CIMAT

Exploration of an unknown environment with a differential drive disc robot

by

Guillermo Jesús Laguna Mosqueda

Supervisors:

Dr. Rafael Murrieta Cid

Dr. Héctor Manuel Becerra Fermín

A thesis submitted in fulfilment of the requirements for the degree of Master in computer science

in the

Computer Science Department

October 2013

Declaration of Authorship

I, Guillermo Jesús Laguna Mosqueda, declare that this thesis titled, 'Exploration of an unknown environment with a differential drive disc robot' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this Research Center.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this Research Center or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: ____

Date: ____

"Research is formalized curiosity. It is poking and prying with a purpose."

Zora Neale Hurston

"This thesis is just the report of a long processs which started more than one year ago. It cannot express the long days spent infront of my computer, battling conceptual and theoretical issues and battling countless bugs and technical troubles. Many things have changed since the original approach to the problem, it has been a rich process of learning, with hard times when attempts were uncsuccessful, but also with great moments of happiness when there was good results. There were many long nights and bad times... But, at the end, the greatest joy of this work is that I could proudly say in a near future that I contributed to the existence of Skynet, the existence of Weyland Industries, or a reality similar to the one presented in the Matrix universe."

Guillermo J. Laguna

CIMAT

Abstract

Computer Science Department

Master in computer science

Exploration of an unknown environment with a differential drive disc robot

by Guillermo Jesús LAGUNA MOSQUEDA

This thesis work addresses the problem of exploring an unknown, planar, polygonal and simply connected environment. A salient object (i.e. a landmark) is located in the environment. The collision-free subset of the robot's configuration space is simply connected or it might have several connected components. The robot is a differential drive system shaped as a disc. The robot has limited sensing, namely it is incapable of measuring any distance or angle, or performing self localization. The exploration problem consists in discovering the environment with the robot's sensors. To solve this problem, a motion policy is developed based on simple sensor feedback and a complete exploration strategy is represented as a Moore machine. The proposed exploration strategy guarantees that the robot will discover the largest possible region of the environment. Consequently, the robot will find the landmark or declare that an exploration strategy to find it does not exist.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisors Dr. Rafael Murrieta and Dr. Héctor Becerra for giving me the opportunity to work with them in such an interesting project, for their patience, motivation, enthusiasm, and immense knowledge. Their guidance helped me in all the time of research and writing of this thesis. I could not have imagined having better advisors and mentors for my master studies.

Besides my advisors, I would like to thank Rigoberto López for his ideas for the project and for the times that he helped me with some technical details of the implementation. And finally, my sincere thanks to Israel Becerra, for his quick review of this work and his tips.

Finally, I would like to thank CONACyT for all the financial support received through the master scolarship, which allowed me to focus completely on my studies, without the neediness of an additional job to cover my expenses.

Contents

Declaration of Authorship **i**

Abstract **iii**

Acknowledgements **iv**

List of Figures **vii**

Abbreviations **x**

Symbols **xi**

1 Introduction **1**
1.1 Research field and motivation 1
1.2 Problem Statement 2
1.3 Main Objective 4
1.4 Outline of chapters 4

2 State of the art **5**
2.1 Environment models 5
2.1.1 Occupancy grid model 6
2.1.2 Topological model 6
2.1.3 Geometric model 7
2.1.4 Semantic model 8
2.2 Incremental construction of geometric and topological maps 8
2.3 Motion strategies for exploration 10
2.3.1 SLAM problem 11
2.3.2 Approach without localization 11
2.4 Motion planning for exploration 13
2.4.1 Greedy approach 13
2.4.2 Gonzalez Banos approach 14
2.4.3 Makarenko approach 14
2.4.4 Amigoni approach 16
2.4.5 Simmons approach 17
2.4.6 Tovar approach 17

2.4.7 Newman approach . . . 19
2.4.8 Feder approach . . . 19
2.4.9 Calisi approach . . . 19
2.5 Multi-robot exploration strategies . . . 20
2.6 The Lopez optimal navigation strategy . . . 20

3 Sensing Model **22**
3.1 Robot's sensor and landmark . . . 22
3.1.1 Gap Navigation Tree . . . 23
3.2 A Tactile Bumper . . . 26
3.3 The observation vector . . . 26
3.4 Landmark Encoding . . . 29

4 Motion Model **32**
4.1 Motion Model . . . 32

5 The Exploration Automaton **34**
5.1 Motion Policy . . . 36
5.2 The Local Exploration Algorithm . . . 37

6 Implementation **42**
6.1 Environment 1 . . . 42
6.2 Environment 2 . . . 44
6.3 Environment 3 . . . 46

7 Conclusion **49**
7.1 Conclusion . . . 49
7.2 Future work . . . 49

A Local Exploration Example **51**

Bibliography **56**

List of Figures

1.1 An unreachable region which belongs to the same connected space of $\mathcal{C}$. A landmark Λ cannot lie inside the unreachable region because it would represent an additional connected component of $\mathcal{C}$. 3

1.2 The robot lies inside a $\mathcal{C}$-space component and there is an unreachable region which belongs to other connected $\mathcal{C}$-space component. Then, a landmark Λ of the same radius than the robot can lie inside $E_{unreach}$. . . 4

2.1 A topological model representation, obtained from [1]. 7

2.2 A real environmental 2-D projection and its reconstructed geometric model. 7

2.3 A semantic model obtained from [2]. 8

2.4 Laser telemeter readings. (a) Scene, (b) Captured points. 9

2.5 Four partial models fused at a single given position. 10

2.6 (a) and (b) represent the robot's visibility polygon and the boundaries between the current visibility region and the environment's non visible regions. They are represented as local maps, (c) is the already exposed environment obtained from two sensing locations and the boundaries betweent the already exposed environment and the regions which have not been seen yet, represented in a global map. 13

3.1 Representation of the robot's sensors. 23

3.2 (a) The robot has its omnidirectional sensor over lp, it lies within E and gaps g_1, g_2 and g_3 are detected. On the left, the visibility region is shown with the angular position of the gaps, it is clear that statements such as $[g_1, g_2, g_3] = [g_2, g_3, g_1] = [g_3, g_1, g_2]$ can be made, (b) The omnidirectional sensor is in contact with ∂E, the same gaps are detected. On the left, the visibility region is partially obstructed by ∂E, it is shown as a gray colored region. The visibility obstruction can be used to establish a linear order of the gaps following a counterclockwise sense $[g_1, g_2, g_3]$, or a clockwise sense $[g_3, g_2, g_1]$. 24

