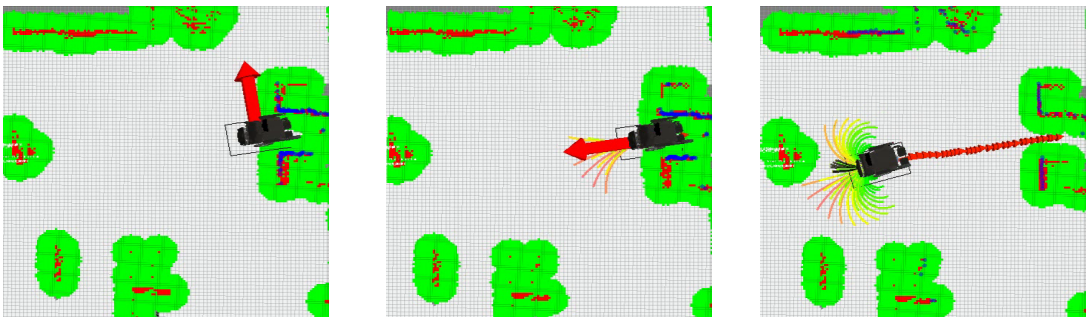
(b) Given the weight of the user command (straight), the best trajectory is found to be in the same direction but with smaller linear velocity (Docking behavior).

(c) Finally the user gives a stop command because he wants to stay there (red dot).

Figure 4.18: Docking Case.

Lets say that the user makes a mistake and gives a command to turn when the wheelchair is between two very close obstacles as shown in Fig. 4.19(a), in this case the wheelchair will try to find a safe way to turn in the selected direction but as any trajectory is found the wheelchair will just ignore the command and stay still. In this

situation; it will be responsibility of the human user to realize that the only way to move is going backwards and give the necessary order to the wheelchair. When the wheelchair receives the "BACK" command it will then move accordingly.



(a) Not safe command: when the user gives a command that cannot be executed the wheelchair does not move.

(b) Safe command: the user sends a valid command and the robot starts moving.

(c) Trajectory followed by the robot going backwards.

Figure 4.19: Thanks to the algorithm the robot will not move until it receives a valid/safe command.

This method will not correct the direction of travel unless it is really necessary. As it can be seen in the example presented in Figure 4.20(b) even if the obstacle to the right is close, the wheelchair is able to simulate that it can pass along the obstacle without changing the direction of the trip so it just keeps moving. In this situation the user is able to go from the elevator to the right door of the INRIA entry-hall with only one command.

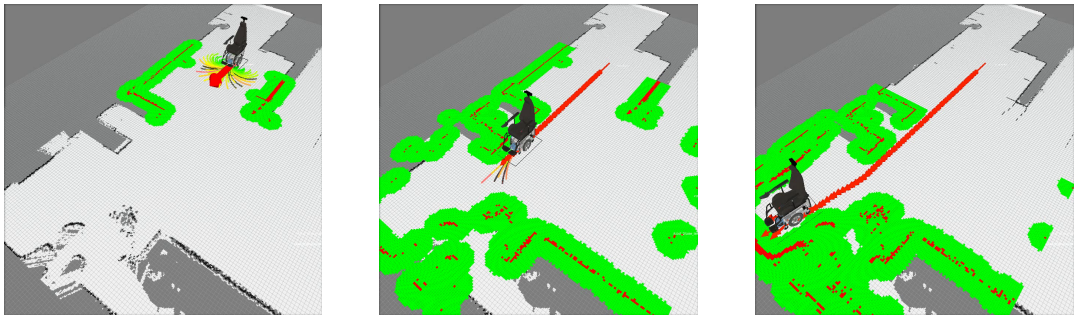
Application to Following a Person While Avoiding Obstacles

Assistance technologies focus their efforts on providing reliable solutions to help people in the everyday life. One of the key requirements of a mobility assistance system is the ability to follow a caregiver or a companion person. The environment should be modeled in such a way the robot can avoid obstacles and pursue the user at the same time.

The modified version of the dynamic window was tested in order to follow a person while avoiding obstacles. The implementation is the same as explained before but in this case the input does not come from the user of the wheelchair, but from a follower controller that tries to stay close to the target person.

The idea is that the robot should just correct the input when it simulates a possible collision in the near future (if there is any obstacle between the target and the robot).

In the example shown in Fig. 4.21 the target person is marked as a red spot, when



(a) The user sends a "forward" command

(b) The robot detects a near obstacle but the trajectory is safe so it keeps moving in the requested direction.

(c) the robot detects an obstacle which invalidates the trajectory given by the user so it searches a better option to avoid the collision

Figure 4.20: The system only corrects the command given by the user when it is necessary. In the scene, detected obstacles are represented by green zones, the command given by the user is a red arrow and the trajectory followed by the robot is the red line.

the person asks to start following him the wheelchair moves trying to keep as close as possible. However, if a possible collision is detected because there is an obstacle between the robot and the target, the robot uses the modified dynamic window cost function to compute new commands that avoid the obstacle while trying to remain close to the tracked person.

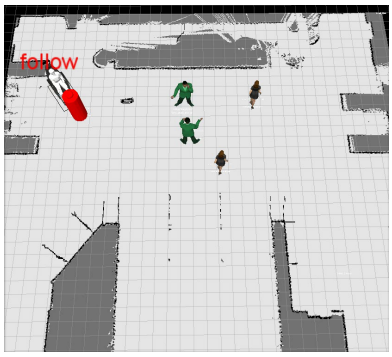
4.3.4 Discussion

In this section we presented the simulated results of our method to mix the input from the user and the wheelchair. This method was designed taking into account the use of discrete interfaces as the sip-and-puff, keyboard, BCI or even a vocal interface. This approach can improve the usability of the robotic wheelchair by itself, specially for users that cannot control the robotic wheelchair using interfaces with more degrees of freedom as a joystick. The system can even be adapted to be used in other tasks such as following a caregiver while avoiding obstacles as it was previously presented.

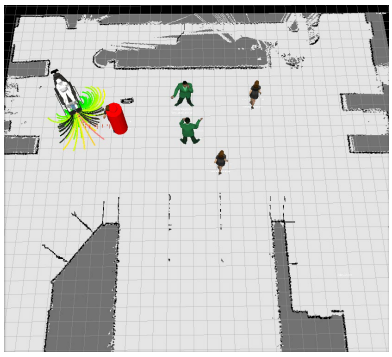
When the shared trajectory controller is used to drive the robotic wheelchair, three different behaviors can be observed.

The command given by the user is safe: If the command given by the user is safe then the wheelchair will not make any correction and it will move following the last command until it reaches an obstacle or the user gives a new command. (Fig. 4.22(a)).

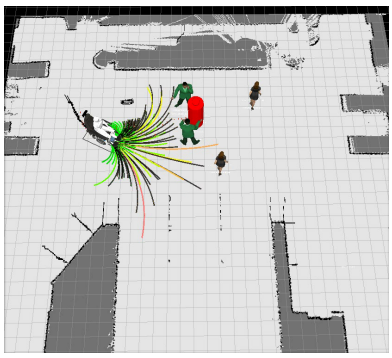
The command given by the user leads to a possible collision: When the



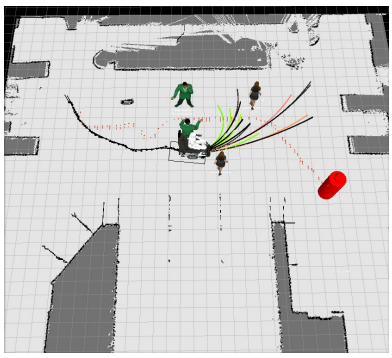
(a) The user activates the "FOLLOWING" mode.



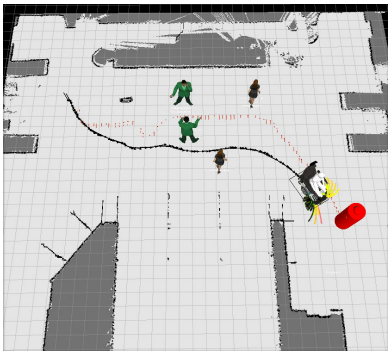
(b) The wheelchair starts following the target person (red).



(c) If an obstacle is detected between the target person and the wheelchair it will look for a path that can avoid the collision.

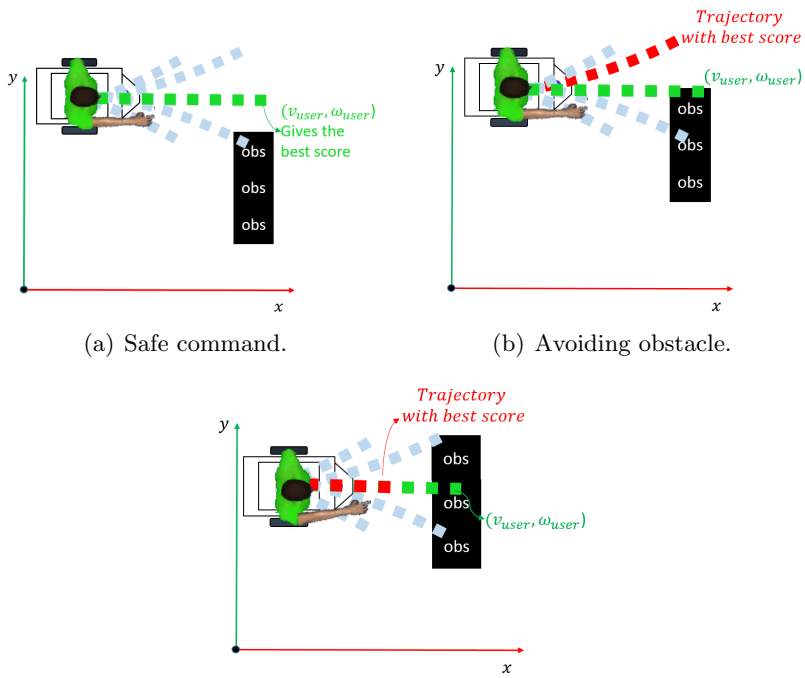


(d) when the obstacle is avoided the wheelchair keeps moving in the direction of the target.



(e) The final paths followed by the target and the wheelchair are shown in red, black respectively.

Figure 4.21: Using the shared local controller to follow a person. The target person is shown with a red marker, the generated trajectories are the shown as colored arcs (red=good score, green=bad score, black=not valid).



(a) Safe command.

(b) Avoiding obstacle.

(c) Docking.

Figure 4.22: Possible behaviors when using the shared trajectory controller without the destination inference system.

wheelchair detects a possible collision with an obstacle, two different behaviors can happen:

- **Avoiding obstacle behavior**: The wheelchair will try to avoid the obstacle by searching in the dynamic window a better trajectory that does not lead to collisions (Fig. 4.22(b)).
- **Docking behavior**: The other possible situation is that any command in the dynamic window can avoid the collision with the obstacle then the speed will start decreasing to stop in front to the obstacle (Fig. 4.22(c)).

The proposed shared trajectory controller was developed to give the user the possibility to drive the wheelchair with more freedom, however, by using only the proposed shared trajectory controller without inferring the final destination, the user is in charge of the proper planning to arrive to his desired destination.

The amount of help provided by the shared trajectory controller can be varied using the parameters η_u and η_r in Eq. 4.32. Our ongoing work is testing the results when using the probability value given by the destination inference system as modulation

term as follows:

$$G_s(v, \omega) = (1 - P) * diff(v, \omega) + P * G_r(v, \omega) \tag{4.36}$$

In this case the modulation parameter P would be the probability computed using our destination inference method for the given goal. The use of the P term allows an automatic adjusting of the amount of assistance provided by the system. Practical experiments will be presented in chapter 5 where the system will be tested also with a multi-modal interface that combines face-tracking system and voice control.