3.3 (a) The robot moves from position a to b, when the robot crosses the inflection ray, a region of the environment is occluded, therefore a new gap appears. A new node is added to the root, (b) The opposite case, the robot moves from a to b, when the robot crosses the inflection ray the gap disappears. Its corresponding node is removed, (c) In position a the sensor detected a gap, then the robot moves to b, when the robot crosses the bitangent complement the gap is divided into two gaps. The original node is removed and replaced by two gaps, (d) In position a the sensor detected two gaps, then the robot moves to b, when the robot crosses the bitangent complement the gaps merge into a new gap. The original gaps become children of a new node that is connected to the root. 25

3.4 (a) The direction rt is aligned with it the edge of ∂E that point rp is touching. Therefore, $aligned = 1$, (b) Point rp is touching a reflex vertex, and the preferential direction rt is not aligned with the first polygonal edge in clockwise sense starting from direction rt that is in contact with rp. Therefore, $aligned = 0$, (c) Point rp is touching a reflex vertex, and the preferential direction rt is aligned with the first polygonal edge in clockwise sense starting from direction rt that is in contact with rp. Therefore, $aligned = 1$. If the omnidirectional sensor is located at lp, an alignement is just the left symmetric case. 27
3.5 Λ is partially visible, it is occluded by the reflex vertex that generates gap a. 29
3.6 Λ is completely visible. 30
3.7 Λ is partially occluded by the refelex vertex that generates a new gap b. . 30

4.1 The motion primitives: (a) Clockwise rotation in place, (b) Counterclockwise rotation in place, (c) Straight line motion, (d) Clockwise rotation w.r.t. rp, (e) Counterclockwise rotation w.r.t. lp. 32

5.1 The finite-state machine that represents the exploration strategy. 36
5.2 This figure shows region R_{na} in white and region R_a in dark grey. The regions are divided by the arc of circle trajectory followed by the omnidirection sensor during a robot's rotation in place. The figure also shows a light source s with a ray of light which goes from R_a to R_{na}. 40

6.1 The robot is executing the straight line motion primitive. The corresponding GNT is shown, the circles represent non primitive nodes (for illustrative purposes, they the same orientation that the gaps they represent) and the triangle is the encoded landmark. 43
6.2 The landmark is partially occluded, the current GNT is presented and it shows that the landmark is encoded in the GNT (the primitive leaf nodes are shown like squares). 43
6.3 The robot is performing rotation w.r.t.rp because it is touching an environment's reflex vertex. 43
6.4 The robot has found a narrow gate, the control applied is u_3, while *local exploration* algorithm is executed. The GNT is shown and the nodes corresponding to the gaps within the unreachable region are not labeled as primitive nodes yet, because *local exploration* has not ended. 44
6.5 The robot and the environment at the beginning of the exploration task. The starting GNT is also shown. 44
6.6 Λ is partially visible for the first time. 45
6.7 The whole area of Λ is visible, the corresponding GNT is shown with the landmark node connected to the root. 45
6.8 Λ becomes partially occluded and the landmark node is now a child of the corresponding GNT node. 45
6.9 The whole landmark is visible again, and it becomes a root's child node as shown in the GNT. 46
6.10 The exploration task ends and the largest possible region of the environment has been discovered. 46

6.11 The robot and the environment at the beginning of the exploration task in an environment where $\mathcal{C}$ has two connected components. The starting GNT is also shown. 47
6.12 The robot faces the gate between two components of connected component of $\mathcal{C}$. *Local exploration* algorithm is applied while the control u_3 is executed. The GNT is also shown. 47
6.13 *Local exploration* ends and the gaps within the unreachable region (the other connected component of $\mathcal{C}$) are labeled as primitive nodes. 48
6.14 The exploration task ends and the largest possible region of the environment has been discovered. 48

A.1 (a) Current GNT at the beginning of *local exploration*, (b) Gaps out of the unreachable region, (c) Gaps within the unreachable region, those gaps must be labeled as primitive ones, after executing *local exploration* algorithm. 51
A.2 (a) GNT with c and d as the children nodes of l due to the merge event, (b) The new gap l. 52
A.3 (a) Current GNT with b and l as the children nodes of m due to the merge event, (b) The new gap m. 52
A.4 (a) Current GNT with e and f as the children nodes of n due to the merge event, (b) Gaps e and f right before the merge event, (c) The new gap n. 53
A.5 (a) Current GNT with a and m as the children nodes of o due to the merge event, (b) The new gap o. 53
A.6 (a) GNT with resulting gaps e and f after gap n splits (they were its children nodes), (b) The instant before n splits, (c) The resulting gaps e and f after the split event. 54
A.7 (a) The GNT after gap e has disappeared, (b) The gaps within the unreachable region and gap e before disappearing, (c) The gaps after the disappearance event. 54
A.8 (a) Current GNT with the new gap p, (b) The instant before the gap appear event, (c) The new gap p due to the gap appear event. 55
A.9 (a) The GNT at the end of *local exploration* when the primitive label has been propagated to the leaf nodes representing unreachable gaps, (b) The gaps outside of the unreachable region, (c) The gaps within the unreachable region that have been correctly labeled as primitive ones. . . 55

Abbreviations

GNT	**G**ap **N**avigation **T**ree
FSM	**F**inite **S**tate **M**achine
SLAM	**S**imultaneous **L**ocalization and **M**apping
NBV	**N**ext **B**est **V**iew

Symbols

r	robot's radius
E	environment
∂E	environment's boundary
$\mathcal{C}$	collision-free subset of the robot's configuration space
t	time
lp	extremal left side point of the robot
rp	extremal right side point of the robot
lt	direction of the line tangent to the robot's boundary at lp
rt	direction of the line tangent to the robot's boundary at rp
Λ	static disc-shaped landmark with the same radius as the robot
ω_l	angular velocity of the left robot's wheel
ω_r	angular velocity of the right robot's wheel
M	robot's planner

I would like to thank God for the life he has given me. I would like to dedicate my thesis to my beloved parents Guillermo Laguna and Fabiola Mosqueda, who have been a great life example for me, they have given me everything in my life. Thanks for your inconditional love, specially in difficult times.

I dedicate this thesis also to my brother Abraham Laguna who has become a very good friend, and whom I forgot to mention in my bachelor thesis.

I would dedicate this thesis also to my friends for their support, and to my classmates for all the favors and help provided during this time of hard work, for the sleepless nights we were working together before deadlines and for all the fun we have had. It was great to share this phase of my life with people like you.

Chapter 1

Introduction

1.1 Research field and motivation

Robotics is a wide research field that has many ramifications. One of the most important areas is robot motion planning, which consists in providing a motion policy and a planner to an autonomous robot. A planner allows the robot to perform a specific task, being the exploration of unknown environments one of the most typical ones. The main objective is to provide the autonomous robots the ability to explore their environment and the ability to build a representations of that environment. Those environments' representations can be used for navigating effectively within the environment. There are many works belonging to this area. In each work the robot is defined with specific sensing and specific motion capabilities. The sensing capabilities allow the robot to obtain information of the work space through sensors' readings, and the robot must use that information for building an internal representation of the environment. This representation could be used in some cases for localizing the robot in a global (or local) frame, estimate its state, and then decide which control must be applied for achieving a goal (i.e. avoiding an obstacle or selecting an optimal movement direction). In each work, the sensing capabilities, the technique used to represent the environment, the features of interest of the environment that must be found by the sensors, and the motion planning policy vary. This variety is the reason why the same problem has been addressed in many different ways. Some related works that have addressed the problem of automatic model building are [1, 3–8]. These works are detailed in Chapter 2 where a summary of the state of the art is presented.

This thesis work addresses the problem of exploring unknown environments. It brings a novel exploration strategy for a differential drive mobile robot with limited sensing capabilities. This strategy is different to the previous works in the literature. There are related works which model the robot as a point. From a theoretical point of view, this approach has allowed to solve some problems of robot navigation; however, for more realistic tasks, this approach is not sufficient. Modeling the robot as a point ignores the robot's physical dimensions and that assumption may impact the true performance. A natural step forward, and more realistic, is to consider the robot as a nonzero size entity. A disc shape is one of the most simple ones. In this work, the robot is modeled as a disc and the exploration strategy is based on that assumption. The robot's size represents additional constraints in the configuration space, specifically, a growth of the obstacle's size in a measure related to the robot's radius. This raises the main conceptual difference between a point robot and a disc robot, which makes necessary the design of navigation strategies specific for a disc robot. For a point robot, the concept of visibility is equal to the concept of reachability. It means that if the robot can see certain environment place, that place is also reachable for the robot. However, this property is not necessarily true for a disc robot due to the configuration space constraints. Indeed, if the configuration space is not observable, the robot cannot measure it. To solve an exploration problem, the disc robot must be able to infer information of the configuration space from the workspace.

In [9], the authors have presented an optimal navigation technique in terms of distance traveled for a disc robot. In this work, a main motivation is to provide a compatible exploration strategy with respect to the work presented in [9], for developing a complete planner (with exploration and navigation strategies) for a disc robot.

The main contribution of this thesis work is a complete exploration strategy that reports whether all the environment has been seen or the largest possible region of it has been seen. A strategy is proposed to deal with cases where inaccessible places are found. These cases represent a challenge given that a portion of the environment might not be visible to the robot and the strategy must ensure to see as much as possible. The environment is represented in an efficient data structure, the GNT. Additionally, the proposed strategy is relatively easy to implement and compact, in such a way that it is represented as a Moore machine.

1.2 Problem Statement

The robot has the shape of a disc with radius r moving in an unknown, planar, polygonal, and simply connected environment which could be any compact set $E \subset \mathbb{R}^2$ for which the interior of E is simply connected[1] and the boundary ∂E of E is the image of a piecewise-analytic closed curve.

However, it is assumed that the collision-free subset of the robot's configuration space $\mathcal{C}$ is simply connected or it might have several connected components. $\mathcal{C}$-space obstacle corresponds to that of a translating disc, that is, the extended boundary of E which is due to the robot's radius[2].

A salient object (i.e. a landmark) is located in the environment. The robot is unable to localize itself in *any reference frame*, and has limited sensing capabilities, namely it is incapable of *measuring any distance or angle*. The detailed description of the robot sensing capabilities is presented in Chapter 3.

There are two kinds of unreachable regions. The first one is presented in Figure 1.1. It shows an unreachable region $E_{unreach} \subset E$ colored in light gray (yellow) which might exist in an environment with the collision-free subset of the configuration space $\mathcal{C}$ simply connected. It might also exist in an environment where $\mathcal{C}$ has more than one connected component. $E_{unreach}$ is small enough in such a way that the robot cannot lie in the interior of $E_{unreach}$, therefore it does not represent an additional connected component of $\mathcal{C}$. The robot is represented as a medium gray (green) disc with its gap sensor as a light gray (yellow) disc over the robot's boundary. A landmark Λ of the same radius as the robot lies inside the unreachable region, it is represented a medium gray disc.

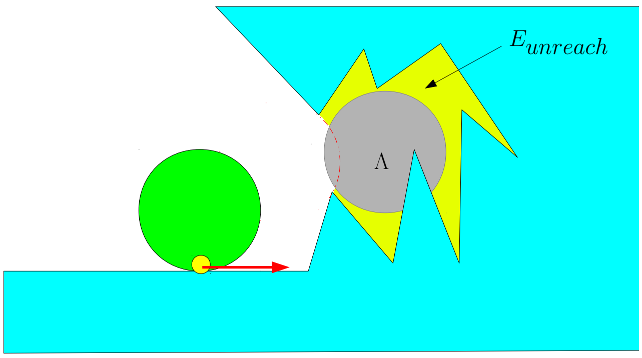


FIGURE 1.1: An unreachable region which belongs to the same connected space of $\mathcal{C}$. A landmark Λ cannot lie inside the unreachable region because it would represent an additional connected component of $\mathcal{C}$.

[1]Every path between two points contained in E can be continuously transformed, staying within E, into any other path while preserving the two endpoints in question. It has a single homotopy manifold.

[2]Note that this is the configuration space for a translating disc rather than for a rigid body because of rotational symmetry.

Figure 1.2 shows the second type of unreachable region. In this case $E_{unreach}$ is one of the connected components of $\mathcal{C}$ where the robot does not lie. This means that $E_{unreach}$ is large enough in such a way that Λ might lie in the interior of $E_{unreach}$.