Chapter 5

Experiments

In this chapter we present the experiments and results that were done as part of this thesis. We describe the experimental wheelchair and our proposed multi-modal interface using face-pose and speech recognition. One of the major difficulties that was faced during our research was the acquisition of data coming from real sensors. This chapter also presents how we acquired the data to model the typical destinations based on observations gathered on two different experimental scenarios.

5.1 Experimental Setup

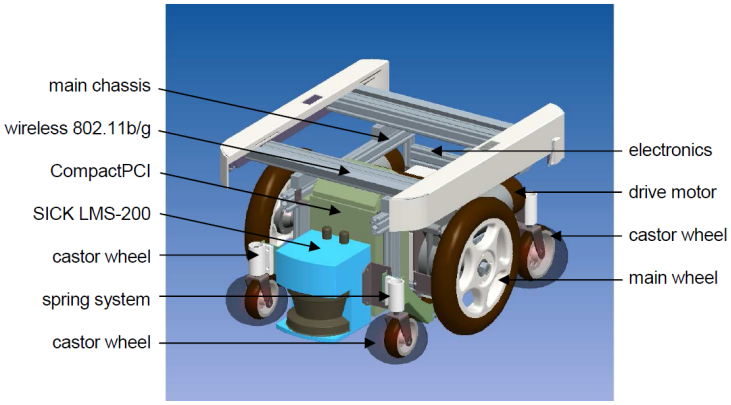
Experiments were made in two different scenarios with available tracking data of people moving around. This condition is necessary in order to extract the typical destinations and initial probability table. The locations are the INRIA entry hall and the French experimental apartment GERHOME.

5.1.1 The Experimental Robotic Wheelchair

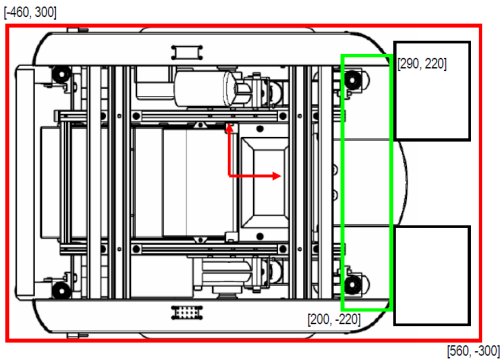
The method was evaluated in the real platform presented in Fig. 5.1. The robotic wheelchair is equipped with one laser range-finder and two 3D point cloud cameras. The laser data is used to build the map of the environment and detect obstacles in a 2D plane. One of the 3D cameras is used to detect and track people around the wheelchair which is necessary to consider the personal and interaction spaces in the costmap. The other camera is used to get the position of the user's face to drive the wheelchair.

The software architecture was developed using the middle-ware ROS (Robot Operating System). The software architecture provided by ROS allows a decoupled operation of the different components. Each component will be responsible of providing a certain

(a) The robotic wheelchair used during this thesis.



(b) The mobile base includes all the electronic components and the computer in charge of the low level control.



(c) Footprint of the robotic base, the coordinates are in millimeters with respect to the center of the base.

Figure 5.1: The INRIA Rhône-Alpes robotic wheelchair.

functionality and thus it can be wrapped in a node. To exchange data with other nodes it publishes/subscribes to topics or request/reply services.

5.1.2 INRIA-hall

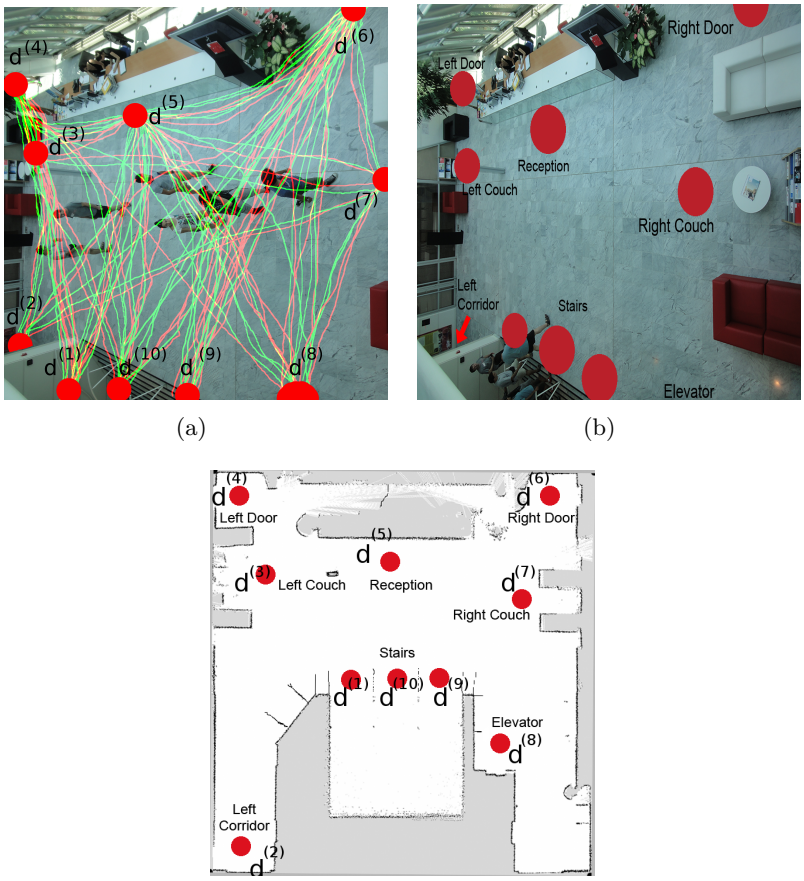


Figure 5.2: INRIA hall experimental scenario. (a) The lines in the figure represent some of the trajectories normally followed by people. Typical destinations (end of a trajectory) are marked with circles. (b) Typical destinations at INRIA Rhône-Alpes hall. (c) The map of the scenario used by the navigation system.

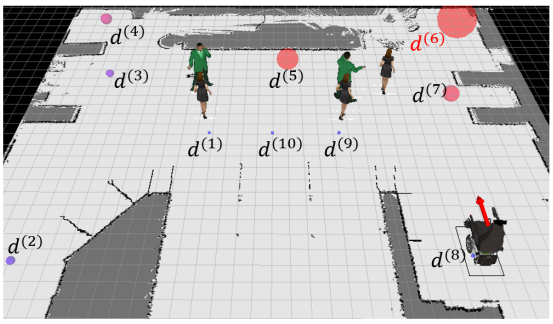
The scenario where we made most of our tests was the INRIA-Rhône Alpes Entry Hall. The map of the hall was built using laser data and it is shown in figure 5.2. White zones in the map of the environment represent free space, black are occupied regions and gray zones are not known regions.

Possible destinations were extracted from real data by tracking people in the scene

and marking those places where a trajectory starts or finishes. The tracking was performed on video-sequences from a camera located above the experimental scenario. [1]. In Fig. 5.2 the points corresponding to possible destinations are marked with red circles (reception, two doors, two couches, elevator, stairs). Each destination has an associated probability value $P(D_0|X_0)$ shown in Fig. 5.3. In practice those elements with value equal to 0 are replaced with a very small value ϵ.

	$d^{(1)}$	$d^{(2)}$	$d^{(3)}$	$d^{(4)}$	$d^{(5)}$	$d^{(6)}$	$d^{(7)}$	$d^{(8)}$	$d^{(9)}$	$d^{(10)}$
$x_0=d^{(1)}$	0.000	0.025	0.050	0.495	0.248	0.149	0.035	0.000	0.000	0.000
$x_0=d^{(2)}$	0.069	0.000	0.139	0.417	0.208	0.069	0.042	0.042	0.000	0.014
$x_0=d^{(3)}$	0.057	0.114	0.000	0.227	0.341	0.028	0.170	0.011	0.017	0.034
$x_0=d^{(4)}$	0.435	0.130	0.087	0.000	0.217	0.000	0.022	0.043	0.022	0.043
$x_0=d^{(5)}$	0.042	0.084	0.046	0.236	0.000	0.253	0.063	0.105	0.042	0.127
$x_0=d^{(6)}$	0.023	0.005	0.005	0.005	0.233	0.000	0.056	0.070	0.465	0.140
$x_0=d^{(7)}$	0.009	0.094	0.189	0.094	0.283	0.142	0.000	0.142	0.028	0.019
$x_0=d^{(8)}$	0.000	0.036	0.054	0.109	0.217	0.435	0.145	0.000	0.004	0.000
$x_0=d^{(9)}$	0.000	0.011	0.056	0.111	0.250	0.444	0.067	0.061	0.000	0.000
$x_0=d^{(10)}$	0.000	0.030	0.059	0.325	0.142	0.314	0.089	0.041	0.000	0.000

(a) Values for the initial probability distribution. In practice those those elements with value equal to 0 are replaced with a very small value ϵ.



(b) The size of the spheres represents the value of the initial probability distribution for the given starting point $X_0 = d^{(8)}$, here we can see that according to the table the destination with the highest initial probability is $d^{(6)}$ with a value of 0.435.

Figure 5.3: Initial probability distribution obtained by observing the human activity at INRIA's entry hall.

1. The video database can be found at http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1

5.1.3 Gerhome

GERHOME is an experimental apartment furnished and instrumented to evaluate solutions aiming to improve the quality of life of the elderly. This apartment is located in the CSTB (Scientific Center of Technical Building) at Sophia Antipolis in France. This experimental site looks like a typical apartment of an elderly person: $41m^2$ with an entrance, a living-room, a bedroom, a bathroom, and a kitchen.

Different sensors are distributed around the apartment to collect data. Through analysis of these data, it is possible to recognize and describe the daily activities of the elderly at home. Figure 5.4 shows a map of the apartment with the different type of sensors available such as temperature, presence, humidity, video cameras, etc.

Real elderly people, aged from 64 to 85 years old, have been observed in the experimental site during an average period of time of 4 hours. The collected data include the 36 video streams and data provided by 24 environmental sensors. [1]

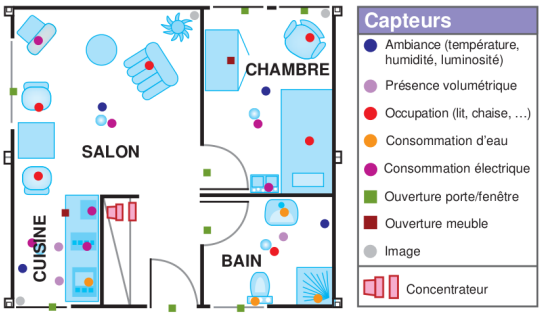


Figure 5.4: The Gerhome platform. The map and the list of sensors distributed in the building is presented.

The environmental sensors provide data when an event occurs. For instance the contact sensor determines an opening and closing events for various devices (e.g. cupboards, drawers, fridge, closets). The provided data is stored in an XML file as shown in figure 5.5 with the following format:

- TimeStamp: represents the moment when the data was provided (YYMMDD-HHMMSS.MS).
- SensorClass: represents the class of information provided by the sensor (e.g. contact, presence, electrical, pressure and water).
- SensorLocation: represents the location of the sensor (e.g. upper cupboard).
- SensorValue: the value provided by the sensor (e.g. "On" if the sensor is activated

1. Data available at www.sop.inria.fr/members/Francois.Bremond/topicsText/gerhomeProject.html.

and "Off" if the sensor is deactivated).

– SensorID: single sensor identifier which is transmitting the data.

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

```
080327-102102.620/OPENCLOSE/KITCHEN.DRAWER.LOWER/open
080327-102108.619/PRESENCE/KITCHEN.STOVE/true
080327-102112.554/OPENCLOSE/KITCHEN.DRAWER.UPPER/open
080327-102113.035/PRESENCE/KITCHEN.STOVE/false
080327-102114.186/OPENCLOSE/KITCHEN.DRAWER.LOWER/close
080327-102118.923/OPENCLOSE/KITCHEN.DRAWER.UPPER/close
080327-102121.036/OPENCLOSE/KITCHEN.REFRIGERATOR/open
080327-102137.679/OPENCLOSE/KITCHEN.REFRIGERATOR/close
```

(e) The log-file with the detected events.