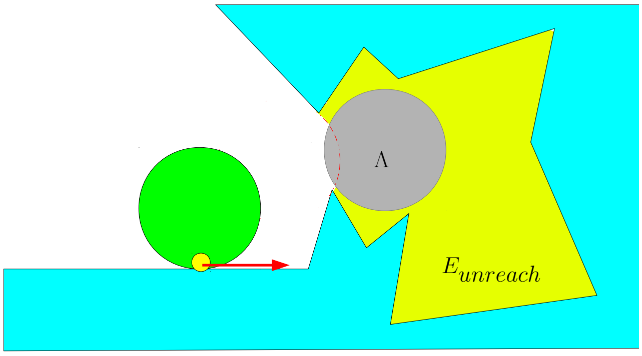


FIGURE 1.2: The robot lies inside a $\mathcal{C}$-space component and there is an unreachable region which belongs to other connected $\mathcal{C}$-space component. Then, a landmark Λ of the same radius than the robot can lie inside $E_{unreach}$.

1.3 Main Objective

The main goal of this strategy is to explore an environment through wall following. That is, while the robot moves, the visibility region of the robot's sensor must cover the environment E at least once, or in the worst case, the largest possible region of E. Consequently, the robot will find the landmark or declare that an exploration strategy to find it does not exist.

1.4 Outline of chapters

The remainder of this work is organized as follows: Chapter 2 presents a summary of the state of the art about exploration and map building of unknown environments.

Chapter 3 and 4 formally present the robot's model including sensing and motion capabilities. Chapter 5 presents an automaton or (Moore) finite-state machine that models the complete exploration strategy of an unknown environment. It also presents a feedback motion policy that maps observations to robot commands.

Chapter 6 presents an implementation in simulation considering different scenarios. Finally, Chapter 7 concludes this work and presents future research directions.

Chapter 2

State of the art

Autonomous robots must possess the ability to explore their environments, build representations of those environments, and then use those representations to navigate effectively in those environments. Automatic model building is then a very important research subject. Some works developed in this area are [1, 4, 10, 11]. The main problem is building an environment representation using sensor information and computational techniques. The models are built incrementally while the robot is exploring, both tasks are made simultaneously.

The models of the environment have been used for dealing with other tasks, for example, navigation (moving from one place to another without colliding with obstacles), finding objects, tracking targets, and many additional tasks. The model of the environment could be built using strong sensors. The standard SLAM approach [12, 13] is a representative example of this paradigm. Other approaches are more minimalistic and use limited sensor capabilities. Due to this limitation, these works must deal with the exploration task without self localization or pose estimation. The work developed in this thesis belongs to this second group.

In general, a system for automatic environment model building must consider three problems related: i) the incremental building of the environment's model, ii) obtaining the robot's configuration with respect to the environment's model, and iii) a robot's motion strategy.

2.1 Environment models

First of all, the model used for representing the environment must be chosen. In [4] three kinds of model are proposed, they are: i) topological maps, ii) geometrical maps, and iii) semantic maps.

Another classification is mentioned in [14]. It includes the topological maps [15], occupancy grids [5] and feature based maps [4, 16, 17].

2.1.1 Occupancy grid model

Occupancy grids use a 2-D array to represent the environment. There, each cell takes one of three values: i) free space, ii) occupied space, or iii) unknown space. Grid-based algorithms have proved to be very simple and quite useful for obstacle avoidance and planning purposes [5]. However, when the size of the environment is large, these models become difficult to handle.

2.1.2 Topological model

Topological maps can be expressed as a graph, where the nodes represent places and the edges represent adjacency, or direct connectivity [4]. The topological model is a hierarchy of various levels. The concept of place is defined as an area which is a topological unit. For example, a room or a corridor are topological units. The concept of place is also defined at various levels: a room, a floor, or a complete building are places, but each of them belongs to a different hierarchy level. In an outdoor environment the concept of place can be defined as well.

Connectors link places, indoor connectors may be doors, corridors, stairs, elevators or even some places are connectors themselves. In a detailed level, the space topology is tipically inferred from a geometric model. The structuration of free space is obtained through building convex polygons called cells, they are based on the vertices and edges of the obstacles. A cell represents a place, and this place is the most basic unit of space partitioning, therefore a cell has minimum spatial hierarchy. A set of adjacent cells might be a place too, and this place yielding more than one cell belongs then to a higher spatial hierarchy. The edges between cells are connectors. The cell's connectivity graph gives the first topological representation of the space structure. While the robot is moving and discovering new areas, the space structuration is made, and the graph is updated.

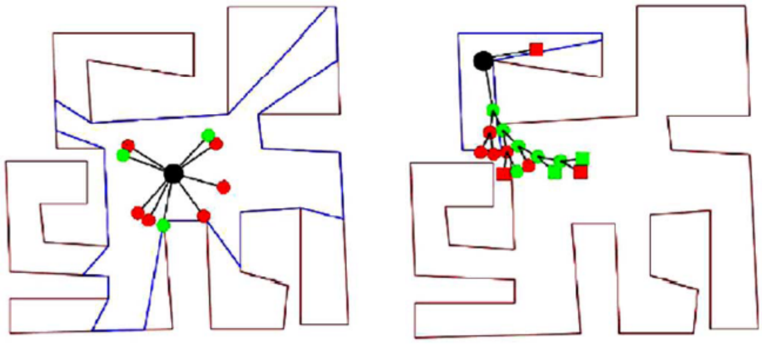


FIGURE 2.1: A topological model representation, obtained from [1].

In [1] a topological structure of navigation (GNT) which is not based on any metric is presented. This structure is based on gap chasing (discontinuities in space) for building the shortest path representation.

2.1.3 Geometric model

The geometric model is deduced directly from perception data. It gives feature based maps such as [16] and it may portray a 2-D [4] or 3-D model [17]. One of the most popular used primitives is the line segment, which can be extracted from ultrasonic data [18], laser range-finder data [19], or vision data [20, 21].

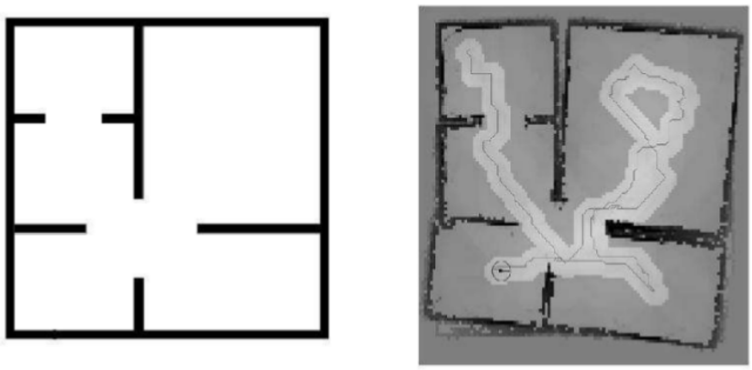


FIGURE 2.2: A real environmental 2-D projection and its reconstructed geometric model.