Figure 5.5: An example of video sequence and the corresponding events detected by the sensors at Gerhome apartment.

By using this data we can count the number of times the user visits each of the typical destinations in the home and use this data to built the initial probability table used by the destination inference system.

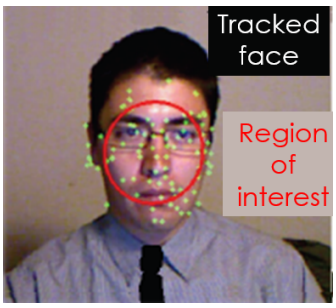
5.2 User-Robot Interfaces

Typical user interfaces for steering robots range from keyboards, over keypads to joysticks. However disabled and elderly people may have difficulties using them. Dedicated user interfaces were developed, such as sip-and-puff systems, single switches, eye tracking systems, etc., described in section 2.3. Our experimental platform was tested using typical input devices as a keyboard and a joystick, but the main research was done using the face pose recognition system and voice control, as it was considered to be a more human-like way of interaction.

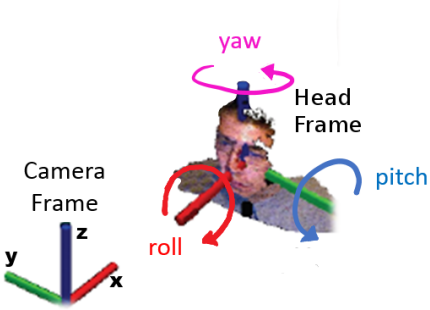
5.2.1 Face Pose Estimation System

The user can control the robotic wheelchair by using the movements of his face. A face tracking system that estimates the direction of the face from a 3D point-cloud (Fig. 5.6). The identification of the face pose is done by a random forest classifier (Fanelli et al. (2011) and Goebel (2014)) which takes as input the 3D data from the

Kinect sensor and gives the estimated position of the face. To improve the results of the 3D tracking the systems uses also a 2D RGB image to detect the face and fix a region of interest in the 3D point cloud. This system is robust enough to track the user's face even under low-light conditions, however, it tends to fail under direct sun-light because the infrared technology of the 3D camera receives high interference on such conditions.



(a) The 2D RGB image is used to detect the face and fix a region of interest in the 3D point cloud.



(b) Angles measured by the face pose estimation system.

Figure 5.6: 3D point cloud image of the user's face is used to estimate its direction.

The system is able to recognize pitch and yaw head's angles accurately, however it can give large errors for roll angle measurements. Yaw face's angle can be measured for movements in the range $[-45°, 45°]$, $pitch \in [-30°, 30°]$ and $roll \in [-6°, 6°]$ (Fig. 5.7).

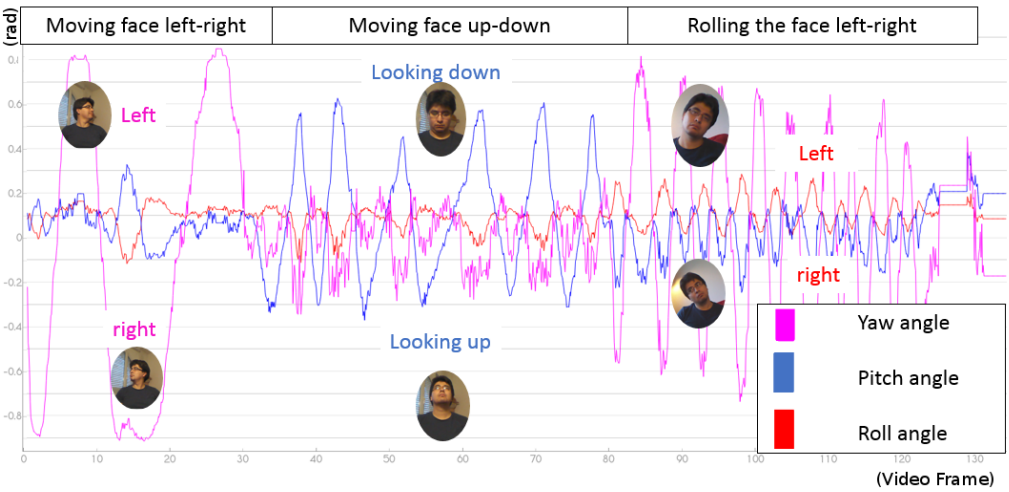
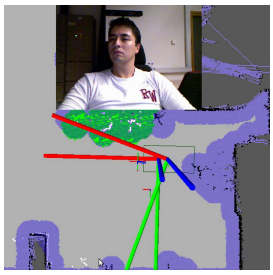
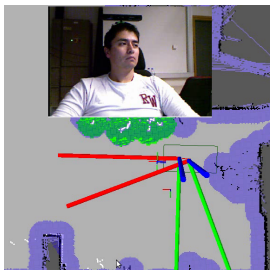


Figure 5.7: Face pose recognizer output angles. Yaw face's angle can be measured for movements in the range $[-45°, 45°]$, $pitch \in [-30°, 30°]$ and $roll \in [-6°, 6°]$

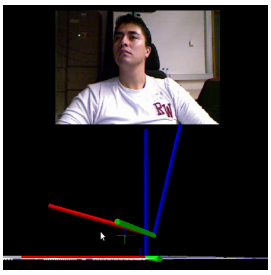
The face pose recognizer system computes the geometric transformation between the camera reference frame and the head reference frame. We assume that the camera will have a fixed position with respect to the wheelchair's reference frame, therefore, we can compute the relative position between the user's head and the wheelchair's reference frame that is used by the navigation system. In figure 5.8 it is shown the resulting head reference frame with respect to the wheelchair's reference frame as it is computed by the system. The yaw angle between both frames is mapped to control the angular velocity of the wheelchair while the pitch can be used to give other orders as brake or go. Those two angles will respect the following convention. If the user is looking to the left the yaw angle will be negative and if the user is looking down the pitch angle will be negative.



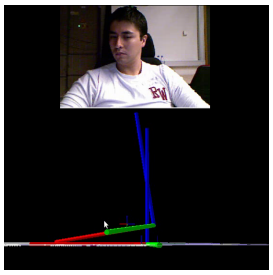
(a) The user is looking to the left, it results in a negative yaw angle.



(b) The user is looking to the right, it results in a positive yaw angle.



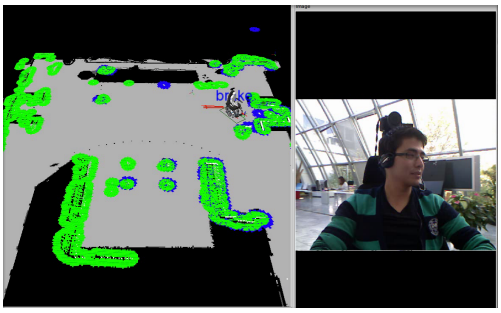
(c) The user is looking up, it results in a positive pitch angle.



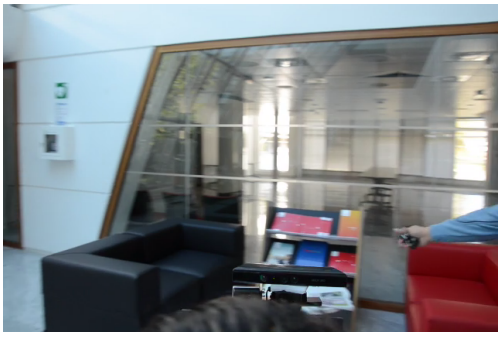
(d) The user is looking down, it results in a negative pitch angle.

Figure 5.8: The face pose recognizer system publishes a transformation between the camera reference frame and the head reference frame. Assuming that the camera will have a fixed position with respect to the wheelchair's reference frame, it can be computed the transformation between the head pose and the wheelchair's pose.

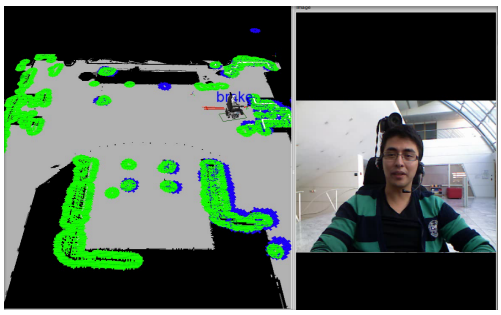
(a) (Frontal view of the wheelchair) Somebody asks the user to look to to the left.



(b) (User's face and map used by the wheelchair) The user moves his face to look what he was asked to.



(c) (Frontal view of the wheelchair) The wheelchair turns to be in a more comfortable position for the user.



(d) (User's face and map used by the wheelchair) The wheelchair stays aligned with the driver's face.

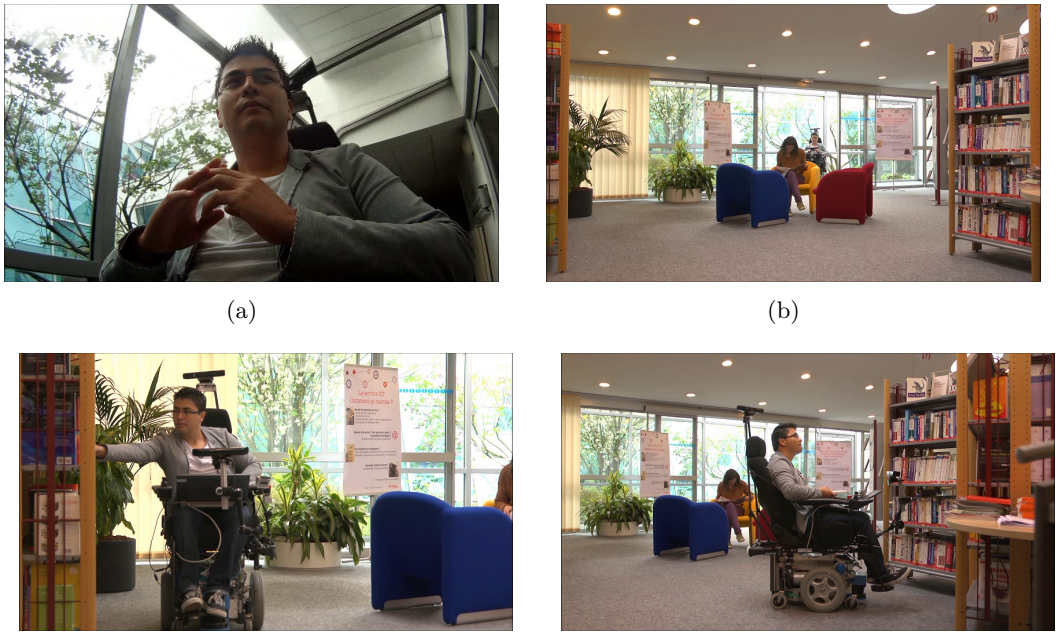
Figure 5.9: The user receives some indications to look around while the wheelchair is in turning mode. The wheelchair turns around according to the user's face position. It is a comfortable way of acting as long as it does not hit obstacles.

5.2.1.1 Turning in place

In the example situation 5.9 the user is driving the wheelchair using the face-pose interface. This allows the user to use his hands for other tasks as carrying a book or opening a door. In this case the user is getting some information from another person indicating to look towards some place while the wheelchair adjusts its direction accordingly.

In another example situation the user drives in an unknown environment (the library at INRIA), as there are no previously recognized destinations, the wheelchair just performs obstacle avoidance using the shared trajectory controller (without path planning). In this test the user drives the wheelchair by showing his desired travel direction with the face. The wheelchair performs obstacle avoidance using the presented shared

local controller. Stop and Go commands are performed through the speech recognition system Fig. 5.10.



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

Figure 5.10: In this test the user drives the wheelchair by showing his desired travel direction with the face. The wheelchair performs obstacle avoidance using the presented shared local controller. Stop and Go commands are performed through the speech recognition system. The full video can be accessed at http://youtu.be/O3IPISn7waw.

5.2.2 Voice Recognition

The voice interface is used to fulfill some lack in functionality that the face pose interface can not supply. We tested two different voice recognition systems; Pocketsphinx from Carnegie Mellon University and the Google speech recognition API.

– **Pocketsphinx**: This voice recognition system was developed at Carnegie Mellon University and described in (Huggins-Daines et al., 2006). The system allows to specify a set of syntactic rules (or grammar) which constrains the ordering of words within a sentence. This grammar enhances the speech recognition quality by reducing the hypothesis space. In general a small vocabulary makes speech recognition more accurate, therefore a dictionary that focuses in a very small fixed set of tasks was considered (go, brake, faster, slower, autonomous, manual). The

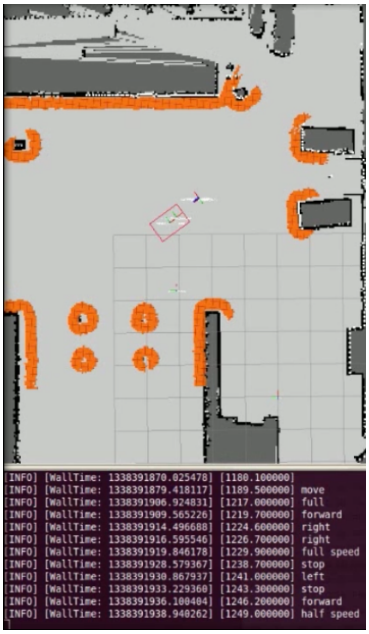


Figure 5.11: Driving the wheelchair using vocal commands, we see the map showing the wheelchair and the list of vocal commands recognized by the speech recognizer. The video corresponding to this image sequence can be watched at http://youtu.be/w5VMVrXhzfg.

code and examples to use can be found at (Carnegie-Mellon-University, 2014) web-page.

– **Google Speech Recognition API**: This is a service delivered from the provider cloud (Schalkwyk et al., 2010). This voice recognition system is capable of handling any query that the Google search engine can handle. the voice of the person has to be recorded in an audio file, then that audio file has to be sent to the Google Speech Recognition Service and it will return a string containing, in plain text, the spoken words contained in the audio file. One drawback is that it can be slow and it is dependent on the availability of an Internet connection. However, one of the main advantages of this service is that the available vocabulary queries is very large and it is available in different languages.

The user can control the linear velocity of the wheelchair by giving orders as faster, slower, full speed or stop. He can change the direction with the back and forward commands and he can ask the wheelchair to start moving with the move and go orders. An optional modality implemented in our wheelchair allows the user to use the left and right orders to control the angular velocity when the face pose tracker is not

present. In the figure 5.12 we show an example of the possible vocal commands to drive the wheelchair. It is also shown the map and detected obstacles in the environment (orange)) as well as the footprint of the wheelchair (red rectangle). The wheelchair will move according to the given vocal command as long as it does not hit any obstacle.

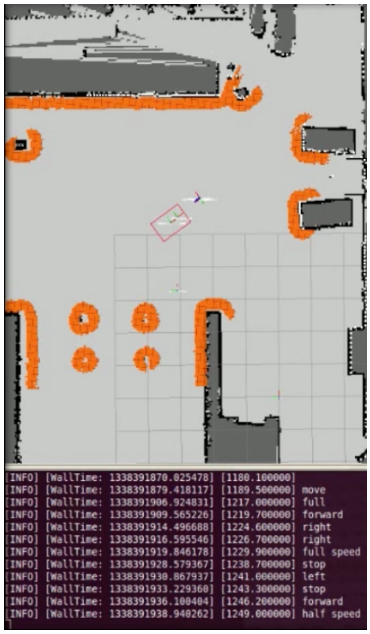


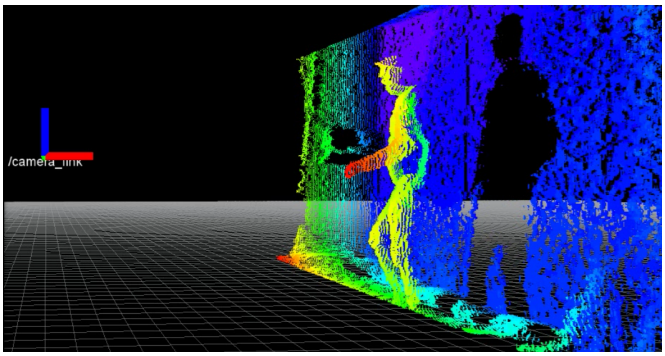
Figure 5.12: Driving the wheelchair using vocal commands, we see the map showing the wheelchair and the list of vocal commands recognized by the speech recognizer. The video corresponding to this image sequence can be watched at http://youtu.be/w5VMVrXhzfg.

5.2.3 3D Human Tracker

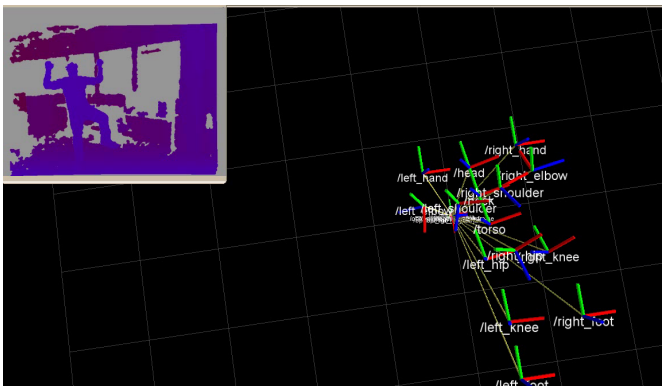
The multiplicity in shape, as well as the complexity of configuration the human body can achieve requires complex sensors to be tracked accurately. Color cameras are amongst the most effective sensors, even if the information are limited to the image plane (McKenna et al., 2000). Lately the development of technology and availability of comparatively affordable RGB-D sensors, adequate to perceive 3D structures, opened the possibility to extend the range image-based recognition (Shotton et al., 2011). It is then reasonable to choose such sensors to capture people positions (Spinello and Arras, 2011).

The availability of a software library for user skeleton detection simplifies the problem of detecting human shapes using the RGB-D data from the Kinect sensor. For

the version 1.5 of OpenNI library (used to track the person), the user had to execute a particular calibration routine (called ψ pose) to be detected. This constricted the usability of the software for multiple people perception. In version 1.5, however, the users can be perceived using a user generator that does not require any calibration at all. The output of this component is a set of people position estimation. In figure 5.13 we can observe the 3D point cloud used by the OpenNI skeleton tracker to estimate the position of the tracked user. The resulting skeleton is a list of links defining the parts of the user body (arms, legs, torso, etc.). The position of humans detected is given in the camera's reference frame.



(a) The 3D Pointcloud



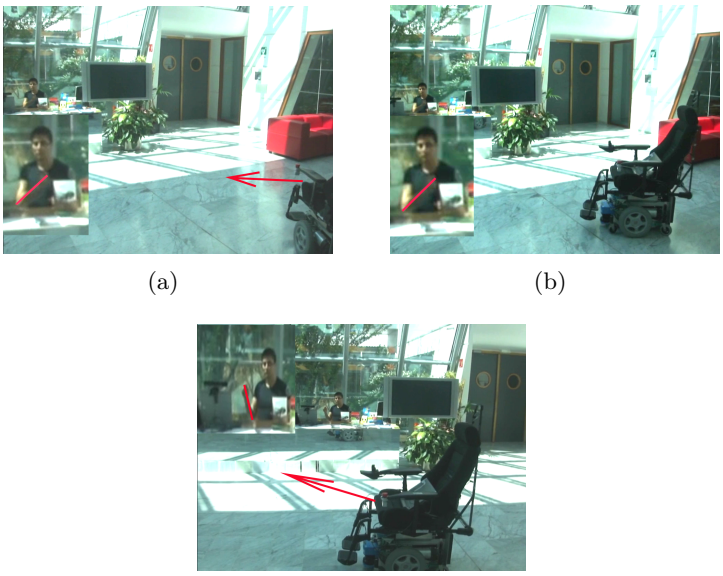
(b) The depth image and the resulting skeleton of the tracked persons.

Figure 5.13: The OpenNI skeleton tracker uses a 3D point cloud to estimate the position (referent to the camera) of the tracked user.

5.2.3.1 Arm Gesture Recognition

Arm gesture was another interface that was contemplated during this study. It consisted in detecting th position of the user hands to drive the wheelchair (figure 5.14). Remotely driving the wheelchair by using the hands could be a useful way to interact with the robot to ask it to get closer for example when the user wants to get off from bed.

The arm detection is performed using the kinect skeleton tracker of the OpenNI library described in section 5.2.3.



(a) (b)

(c)

Figure 5.14: The skeleton tracker can be used to drive the wheelchair using gestures. In the scene our experimental robotic wheelchair is driven using hand-gestures. The video corresponding to this image sequence can be watched at http://youtu.be/w5VMVrXhzfg.

5.2.4 Keyboard

During experiments in simulation we used a keyboard interface to control the wheelchair. By using 5 keys we can send a front, back, left, right and stop signals used by the user to drive the wheelchair. To emulate the use of a vocal interface we use some other keys to give orders such as "GO", "JOIN", "FASTER", and "SLOWER". This interface is used to test the performance of the methods when using a low-bandwidth

discrete interface given that a future application could be centered on the use of a brain computer interface which normally gives the same quantity and type of orders.

5.3 Multimodal Control using Face Pose and Speech Recognition

The wheelchair can be controlled in **semi-autonomous** mode employing the user intention prediction module or in **manual** mode. In manual mode the user controls the wheelchair's angular speed by moving his face. The linear speed is controlled with vocal commands as explained in section 5.2.2.

In semi-autonomous mode; the user shows the direction to his desired destination by looking towards it. The user intention module computes the destination with the highest posterior probability, depicted in Fig. 5.17 (a) as the biggest circle. The navigation module receives the map of the environment, the list of humans present in the scene and the currently estimated goal to compute the necessary trajectory to the goal as it is shown there.

When moving the user does not have to worry about the necessary planning to avoid obstacles because the autonomous navigation system is in charge of that, however, he can stop or start the wheelchair by using the speech recognition system.

The main function of the speech interface is to switch between manual and semi-autonomous modes by saying the **"manual"** and **"autonomous"** vocal commands.

- Autonomous: The wheelchair computes the most likely intended destination of the user and navigates autonomously towards it.
- Manual: The linear speed of the wheelchair is regulated using vocal commands while the angular speed is controlled by moving the face. No obstacle avoidance or path planning is provided by the wheelchair.

The speech recognition system is used in combination with the face pose estimator to provide different operation modes as detailed in Table 5.1

5.3.1 Test 1: Driving with Destination Inference Method vs Driving without Assistance

To evaluate the performance of the method, four different persons were asked to drive the wheelchair in **manual** and **autonomous** modes. The wheelchair could be controlled in autonomous mode where the desired destination is inferred by the system and then the wheelchair navigates autonomously to reach it, or in manual mode where

Table 5.1: Vocal commands and associated driving behaviors in manual and autonomous modes.

Autonomous Mode

go + face direction	Computes the most probable destination in that direction and navigates autonomously against it.
one brake + face direction	Stops the wheelchair and keeps turning in the same position.
brake (2 times)	Stops completely the wheelchair.

Manual Mode

go + face direction	Moves the wheelchair with a constant linear speed; the angular speed is proportional to the face direction.
faster	Increases the linear speed
slower	Decreases the linear speed
one brake + face direction	Stops the wheelchair and keeps turning in the same position.
brake (2 times)	Stops completely the wheelchair.

the user drives the wheelchair without assistance. To switch between these two modes, we use the voice recognition system.

The user is asked to start the movement from one of the labeled destinations shown in Fig. 5.2(a) and visit a list of targets e.g (go to the reception, then go to the right door and then come back).