In [4] the objects have polyhedral shapes and they are projected in the plane for obtaining a 2-D model. This model basically includes the position of the robot and the obstacles defined with respect to a global reference frame. This representation can be manipulated for deducing the topological and semantical models. Some results from related works are shown in Figure 2.2.

2.1.4 Semantic model

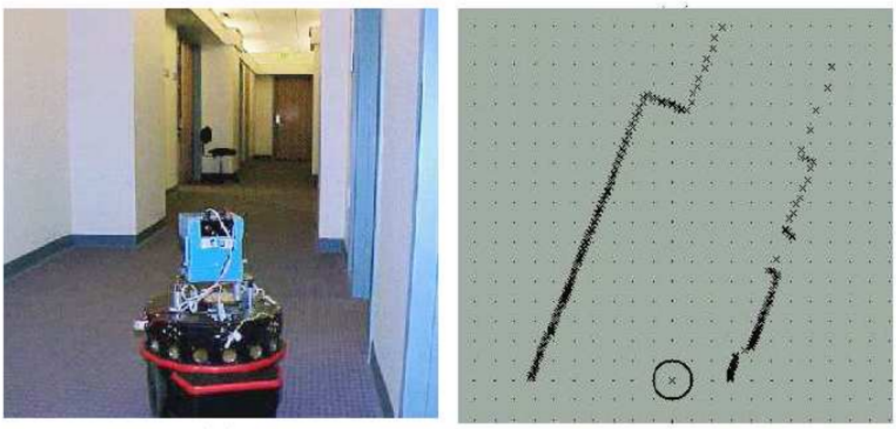
The semantic model contains information about objects, for example, their class or category and properties about the space and its relations. This model is partially distributed between the geometric model and the topological model. In [22], semantic labels of places (corridors and rooms) are obtained through the information given by a laser telemeter. Figure 2.3 shows a semantic model example of an outdoor environment.

FIGURE 2.3: A semantic model obtained from [2].

2.2 Incremental construction of geometric and topological maps

Building maps is an incremental process, due to the limited sensors' range. Occlusions may occur during exploration, and consequently the building process could be inaccurate. Therefore, the map must be built using many sensor readings at different positions and then these measures must be integrated into a global map.

Ideally, a 2-D indoor environment's model is the projection over a horizontal plane of the objects in the environment which represent obstacles for the robot, and that occlude the robot's sensors view. However, due to the sensors' limitations, the models are often an approximation of a transversal section of the environment at a fixed height. A frequently used approach is to integrate information provided by a laser telemeter. Figure 2.4 shows the information collected by a laser telemeter. In [23], the robot builds a polygonal model while moving to different locations, which are selected by a planner (a human user could select these locations too). Figure 2.5 shows four partial models of an indoor environment. The robot is located in a place where it rotates at four consecutive orientations, each one separated by 90 degrees.



(a) (b)

FIGURE 2.4: Laser telemeter readings. (a) Scene, (b) Captured points.

The model in (a) was built at the first orientation. The model in (b) was obtained by integrating the model in (a) with the local model generated at the second orientation, and so on. For the global geometric map building, an alignment process or registration between the global model and the local models might be required. As it was said before, the global model is built in an incremental way (see Figure 2.4). Additionally, raw data obtained by the sensors could be clustered in other entities. For example, the points obtained by a laser may be grouped in line segments through clustering techniques and line fitting. The resulting line segments might be also grouped in segments lists which yield polygons.

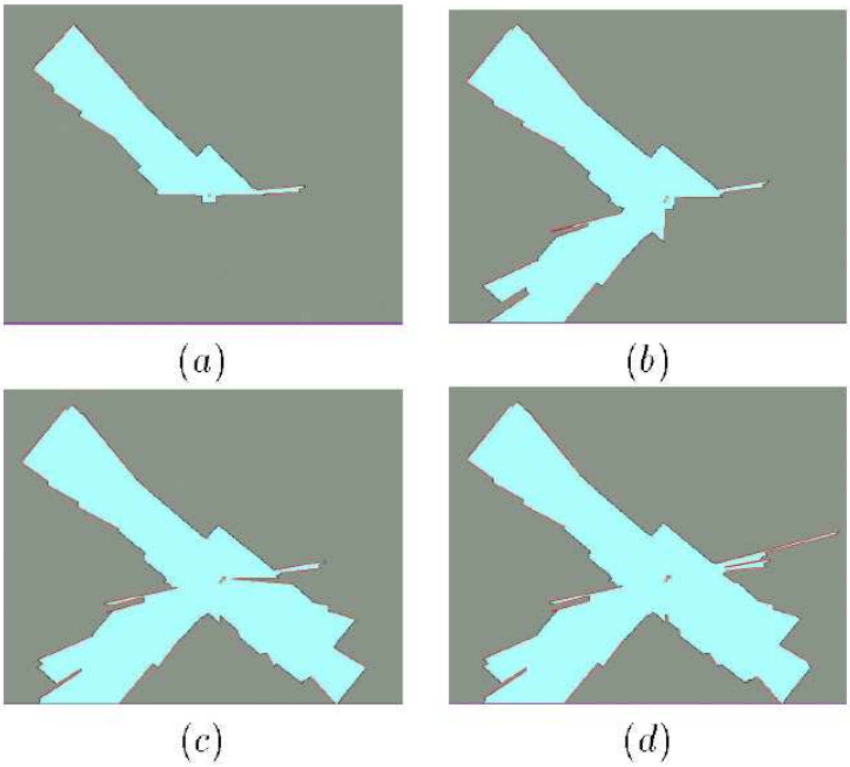


Figure 2.5: Four partial models fused at a single given position.

2.3 Motion strategies for exploration

During the incremental map building process, the motion strategies for exploration must handle the problem of selecting sensing configurations based on the global map built until that time. This problem makes this task different to the problem of finding a collision free path between an initial and a final configuration because the subgoals (sensing configurations) must be determined automatically. Some exploration strategies in the literature, in which the robot is not handled manually will be mentioned further in this Chapter.