Figure 5.15: Using face pose to estimate the user's desired destination. (Video at http://goo.gl/VvtezQ).

Every command and computed intent was stored at each time step. This allows to compare the output of the estimation with the actual intention of the human. Second, when driving towards the destination, global measures such as the total time needed, the total distance traveled and the followed trajectory were logged to quantify the performance.

In Fig. 5.16 some of the resulting trajectories are presented, the first thing that can be noticed is the improved performance accomplished when using the user intention estimation algorithm. This avoids abrupt modifications in the trajectory due to involuntary changes in the direction of the face produced when the user is exploring the surroundings. The time when executing the task was also improved as shown in the following table. The most important fact to be noticed is that collisions were completely avoided when using the user intention algorithm, on the other hand, in manual mode some collisions were produced.

5.4 Destination Inference with Dynamic Destinations

In the example situation (Fig. 5.15), the user is looking to his left so that it is more probable that he is aiming to go to the coaches located in that direction or to join the group. The direction of his face is computed as explained in section 5.2.1.

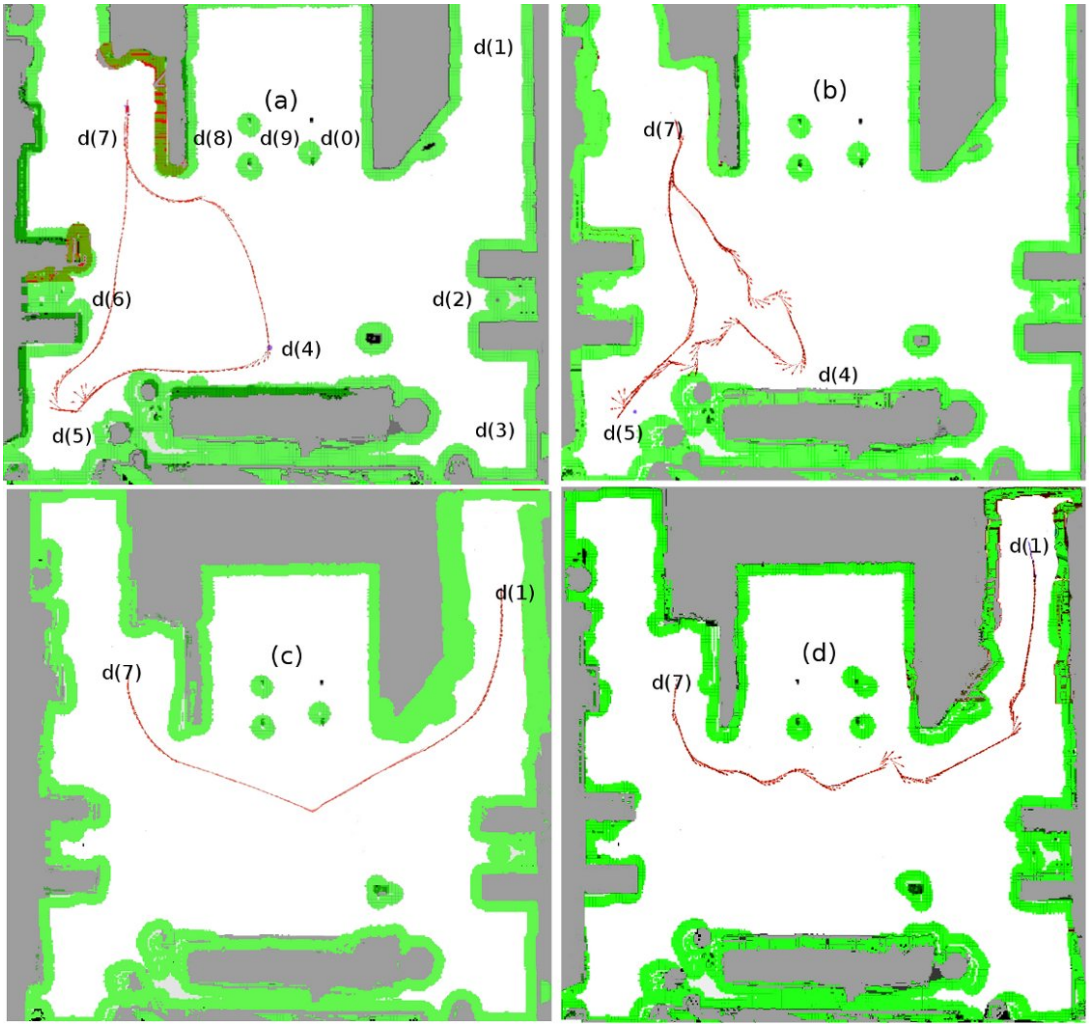


Figure 5.16: Some samples of the logged trajectories are presented. (a) and (c) show the results when using the assistance of the user intention estimation system. (b) and (d) were achieved by driving the wheelchair using the face without any assistance. Here we can observe the oscillations in the trajectory due to involuntary changes in the face direction produced when observing the surroundings.

Typical destinations are marked in the map and whenever a new command is read from the face control, the user estimation module computes the one with the highest posterior probability. The navigation module receives the map of the environment, the currently computed goal and the list of people present in the scene to compute the necessary trajectory to the goal.

In the example there are two persons talking between them and they are positioned

Table 5.2: Performance metrics when driving the wheelchair with and without assistance

Semi-autonomous mode	Time [s]	Distance [m]	Num. of collisions
User 1	32.1	24.4	0
User 2	34.5	25.8	0
User 3	35.3	26.2	0
User 4	32.5	25.1	0
Manual Mode	Time [s]	Distance [m]	Num. of collisions
User 1	54.0	28.2	0
User 2	68.3	29.3	0
User 3	75.4	29.4	1
User 4	63.2	27.0	1

in the middle of the path between the wheelchair and the current estimated goal.

Even if the user is pointing to the goal located in the other side of the two persons he does not have to worry about the necessary planning and commands to avoid interrupting the conversation because the autonomous navigation system is in charge of that.

Data was gathered on two distinct levels. First, every command sent by interfaces was logged, as well as the intent of the subject at that time. This allows to compare the output of the classifier with the actual intention of the human on the individual command level. Second, when driving towards the goal position, global measures such as the total time needed, the total distance traveled and the followed trajectory were logged to quantify the task performance.

In case of an error, for example when the inferred destination can not be reached because it is occupied or there is not enough evidence to decide which destination is better, the system will ask for help and wait for the user to give a new command as shown in Fig. 5.19.

In the case of an ambiguous destination, the wheelchair will present the two goals with higher probability to the user who will disambiguate by giving a new command. The direction of the new command is used and the probabilities are updated. If the new destination is not occluded then, the wheelchair will start moving.

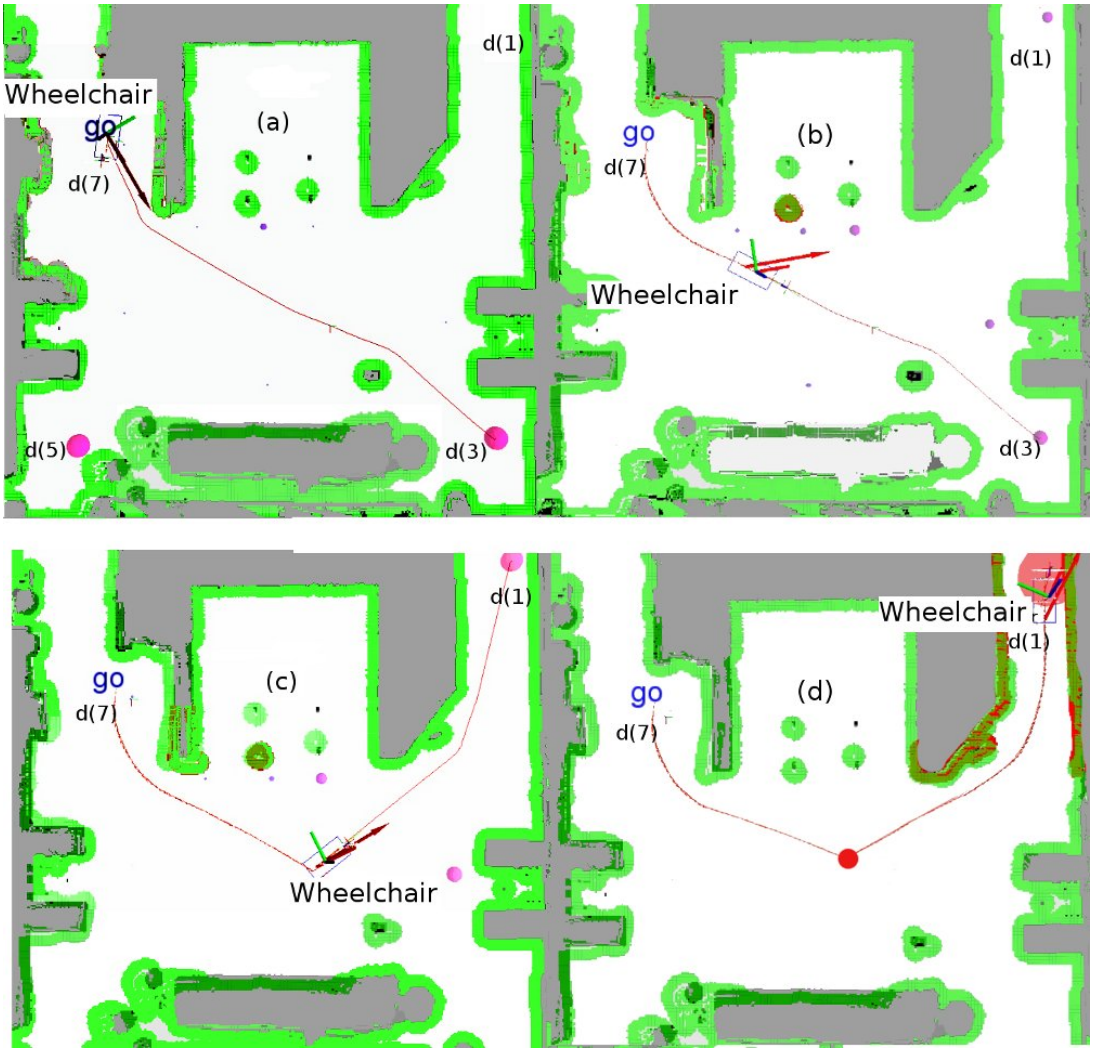


Figure 5.17: Experimental Evaluation, the user is asked to go from $d^{(7)}$ to $d^{(1)}$ using the user intentions system. (a) As the destination with the highest prior probability from $d^{(7)}$ is $d^{(5)}$ in the beginning of the test the results are biased to that side that is why $d(3)$ is initially chosen as the most likely goal. (b) As the user keeps looking to the left all the destinations in that direction become more likely. (c) When the wheelchair has enough evidence, it changes the desired destination to $d^{(1)}$. (d) The user arrives to his desired destination, the spot in the middle of the trajectory marks the place were the change of destination was produced.

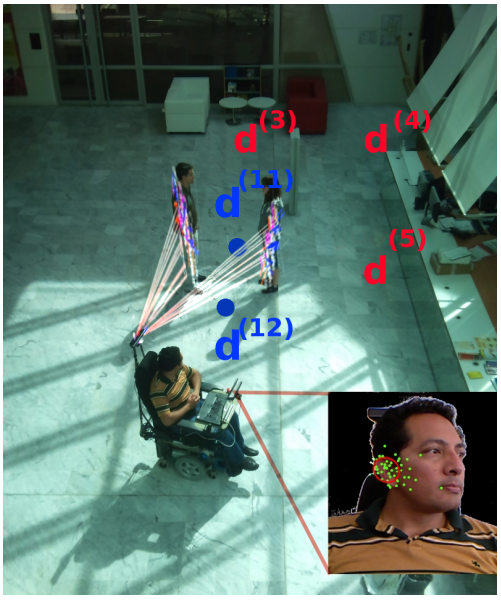
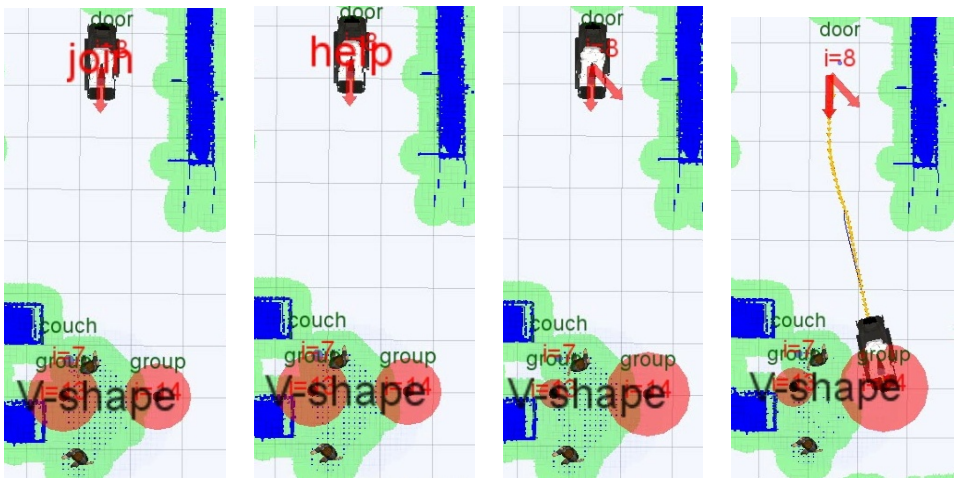


Figure 5.18: Example of an experimental scenario where some typical destinations and meeting points have been detected. In the intended application, the user points in the direction of his desired destination and use a vocal command to order the wheelchair to move.



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

Figure 5.19: Error handling. When the system detects an error in the inferred destination (it is blocked or there is not enough evidence to decide) (a) it asks for help to the user (b). The user gives a new command (c) and if the new destination is valid it will move towards it.

5.5 Caregiver Following

The wheelchair was adapted with the capability of following a caregiver or companion person. To track a person we used the OpenNI skeleton tracker described in section 5.2.3. The movement of the wheelchair is computed by a PID controller that takes as input the distance from the wheelchair to the user and outputs a velocity control that tries to reduce this distance. This command is then checked using our shared control presented in section 4.3.2 in order to perform obstacle avoidance.

In figure 5.20 we show a scene of the robotic wheelchair following the user. This is an important characteristic that a robotic wheelchair should pursuit to ease the job not only for the user but also for the medical staff an his family.

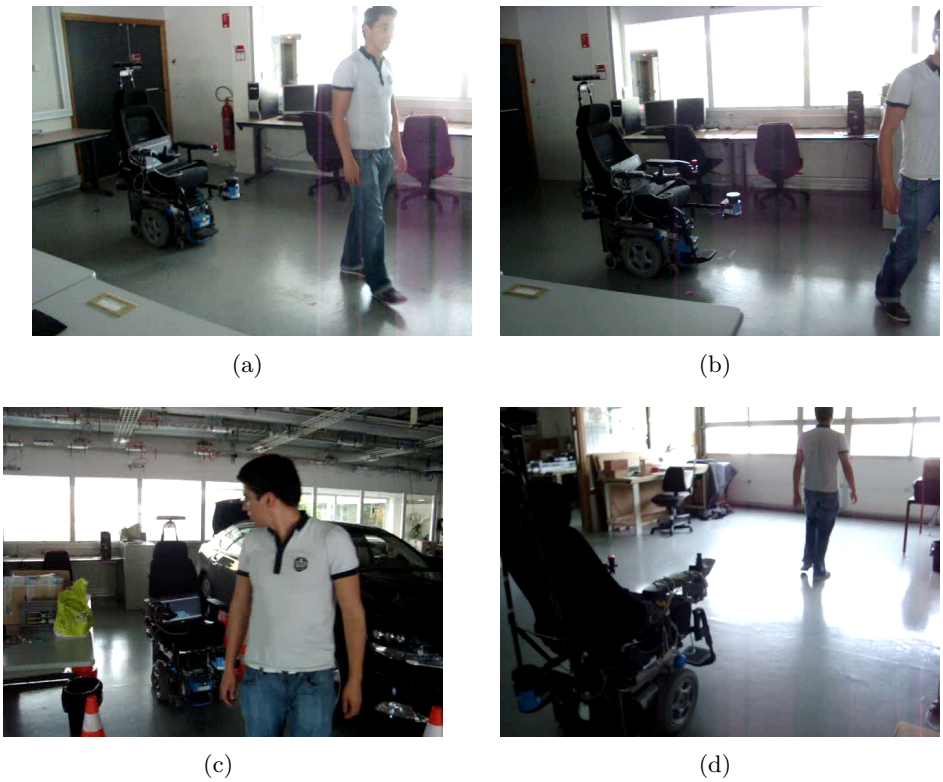


Figure 5.20: A scene of the robotic wheelchair following the user. This is an important characteristic that a robotic wheelchair should pursuit to ease the job not only for the user but also for the medical staff an his family.

5.6 Discussion

The new application of the wheelchair system with more natural and easy-to-use human machine interfaces was one of the main contributions. Human aware navigation, path planning and obstacle avoidance are performed by the robotic wheelchair while the user is just concerned with "looking where he wants to go".

This system was improved by using the user intention estimation algorithm, because it avoids abrupt modifications in the trajectory due to involuntary face movements produced when the user is exploring the surroundings. The time when executing the task was also improved because the user does not have to waste time in increasing/decreasing the linear speed with a vocal command.

The most important fact to be noticed is that collisions were completely avoided when using the user intention algorithm, on the other hand, in manual mode some collisions were produced.

A more sophisticated/accurate user intention algorithm combining maybe machine learning techniques is desired in order to add the capability to be autonomously adapted to the user disability.

Chapter 6

Conclusions and perspectives

The principal objectives of this thesis were to improve the compliance, usability of a robotic wheelchair by estimating the user intention and to improve the comfort for people around the wheelchair by respecting typical social conventions.

The modular architecture of the wheelchair navigation system granted the capability to use functional modules provided by the open source community as the A* path planner, the AMCL self-localization, etc. This design promoted the improvement of several already-existent methods to the particular needs stated as design requirements.

The destination inference algorithm was used to translate simple input commands (travel direction) into high level orders (the desired destination). This reduces the necessary time to learn how to drive the wheelchair and lessen the user involvement while driving.

The algorithm was tested in experimental environments, however, it will be necessary to try it under real-life situations and real users in order to prove conclusively the benefits of the system. To work in a non-supervised environment, the user's intention algorithm must be extended and perform an on-line learning to add new important places and user's habits updating the initial probability model described.

One of the main issues of the destination inference method is the decrease of freedom in the number of possible movements that the user can perform. This problem can be solved while still keeping the obstacle avoidance advantages when using the proposed shared velocity controller alone and drive the wheelchair in a shared control mode. In that mode the user is in charge to indicate the travel direction to the wheelchair while it will compute the necessary speed commands to avoid obstacles.

6.1 Meeting Points Computing

The estimation of meeting points is an important part of our system because it is used to infer the desired destination of the wheelchair user. The geometrical model to approach humans gave us some first insights in the understanding of people's reaction when the robotic wheelchair approaches groups of people, however it should be extended to consider the dynamic evolution of the group once the robot joins it.

6.2 Multimodal Interface

We proposed the use of a multi-modal interface using speech recognition and face movements. The destination inference module was used to avoid abrupt changes in the trajectory due to involuntary face movements. It also can solve the problem of driving the wheelchair against obstacles due to the natural human behavior of looking towards risky zones.

6.3 Respect of Social Conventions

Autonomous navigation techniques for robots moving among humans must be aware of social aspects of human interactions. The work presented in this thesis addressed that problem by using methods which take into account social conventions to improve the wheelchair's navigation system. Although the method used as part of this thesis is helpful to prevent possible embarrassing and even dangerous situations; the employed models of social spaces are static. It will be important to integrate methods that consider the movements of the persons and nearby objects to estimate the interactions in a dynamic way.

The social defined regions were successfully implemented as a cost-grid that can be employed and tested by the interested reader using the open source ROS navigation stack to reproduce the experiments of this thesis.

6.4 Final Remarks

As a final remark, smart wheelchairs are excellent test benches to study robot control architectures, human-robot interaction, shared control and novel access devices. However, there are still several barriers to overcome before smart wheelchairs can become widely used by real patients. It is necessary to improve the sensing system to

have an inexpensive device able to detect obstacles and drop-offs over a wide range of operating conditions and surface materials. Up to the moment smart wheelchairs are designed to navigate in modified indoor environments where access to drop-offs is restricted.

Appendix A

Appendix

A.1 Introductory Concepts of Probability

Due to the lack of information or noise in the physical world every mathematical model of a real system is in some way incomplete. This incompleteness in the mathematical models produce that the outcome is up to some level uncertain. Probability is one of the most important mathematical tools used in robotics. Probabilistic reasoning proposes a way to deal with the inherent uncertainty of the world.

A.1.1 Logic, Probability and Reasoning under Uncertainty

Traditional logic reasoning can be classified in deductive and plausible reasoning.

Definition 14 ***Deductive reasoning*** *allows reasoning under the logical consequence of events. Logical propositions are either true or false, implying true or false consequences Jaynes (2003).*

Definition 15 *In* ***plausible reasoning****, each logical proposition has an attached degree of* plausibility*, with bounds* not plausible*, meaning the proposition is false, and* completely plausible *indicating the proposition is true.*

When plausibility values are near limits (not plausible or completely plausible), plausible reasoning is similar to deductive reasoning. Jaynes (2003) proposed applying probabilities as plausibility values. This method supplies the rigid theoretical formalization of Probability Theory.

Definition 16 *A **logical proposition** is an statement with a precise meaning and specific logical type (true or false). Logical propositions are denoted by small letters, and they can be combined, applying logical operators, to obtain new propositions.*

As an example if the proposition a means "The robot is at the office", and proposition b is "The robot is going to the kitchen", a new proposition $a \wedge b$ (the *conjunction* of a and b) would have the meaning "The robot is at the office and is going to the kitchen". It is also possible to express the *disjunction* of propositions a and b, which results in a new proposition $a \vee b$ meaning "The robot is at the office or is going to the kitchen". The *negation* of a proposition, as in $\neg a$, is another proposition with opposite logical type "The robot is not at the office".

Propositions can also mean that a variable has an associated value, as in proposition v meaning "The current robot speed is $3.0m/s$". We can say that, if the robot speed is indicated by a variable V, v means $[V = 3.0]$. This kind of proposition is used extensively when programming a robot and leads to the definition of a discrete variable.

Definition 17 *A **discrete variable** V is then defined as a set of n logical propositions, one for each possible value in the domain $D_v = v_1, v_2, \ldots v_n$. These logical propositions are then mutually exclusive and exhaustive.*

$$v_i \wedge v_j = false, \textbf{for} i \neq j; \tag{A.1}$$

$$v_i \vee v_j \vee \ldots v_n = true \tag{A.2}$$

Where, n denotes the cardinality of a variable.

A variable can only be assigned values in its domain, and only one value at a time. Variables are usually denoted by names beginning with capital letters, while the values in its domains are denoted by small letters. As each possible value in the domain is associated with a logical proposition, there is no ambiguity in this notation: value v_j is equivalent to the proposition $V = v_j$.