There are systems which use a fixed path for exploring an environment. For example, the exploration strategy proposed in [24] includes a strategy where a robot traces concentric circles successively starting at its initial position. This approach is extended in [25] using parameterized spiral trajectories for managing the exploration of an environment. Other strategies move the robot to random sensing configurations without an explicit evaluation of the utility of the next sensing configuration.

2.3.1 SLAM problem

There are some models of environment exploration which are able to locate the robot with respect to a global reference frame and which can also estimate the robot's pose. These works define the problem of Simultaneous Localization and Mapping (SLAM) [26–29]. SLAM has become a very active topic of research in last decades.

Up to now, work on SLAM has focused primarily on issues related to uncertainty in sensing. Early research on SLAM [29] used a Kalman filtering approach to manage uncertainties that accumulate during the robot's motion, simultaneously providing an estimate of the robot's position and landmarks' locations. More recently, generalized Bayesian approaches have been proposed. For example, in the work of Montemerlo [28] the authors propose to relax the restrictive conditions imposed by Kalman filtering methods using a Monte Carlo approach for the SLAM problem. The mentioned works have focused on developing techniques to extract relevant information from raw data and to integrate the collected data into a single model. Standard SLAM approach does not give a motion strategy but it could be applied to obtain a map that can be used as input to motion strategies for exploration.

2.3.2 Approach without localization

Some exploration strategies are unable of self localization and have limited sensing capabilities. In [11] for example, a method is proposed for building a global geometric map by registering scans collected by a laser range sensor without pose estimation. In [8] is presented a work which addresses the problem of exploring an unknown environment for searching one or more recognizable targets. This method assumes that the robot has limited sensing capabilities, and the environment is represented in the so-called boundary place graph, which records a set of landmarks.

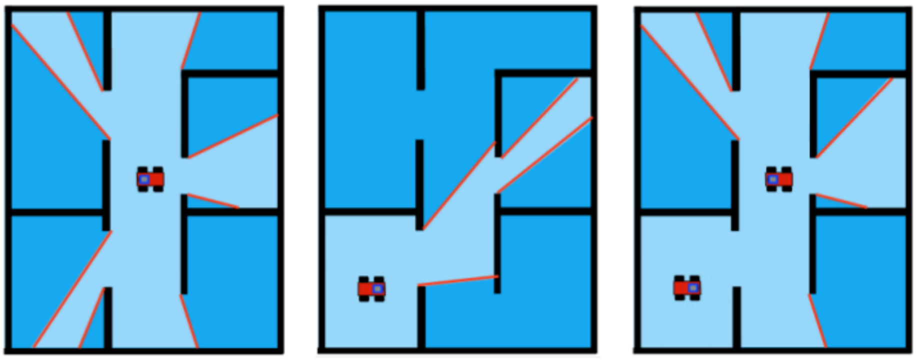
A method for robot navigation without the ability of sensing orientation but with the ability of sensing range discontinuities (gaps) is presented in [1]. In that work, a combinatorial structure called Gap Navigation Tree (GNT) which encodes a representation of the environment and records the shortest path to a set of landmarks is proposed. The GNT is dynamically built based on critical events detected with the robot's sensors, the GNT encodes trajectories from the current robot's position to any environment's region and these trajectories are optimal in terms of Euclidean distance. While the robot is moving, the GNT is updated for mantaining the information of the shortest distances from the current position regardless the lack of geometrical information. This work was designed for exploration and optimal navigation and the robot is modeled as a point.

The work in [1] has been a basis for several related works such as, [30] where a probabilistic model for the gaps in the GNT is presented. The global goals and gaps are chased probabilistically, which improves robustness given that the model deals with the presence of noise at sensor's measurements. The algorithm is adapted in such a way that it is suitable for selecting the next target in a SLAM system. The GNT was also extended to point cloud models in [31].

In [7], a wall following approach has been proposed for exploring a simply connected environment with a point robot. A data structure called cut ordering is proposed in that work. The point robot is able to identify whether the robot at its current location is touching an environment wall, a convex vertex [1], a reflex vertex [2] or whether it lies on the interior of the environment. Once the cut ordering representation is built, it is used to address a pursuit/evasion problem. In that work, there are different strategies depending on the robot's sensor capabilities, one of them is a pursuit strategy using the gap sensor defined in [1].

In contrast with the approach explained in this section, there are complex exploration strategies which try to determine the best sensing location (often called the next best view) to improve the exploration task efficiency. Often, motion planning is used for deciding the next best view. Many of the strategies that determine a sensing location for environment's exploration send the robot to the boundary between the area that has been perceived by the robot's sensors and the unknown region of the environment. Figure 2.6 shows an example of this. In the works developed in [32] and [14] some exploration strategies are described and compared in detail. In Section 2.4 we present some exploration strategies which actually use motion planning for determining the best sensing locations for future observations in a partially known environment.

[1] A convex vertex is a polygon vertex of an internal angle smaller than π.
[2] A reflex vertex is a polygon vertex of an internal angle greater than π.



(a) Mapa Pasillo (b) Mapa Oficina (c) Fusión de mapas

FIGURE 2.6: (a) and (b) represent the robot's visibility polygon and the boundaries between the current visibility region and the environment's non visible regions. They are represented as local maps, (c) is the already exposed environment obtained from two sensing locations and the boundaries betweent the already exposed environment and the regions which have not been seen yet, represented in a global map.

2.4 Motion planning for exploration

Most of the described previous works are focused in developing techniques for extracting relevant information from raw data and integrating them in a single model. However, typically, a motion planning for exploration is not developed. In order to improve the map building and exploration tasks, it is possible to use motion planning for determining the most convenient robot sensing location for future observations in a partially known environment. The objectives of motion planning in this problem might be reducing the exploration time, minimizing the observation steps or minimizing the traveled distance while an accurate environment's representation is built.

2.4.1 Greedy approach

The greedy approach evaluates a candidate observation position based on its gain. This gain is commonly defined as the amount of new environment's information expected by the robot at position p. Moreover, the planning horizon is just one step in the future, that is the reason why it is called greedy.

2.4.2 Gonzalez Banos approach

In [23, 33] a map building motion planning strategy is presented. This research work has shown that it is possible to find a function that reflects intuitively how the robot should explore the space.

In a simple scheme, the evaluation function should assign a greater value to the position that best fits the compromise between possible elimination of unexplored space and traveled distance. It is based on the computation of the next best view (NBV) and the use of randomized motion planning. In [33], the concept of the next best view is based mainly on two factors: Estimated cost to the next sensing location related with the distance between the current robot position and the sensing location, and the associated gain in terms of the new sensed area. Visibility type methods [34, 35] are used to estimate information gain. The evaluation strategy for a given candidate observation position p is given by:

$$g(p) = A(p) \cdot e^{-\lambda L(p)} \tag{2.1}$$

where $A(p)$ is an estimation of the visible area from p, $L(p)$ is the trajectory length connecting the robot and p, and λ is a weight associated with the traveling cost towards the robot's next position.

Overlapping between a local perception (local map) and the current global map is sufficient to merge a local map with the global one. The amount of overlapping between the maps is measured by using the size of the common perimeter between them.

2.4.3 Makarenko approach

The approach presented in [36] proposes an exploration strategy for map building and localization, with a grid-based modeling. The boundary method [37] is used for proposing future sensing locations which normally lie on the boundary between the known and unknown environment's regions. The exploration strategy uses an utility function that evaluates the next robot sensing location. This utility takes into account three elements: i) the information gain, ii) the distance to the sensing location (cost), and iii) the ability to locate the robot based on a covariance matrix.

Information gain utility. This utility is designed for boosting locations which offer large information gain. Information utility is obtained from an occupancy grid, and a cell may take two possible values, occupied space and empty space, $s(C) = OCC|EMP$.

In order to measure the amount of available information for each cell, the entropy H of the binary distribution $s(C)$ is computed as:

$$H(s(C)) = -\sum_{s(C)} P(s(C)) \log P(s(C)) \tag{2.2}$$

The utility of performing an observation from the goal location p_i is defined as the average entropy of the sensed region W_i around the goal location.

$$U_i^I = -\sum_{C \in W_i} \boldsymbol{H}(s(C)) \tag{2.3}$$

where the entropy $\boldsymbol{H}$ is calculated for each cell C within region W_i. The larger the entropy is, the less information about the region is actually available, and in consequence, it is more attractive for exploration.

Navigation utility. Long travels between locations reduce exploration efficiency. Navigation utility is used for making the shorter paths more attractive.

An instance of the navigation function family of algorithms [38] is used to calculate the cost of driving from the current position to the proposed destinations based on the information in the occupancy-grid map. The navigation function encapsulates the cost of reaching any location on the map from the current position. The navigation utility at a destination point p_i is simply the negative of the navigation function V at that location.

$$U_i^N = -V(p_i) \tag{2.4}$$

Utility of the ability to localize the robot. This utility is used to distinguish locations with different localization quality. Localization quality at a given place p_i is determined by the generated uncertainty on the configuration until that place is reached. Uncertainty is described as a covariance matrix of the state $\boldsymbol{P_{vv}}(p_i, t)$ at time t at location p_i. Entropy H for a Gaussian distribution from the robot's location to the goal location p_i is:

$$U_i^L = -H(P_{vv}^k) = -\frac{1}{2} \log \left((2\pi e)^n |P_{vv}^k| \right) \tag{2.5}$$

where n is the dimension of the robot's state and P_{vv}^k is the covariance of the robot's state after k observations.

The **total utility** of the target location is the weighted sum of the individual utilities.

$$U^{TOT} = \omega_I U^I + \omega_N U^N + \omega_L U^L \tag{2.6}$$

where relative weights ω_I, ω_L and ω_L depend of the main objective. They can be abruptly changed or continuously modified. The location with the maximum utility is selected as the next best sensing location.

$$x_* = \arg\max_x (U^{TOT}) \tag{2.7}$$

2.4.4 Amigoni approach

In [39], an exploration strategy based on information derived from the concept of relative entropy is presented. The function used for comparing sensing locations is:

$$f(p) = \frac{1}{N+A} \sum_{i \in \mathcal{A} \cup \mathcal{N}} \ln\left(\frac{\sigma_{unc,i}}{\sigma}\right) + N \ln\left(\frac{\sigma}{P}\right) + \sum_{i \in \mathcal{A}} \ln\left(\frac{\sigma}{\sigma_{p,i}}\right) + N \ln\left(\frac{2\pi c}{\sigma}\right) \tag{2.8}$$

where $N = |\mathcal{N}|$ is the number of new sensed points (obtained with a laser telemeter) from p, $A = |\mathcal{A}|$ is the number of already sensed points which are sensed again from p. $\sigma_{unc,i}$ is the standard deviation of the contribution of the error's measurements due to uncertainty at the robot's location. σ is the standard deviation of the sensor's correctness. P is the expected perimeter of the area to be added to the map. $\sigma_{p,i}$ is the previous standard deviation of the sensed point i. And c is the distance between the robot's position and p. In this case, the smallest value of $f(\cdot)$ corresponds to the best selected sensing location. The first term of equation 2.8 is the entropy's contribution of all the points sensed from p, the second term is the entropy's contribution of the new points sensed from p, the third term is the entropy's contribution of the already sensed points which are sensed again from p, and the last term is the entropy's contribution of the cost for reaching position p. In general, the first term increases the entropy, while the second and third terms reduce it.

In order to determine the best next sensing location, the exploration strategy merges the expected information adquired by the sensors and the robot's traveled distance towards the new position. The information gain is derived from the points that become visible from the sensing location and it is also derived from the reduction of uncertainty of the localization of the previously observed points. This exploration strategy is theoretically founded on *information theory*.

2.4.5 Simmons approach

The work presented in [40] deals with the problem of exploration and mapping of an unknown environment by multiple robots. A probabilistic grid is used to represent the environment. Each cell of the grid corresponds to one of three types: i) obstacle (probability of occupancy above a given threshold value po), ii) clear (probability below a threshold pc), and iii) unknown (it has never been sensed or its probability lies between po and pc). The distance between the robot and the frontier between the known and unknown environment's regions is used as an exploration cost. The number of unknown cells that fall within the robot's sensor range (for a possible next sensing location) is used as the information gain for sending the robot to that location. In [40], an utility value over candidate sensing locations is defined. The utility of a candidate sensing location is equal to its information gain minus its exploration cost, where the information gain is based on a probabilistic estimation of the frontier's size. A central computer tries to maximize the total utility by minimizing the overlapping between the areas that will be explored. To coordinate the robots (robots are related with sensing locations), the information gain is used.

2.4.6 Tovar approach

In [14, 41, 42], a function for evaluating future candidate sensing locations is presented. Some attributes that should have the sensing locations, and their evaluation process using an utility function are described.