A.1.2 Probability of a Proposition

Sometimes, we do not know if a particular proposition x is true or not, but we still have reasons (eg prior knowledge or evidence) that make us believe that one of those values is more likely than the other. We express this using the notation $P(x|\pi)$ which is read "the conditional probability of x given our former knowledge π". Since -under

the subjective interpretation- probabilities are always estimated on the basis of former knowledge, it does not make sense to simply write $P(x)$ or probability of x. That said, we will systematically omit the specification of former knowledge π, but it should be noted that this is only a notational shortcut. Let X denote a random variable and x denote a specific value that X might take on. If the space of all values that X can take on is discrete, we write

$$p(X = x) \tag{A.3}$$

A.1.3 Probability Theorems

They are mathematical identities used to manipulate proposition probabilities in order to perform inference.

A.1.3.1 The Product Rule

The product rule relates the probability of the logical product $x \wedge y$ to the individual probabilities of x and y :

$$P(x, y) = P(x)P(x|y) = P(x)P(x|y) \tag{A.4}$$

A.1.3.2 The Normalization Rule

The normalization rule expresses the relationship between the probability of a proposition and that of its negation.

$$P(x) + P(\neg x) = 1 \tag{A.5}$$

A.1.3.3 The Sum Rule

Just as in classical logic it is possible to construct every possible logical expression using only negation and conjunction, two identities -the product and the normalization rules- are used to derive all possible probabilistic computations. One of the rules which is possible to obtain from A.4 and A.5 is the sum rule:

$$P(x + y|z) = P(x|z) + P(y|z) - P(x, y|z) \tag{A.6}$$

A.1.4 Theorem of Total Probability

This theorem follows from the definition of conditional probability and the axioms of probability measures, the discrete case is referred as:

$$p(x) = \sum_y p(x|y)p(y) \tag{A.7}$$

A.1.5 The Bayes Rule

Equally important is Bayes rule, which relates conditionals of the type $p(x|y)$ to their 'inverse,' $p(y|x)$. The rule, as stated here, requires $p(y) > 0$:

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\sum_x p(y|x)p(x)} \tag{A.8}$$

Bayes rule plays a predominant role in probabilistic robotics. If x is a quantity that we would like to infer from y, the probability $p(x)$ will be referred to as *prior probability distribution*, and y is called the data (e.g., a sensor measurement). The distribution $p(x)$ summarizes the knowledge we have regarding X prior to incorporating the data y. The probability $p(x|y)$ is called the posterior probability distribution over X. As (Eq A.8) suggests, Bayes rule provides a convenient way to compute a posterior $p(x|y)$ using the inverse conditional probability $p(y|x)$ along with the prior probability $p(x)$. In other words, if we are interested in inferring a quantity x from sensor data y, Bayes rule allows us to do so through the inverse probability, which specifies the probability of data y assuming that x was the case.

An important observation is that the denominator of Bayes rule, $p(y)$, does not depend on x. Thus, the factor $p(y)^{-1}$ in (Eq. A.8) will be the same for any value x in the posterior $p(x|y)$. For this reason, $p(y)^{-1}$ is often written as a normalizer variable, and generically denoted η:

$$p(x|y) = \eta \; p(y|x) \; p(x) \tag{A.9}$$

A.1.6 Conditioning on Other Variables

In many cases it is perfectly fine to condition any of the rules of probability theory on arbitrary other random variables, such as the variable Z. For example, conditioning

Bayes rule on $Z = z$ gives us:

$$p(x|y,z) = \frac{p(y|x,z)\ p(x|z)}{p(y|z)} \tag{A.10}$$

Similarly, we can condition the rule for combining probabilities of independent random variables on other variables z:

$$p(x,y|z) = p(x|z)\ p(y|z) \tag{A.11}$$

Such a relation is known as conditional independence. Conditional independence plays an important role in probabilistic robotics. It applies whenever a variable y carries no information about a variable x if another variable value z is known. Conditional independence does not imply (absolute) independence. The converse is also in general untrue: absolute independence does not imply conditional independence. In special cases, however, conditional and absolute independence may coincide.

Background knowledge about the meaning and scope of a proposition also called preliminary knowledge, is usually expressed by small Greek letters. They condition the assigned probability: $P(a|\pi)$ stands for the plausibility that proposition a is true, knowing the preliminary knowledge summarized by π. The plausibility that the variable X has a value x_i is then expressed using a probability value $P([X = x_i]|\pi)$ or, identically, $P(xi|\pi)$. It is also possible to write $P(X|\pi)$, as the plausibility of variable X being equal to each value in the domain D_X : in other words, the probability distribution over values for variable X.

A.1.6.1 Internal Variables

When designing a system to control a robot, input variables are associated with perceptions (in other words, measures from sensors), as output variables are related to actuators. Except for very simple behaviors it is very difficult and not always possible to establish a direct relationship between input and output variables for a robot control system.

As an additional complication, sensors usually are not able to supply all the information necessary about the environment: processing as well as fusion of information from several sensors is often necessary. Some level of abstraction based on the sensor measures is unavoidable, and it is usually achieved by employing internal variables, called states. Internal variables fulfill several functions, for example they describe environment features, reason and deduce internal decisions, and express behavior-inherent

quantities.

A.1.6.2 Conditional Independence of Variables

edited In order to model complex behavior, internal variables are essential. As the number of variables in the control system increases (observation, motor and internal variables), it becomes critical to consider variable dependencies. In a complete model, it is assumed that all variables depend on each other, but in practical situations, especially with a very large number of variables, considering all possible dependencies between variables implies infeasible complexity. In these situations, conditional independence can be useful to reduce the problem complexity. In reality, two variables can be independent, but conditioned by a third variable. A practical example is when several sensors, identical or not, measure the same phenomenon. A basic assumption widely considered is that sensor readings depend only on the phenomenon being measured, and no dependence between the sensor readings is considered once the phenomenon is given. Whenever it is possible and reasonable, conditional independence is be applied to reduce intricacy in joint distributions.

A.1.7 Recursive Calculation

At each time step, states prediction and estimation depend on all past time states and this implies that number of calculations would increase with time. However, in a Bayes filter, it is possible to reuse last time step calculations.

States prediction calculation at a time step t uses states estimation at time t?1. Recursive calculation consists in the following steps:

- Initially finding the expression for prediction question.
- Then, for all other questions asked to the filter, find the expression in relation to the prediction question.
- Last step consists in rewriting prediction expression in a way that the estimation expression of previous time step can be used, which proves that recursive calculation is possible.

A.1.8 State Filtering

probabilistic questions for Bayes filters the name comes from the fact that its purpose is to filter out observation noise in order to estimate the system state. It is done using expression A.12 , which is conceptually divided in two steps: a) prediction which

projects the current belief state one time step into the future; and b) update, which improves the estimation by integrating the last obtained observation:

$$P(S_t|O_{1:t}) = \frac{1}{Z}\ P(O_t|S_t) \sum_{S_{t-1}} [P(S_t|S_{t-1})P(S_{t-1}|O_{1:t-1})] \tag{A.12}$$

Prediction is the most computationally expensive step, typically, this operation involves iterating through the N possible transitions for every state, hence, its time complexity is $O(N^2)$. However, the complexity of the prediction step may be reduced by imposing constraints on the structure of the transition matrix.

A.1.9 Bayesian Networks

Bayesian Networks, first introduced by Pearl (1988), have emerged as a primary method for dealing with probabilistic and uncertain information.

It is a directed acyclic graph encoding assumptions of conditional independence. Nodes in the Bayesian Network represent stochastic variables whereas arcs represent causal dependence. The network defines a joint (conditional) probability distribution (CPD) where the probability of an assignment to the stochastic variables is given by the product of the probabilities of each node. The probability distribution for each node is conditioned on its predecessors in the graph (Russell and Norvig, 2010).

References

Althaus, P., Ishiguro, H., Kanda, T., Miyashita, T., and Christensen, H. (2004). Navigation for human-robot interaction tasks. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2004*, volume 2, pages 1894–1900 Vol.2. IEEE. 25

Banerjee, C., Gupta, H., and Sushobhan, K. (2010). Low cost speech and vision based wheel chair for physically challenged. In *2nd International Conference on Computer and Automation Engineering (ICCAE)*, volume 1, pages 706–709. IEEE. 18, 27

Berjon, R., Mateos, M., Barriuso, A., Muriel, I., and Villarrubia, G. (2011). Alternative Human-Machine Interface system for powered wheelchairs. In *IEEE 1st International Conference on Serious Games and Applications for Health (SeGAH)*, pages 1–5. IEEE. 18, 19, 27

Bourhis, G. and Agostini, Y. (1998). The Vahm Robotized Wheelchair: System Architecture and Human-Machine Interaction. *Journal of Intelligent and Robotic Systems*, 22(1):39–50. 11

Bourhis, G. and Pino, P. (1996). Mobile robotics and mobility assistance for people with motor impairments: rational justification for the VAHM project. *IEEE Transactions on Rehabilitation Engineering*, 4(1):7–12. 23

Burke, J., Murphy, R., Rogers, E., Lumelsky, V., and Scholtz, J. (2004). Final Report for the DARPA/NSF Interdisciplinary Study on HumanRobot Interaction. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, 34(2):103–112. 57

Carnegie-Mellon-University (2014). CMU Sphinx - Speech Recognition Toolkit Webpage, http://cmusphinx.sourceforge.net/. 101

Carton, D., Turnwald, A., Kuhnlenz, K., Wollherr, D., and Buss, M. (2012). Proac-

tive human approach in dynamic environments. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3320–3321. IEEE. 25

Chung, S.-Y. and Huang, H.-P. (2010). A mobile robot that understands pedestrian spatial behaviors. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5861–5866. IEEE. 24

Ciolek, T. M. and Kendon, A. (1980). Environment and the Spatial Arrangement of Conversational Encounters. *Sociological Inquiry*, 50(3-4):237–271. 38

Demeester, E., Huntemann, A., del R. Millan, J., and Van Brussel, H. (2009). Bayesian plan recognition for Brain-Computer Interfaces. In *2009 IEEE International Conference on Robotics and Automation*, pages 653–658. IEEE. 23

Demeester, E., Huntemann, A., Vanhooydonck, D., Vanacker, G., Degeest, A., Brussel, H., and Nuttin, M. (2006). Bayesian Estimation of Wheelchair Driver Intents: Modeling Intents as Geometric Paths Tracked by the Driver. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5775–5780. IEEE. 23, 26

Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57. 30

Escobedo, A., Spalanzani, A., and Laugier, C. (2013). Multimodal control of a robotic wheelchair: Using contextual information for usability improvement. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4262–4267. IEEE. 4, 11

Fanelli, G., Gall, J., and Van Gool, L. (2011). Real time head pose estimation with random regression forests. *CVPR 2011*, pages 617–624. 18, 96

Fehr, L., Langbein, W. E., and Skaar, S. B. (2000). Adequacy of power wheelchair control interfaces for persons with severe disabilities: a clinical survey. *Journal of rehabilitation research and development*, 37(3):353–60. 2

Fox, D., Burgard, W., and Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, 4(1):23–33. 5, 58, 75, 80

Goebel, P. (2014). Pi Robot Webpage, http://www.pirobot.org/. 96

Grasse, R., Morère, Y., and Pruski, A. (2010). Assisted Navigation for Persons with Reduced Mobility: Path Recognition Through Particle Filtering (Condensation Algorithm). *Journal of Intelligent and Robotic Systems*, 60(1):19–57. 11

Hall, E. T. (1969). *The Hidden Dimension*. Anchor Books, first edit edition. 30, 36

Hart, P., Nilsson, N., and Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107. 29, 31, 33, 45

Hashimoto, M., Takahashi, K., and Shimada, M. (2009). Wheelchair control using an EOG- and EMG-based gesture interface. In *2009 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 1212–1217. IEEE. 18, 27

Hayduk, L. A. (1978). Personal space: An evaluative and orienting overview. *Psychological Bulletin*, 85(1):117–134. 36

Hemachandra, S., Kollar, T., Roy, N., and Teller, S. (2011). Following and interpreting narrated guided tours. In *2011 IEEE International Conference on Robotics and Automation*, pages 2574–2579. IEEE. 11

Huggins-Daines, D., Kumar, M., Chan, A., Black, A., Ravishankar, M., and Rudnicky, A. (2006). Pocketsphinx: A Free, Real-Time Continuous Speech Recognition System for Hand-Held Devices. In *2006 IEEE International Conference on Acoustics Speed and Signal Processing Proceedings*, volume 1, pages 185–188. IEEE. 100

Iturrate, I., Antelis, J., Kubler, A., and Minguez, J. (2009). A Noninvasive Brain-Actuated Wheelchair Based on a P300 Neurophysiological Protocol and Automated Navigation. *IEEE Transactions on Robotics*, 25(3):614–627. 17

Jaynes, E. T. (2003). *Probability Theory: The Logic of Science*. Cambridge University Press. 119

Kendon, A. (1976). Spatial organization in social encounters: The F-formation system. *Man Environment Systems*, 6:291–296. 37

Kendon, A. (2010). Spacing and Orientation in Co-present Interaction. In *Development of Multimodal Interfaces: Active Listening and Synchrony*, volume 5967 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin / Heidelberg. 37

Kirby, R., Simmons, R., and Forlizzi, J. (2009). COMPANION: A Constraint-Optimizing Method for Person-Acceptable Navigation. In *RO-MAN 2009 - The 18th IEEE International Symposium on Robot and Human Interactive Communication*, pages 607–612. IEEE. 29

Kollar, T., Tellex, S., Roy, D., and Roy, N. (2010). Grounding Verbs of Motion in Natural Language Commands to Robots. In *The 12th International Symposium on Experimental Robotics*, pages 31–47. Springer Berlin Heidelberg. 19

Konolige, K., Marder-Eppstein, E., and Marthi, B. (2011). Navigation in hybrid metric-topological maps. In *2011 IEEE International Conference on Robotics and Automation*, pages 3041–3047. IEEE. 45

Kuno, Y., Shimada, N., and Shirai, Y. (2003). A robotic wheelchair based on the integration of human and environmental observations - Look where you're going. *IEEE Robotics and Automation Magazine Automation Magazine*, 10(1):26–34. 18, 27

Laga, H. and Amaoka, T. (2009). Modeling the spatial behavior of virtual agents in groups for non-verbal communication in virtual worlds. In *Proceedings of the 3rd International Universal Communication Symposium*, page 154, New York, New York, USA. ACM Press. 36

Lam, C.-P., Chou, C.-T., Chiang, K.-H., and Fu, L.-C. (2011). Human-Centered Robot NavigationTowards a Harmoniously HumanRobot Coexisting Environment. *IEEE Transactions on Robotics*, 27(1):99–112. 25

Lankenau, A., Meyer, O., and Krieg-Bruckner, B. (1998). Safety in Robotics: the Bremen Autonomous Wheelchair. *5th International Workshop on Advanced Motion Control*, pages 524–529. 11, 20

Lopes, A. C., Nunes, U., Vaz, L., and Vaz, L. (2010). Assisted navigation based on shared-control, using discrete and sparse human-machine interfaces. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, volume 2010, pages 471–474. 22, 26

Lopes, A. C., Pires, G., and Nunes, U. (2012). RobChair: Experiments evaluating Brain-Computer Interface to steer a semi-autonomous wheelchair. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5135–5136. IEEE. 11, 17, 22, 27

Madarasz, R., Heiny, L., Cromp, R., and Mazur, N. (1986). The design of an autonomous vehicle for the disabled. *IEEE Journal on Robotics and Automation*, 2(3):117–126. 9

Marder-Eppstein, E., Berger, E., Foote, T., Gerkey, B., and Konolige, K. (2010). The Office Marathon: Robust navigation in an indoor office environment. In *2010 IEEE International Conference on Robotics and Automation*, pages 300–307. IEEE. xv, 5, 30, 33, 45, 79, 80, 81

Matsumoto, Y., Inot, T., and Ogsawara, T. (2001). Development of intelligent wheelchair system with face and gaze based interface. In *Proceedings 10th IEEE International Workshop on Robot and Human Interactive Communication. ROMAN 2001*, pages 262–267. IEEE. 18

McKenna, S. J., Jabri, S., Duric, Z., Rosenfeld, A., and Wechsler, H. (2000). Tracking Groups of People. *Computer Vision and Image Understanding*, 80(1):42–56. 102

Mortenson, W. B., Miller, W. C., Backman, C. L., and Oliffe, J. L. (2012). Association between mobility, participation, and wheelchair-related factors in long-term care residents who use wheelchairs as their primary means of mobility. *Journal of the American Geriatrics Society*, 60(7):1310–1315. 1

Murai, A., Mizuguchi, M., Saitoh, T., Osaki, T., and Konishi, R. (2009). Elevator available voice activated wheelchair. In *IEEE International Symposium on Robot and Human Interactive Communication RO-MAN*, pages 730–735. IEEE. 19

Murphy-Chutorian, E. and Trivedi, M. M. (2009). Head pose estimation in computer vision: a survey. *IEEE transactions on pattern analysis and machine intelligence*, 31(4):607–26. 18

Mutlu, B. and Forlizzi, J. (2008). Robots in organizations. In *Proceedings of the 3rd international conference on Human robot interaction - HRI '08*, pages 287–294, New York, New York, USA. ACM Press. 29

Open-Source-Robotics-Foundation (2014). ROS.org webpage, http://www.ros.org/. xiii, 54

Papadakis, P., Spalanzani, A., and Laugier, C. (2013). Social mapping of human-populated environments by implicit function learning. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1701–1706. IEEE. 54

Pasteau, F., Babel, M., and Sekkal, R. (2013). Corridor following wheelchair by visual servoing. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2013, pages 590–595. IEEE. 11, 21

Patel, S., Ostrowski, J., Rao, R., and Taylor, C. (2002). Sensor based door navigation for a nonholonomic vehicle. *Proceedings 2002 IEEE International Conference on Robotics and Automation*, 3:3081–3086. 11, 21

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc. 125

Perrin, X., Chavarriaga, R., Colas, F., Siegwart, R., and Millán, J. d. R. (2010). Brain-coupled interaction for semi-autonomous navigation of an assistive robot. *Robotics and Autonomous Systems*, 58(12):1246–1255. 22, 24

Pope, A. M. and Tarlov, A. R. (1991). *Disability in America: Toward a National Agenda for Prevention*. Committee on a National Agenda for the Prevention of Disabilities, Institute of Medicine USA. 1

Rebsamen, B., Guan, C., Zhang, H., Wang, C., Teo, C., Ang, M. H., and Burdet, E. (2010). A brain controlled wheelchair to navigate in familiar environments. *IEEE transactions on neural systems and rehabilitation engineering.*, 18(6):590–8. 17, 27

Rios-Martinez, J., Spalanzani, A., and Laugier, C. (2011). Understanding human interaction for probabilistic autonomous navigation using Risk-RRT approach. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2014–2019. IEEE. 5, 32, 38

Rios-Martinez, J.-A. (2013). *Socially-Aware Robot Navigation : combining Risk Assessment and Social Conventions.* PhD thesis, University of Grenoble. 25, 29, 32, 33, 36, 38, 42

Russell, S. J. and Norvig, P. (2010). *Artificial intelligence: A modern approach.* Prentice Hall, third edition. 45, 125

Ruzaij, M. F. and Poonguzhali, S. (2012). Design and implementation of low cost intelligent wheelchair. *2012 International Conference on Recent Trends in Information Technology*, pages 468–471. 19

Schalkwyk, J., Beeferman, D., Byrne, B., Chelba, C., Cohen, M., Garret, M., and Strope, B. (2010). Google Search by Voice: A case study. *Advances in Speech Recognition*, pages 61–90. 101

Sehestedt, S., Kodagoda, S., and Dissanayake, G. (2010). Robot path planning in a social context. In *2010 IEEE Conference on Robotics, Automation and Mechatronics*, pages 206–211. IEEE. 24

Shi, C., Shimada, M., Kanda, T., Ishiguro, H., and Hagita, N. (2011). Spatial Formation Model for Initiating Conversation. In *Proceedings of the VII Robotics Science and Systems Conference*, Los Angeles. 25

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A. (2011). Real-time human pose recognition in parts from single depth images. In *CVPR 2011*, pages 1297–1304. IEEE. 102

Simmons, R. (1998). The lane-curvature method for local obstacle avoidance. In *Proceedings. 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications*, volume 3, pages 1615–1621. IEEE. 23

Simpson, R. C. (2005). Smart wheelchairs: A literature review. *The Journal of Rehabilitation Research and Development*, 42(4):423. 10, 21, 26

Simpson, R. C. and Levine, S. P. (2002). Voice control of a powered wheelchair. *IEEE transactions on neural systems and rehabilitation engineering*, 10(2):122–125. 19, 27

Simpson, R. C., LoPresti, E., and Cooper, R. A. (2008). How many people would benefit from a smart wheelchair? *The Journal of Rehabilitation Research and Development*, 45(1):53–72. 1

Sisbot, E., Marin-Urias, L., Alami, R., and Simeon, T. (2007). A Human Aware Mobile Robot Motion Planner. *IEEE Transactions on Robotics*, 23(5):874–883. 24, 27

Spinello, L. and Arras, K. O. (2011). People detection in RGB-D data. *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3838–3843. 102

Taha, T., Miro, J. V., and Dissanayake, G. (2008). POMDP-based long-term user intention prediction for wheelchair navigation. In *2008 IEEE International Conference on Robotics and Automation*, pages 3920–3925. IEEE. 24, 26

Tahboub, K. a. (2006). Intelligent Human-Machine Interaction Based on Dynamic Bayesian Networks Probabilistic Intention Recognition. *Journal of Intelligent and Robotic Systems*, 45(1):31–52. 58

Tanaka, K., Matsunaga, K., and Wang, H. (2005). Electroencephalogram-based control of an electric wheelchair. *IEEE Transactions on Robotics*, 21(4):762–766. 27

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, California. 32, 50

Thrun, S., Fox, D., Burgard, W., and Dellaert, F. (2001). Robust Monte Carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2):99–141. 49, 50

Tranberg Hansen, S., Svenstrup, M., Andersen, H. J., and Bak, T. (2009). Adaptive human aware navigation based on motion pattern analysis. In *RO-MAN 2009 - The 18th IEEE International Symposium on Robot and Human Interactive Communication*, pages 927–932. IEEE. 24

Vanacker, G., del R Millán, J., Lew, E., Ferrez, P. W., Moles, F. G., Philips, J., Van Brussel, H., Nuttin, M., Mill, R., Gal, F., and Brussel, H. V. (2007). Context-based filtering for assisted brain-actuated wheelchair driving. *Computational intelligence and neuroscience*, 2007:25130. 22

Vanhooydonck, D., Demeester, E., Hüntemann, A., Philips, J., Vanacker, G., Van Brussel, H., and Nuttin, M. (2010). Adaptable navigational assistance for intelligent wheelchairs by means of an implicit personalized user model. *Robotics and Autonomous Systems*, 58(8):963–977. 11

Wang, Y. and Chen, W. (2011). Hybrid map-based navigation for intelligent wheelchair. *2011 IEEE International Conference on Robotics and Automation*, pages 637–642. 23

Wei, L. and Hu, H. (2011). A multi-modal human machine interface for controlling an intelligent wheelchair using face movements. *2011 IEEE International Conference on Robotics and Biomimetics*, pages 2850–2855. 27

Yamaoka, F., Kanda, T., Ishiguro, H., and Hagita, N. (2009). Developing a model of robot behavior to identify and appropriately respond to implicit attention-shifting. In *4th ACM/IEEE International Conference on Human-Robot Interaction (HRI), 2009*, pages 133–140, La Jolla, CA. IEEE. 25

Yamaoka, F., Kanda, T., Ishiguro, H., and Hagita, N. (2010). A Model of Proximity Control for Information-Presenting Robots. *IEEE Transactions on Robotics*, 26(1):187–195. 25