For example, the evaluation function must prefer: future configurations with visual landmarks with high recognition probability, future configurations with the largest number of distinctive geometric primitives (e.g. corners of the environment), future configurations close to the boundaries between the known and unknown environment's regions (free borders), trajectories that can minimize the uncertainty at each location, trajectories which require the minimum number of sensing locations, and trajectories that minimize the total robot's traveling distance.

The work presented in [14], also addresses issues related to uncertainty in sensing and control, but with primary focus on the problem of planning optimal exploration strategies. A formalism for planning exploration strategies that optimize criteria such as information gain and uncertainty reduction is developed, where methods for extracting entities from sensor data (predefined landmarks or geometric features such as corners), and fusing these features into a common, global map are presented. The result is a method that autonomously explores its environment, optimizing the exploration at each stage and merging newly acquired sensor data with the existing map. The proposed utility function in [14] is constructed in such a way that it balances the desire to see as much of the as-yet-unseen environment's regions. At the same time, the utility function uses the landmark information and the current overlapping with the already sensed environment's region to guarantee good map registration and robot localization. This utility function, has a multiplicative form, which is helpul for discarding locations with small values in some parameters. Similar forms have been proposed in [41, 42]. This utility function has an important advantage, since it can be used for a single robot's sensing location or can be used for a path associated with a sequence of various sensing locations. The utility of a sequence is just the sum of the sub-goal sensing locations within the trajectory path. The function is:

$$\mathcal{T}_i = \sum_{i=1}^{m} \left(e^{(lv_i - sv_i)} \prod_{j=1}^{q_i} \left(\frac{e^{|\theta_j|}}{\sqrt{s_j} + 1} \right) \times \left(\frac{1}{n_i} \sum_{k=1}^{n_i} p_k + Ne_i \right) fmin_i(d_l) \right) \tag{2.9}$$

where i is the location of a sensing operation, m is the total number of sensing operations, q_i is the total number of robot stops to reach location i, lv_i is the length of the closest free edge at location i, s_i is the distance from the robot to the next possible location i, sv_i is the distance from the next possible location i to the closest free edge, j is the index for a robot configuration, θ_j is the orientation change to reach the next configuration j, p_k is the identification probability of landmark k at location i, k is the index for a given landmark, n_i is the number of landmarks inside a visibility region at location i, Ne_i is the number of corners inside robot visibility region at location i, $fmin_i$ is a function that penalizes location i, and d_l is the minimum distance from a full edge. The function $fmin_i$ is used to map the distance from an obstacle edge to a utility value, since for minimizing the effects of occlusion, it is desirable to maximize the distance to obstacles.

Computer vision is used to recognize landmarks using a Bayesian approach. A laser range finder is used to find straight lines in the environment (using least squares fitting), and lines obtained in consecutive sensing operations are fused by minimizing a partial Hausdorff distance. The final result of the exploration is a multi-representational map consisting of polygons and landmarks, and it includes a roadmap (backtracking graph) constructed from the trajectory followed by the robot.

2.4.7 Newman approach

In [43], an algorithm for feature-based exploration of an unknown environment is proposed. In that approach the candidate next robot's sensing locations (goals) are associated with all the geometrical features of the environment, then visibility computation is used to estimate the pertinence of a next robot's sensing location using a feature-based map. One major objective in [43] is to locally explore free regions. To achieve that objective, the following procedure is proposed. First, for each goal generated, sample points are regularly scattered around it at a constant radius β. Then, a circle of radius α centered on each sample is drawn. The final size of the sampling set is the available number of sample points for the goal that satisfy these conditions: i) each point has a clear line of sight to the goal, and ii) it has no line of sight to any other circle of radius α. The score $\eta \in [0, 1]$ for evaluating a goal location is set to be the ratio of the final and initial sizes of the sampling set. It is important to point out that the location of a feature is uncertain, and it is represented by a set of probability distribution functions. This is an approach that captures real world scenarios in a better way.

2.4.8 Feder approach

Another robot motion exploration strategy is presented in [44]. In that work, a metric for adaptive sensing that is defined in terms of the Fisher information is proposed. This metric is used to choose among discrete robot actions that given the current state, might locally maximize the information gained in the next sensor reading. As a result of applying that algorithm, the robot tends to explore selectively different objects in the environment, but this approach does not consider either path planning or obstacle avoidance.

2.4.9 Calisi approach

In [45], the problem of simultaneous exploration and searching is addressed. The action of chasing a target can be interrupted if a better target is detected during navigation. For example, in rescue missions it is desired to obtain information about possible victims as fast as possible before continuing with the exploration. This approach gives flexibility between deciding: (1) When decisions must be considered, and (2) How to choose among the current different targets. The basis of the structure of the exploration and searching strategies is inspired in the structure of the next best view (NBV) algorithm, and it is divided in two steps: i) detection, evaluation, and selection of targets, and ii) navigation towards the selected target.

2.5 Multi-robot exploration strategies

The use of multiple robots has many advantages in comparison of using a single robot. Robot teamwork allows achieving tasks faster. There are some works proposing multi-robot teamwork for exploration and map building [40, 46, 47]. In some works, the information gain and exploration cost are simultaneously considered for choosing the next sensing location for each robot of the team, [40] and [46] are some examples.

In [47], an occupancy grid is used for map representation, and the possible locations for the next sensing stage are selected among the cells found in the boundary between the known and unknown environment's regions. This approach is expanded in [22] by taking into account semantic information of the sensing locations. In [47] the authors propose strategies for multiple robots using segmentation of the environment for determining a goal for each robot. This segmentation improves the robots' distribution over the environment. In [14], a technique which allows one or multiple robots to explore efficiently and to build a model of the environment is presented. An utility function for measuring the quality of the proposed sensing locations is used together with an algorithm that generates and selects goal regions for the next sensing location. That work considers a team of robots with the same sensing and motion capabilities. However, some works have proposed multiple robot exploration with different sensing and motion capabilities for each robot. The work presented in [48] considers a team of robots of different size. If during exploration a robot is too large for navigating between obstacles and reaching a sensing location, then it asks smaller robots if they can do the task. Finally, in [49], the robots have one of two roles: navigator or cartographer. Navigators move randomly on the environment until they find a target sensing location for the cartographer, then, the cartographer moves to the target location. In the previously mentioned works, each robot has a specific role and it is predefined according to its type.

2.6 The Lopez optimal navigation strategy

The work presented in [9] proposes an approach for optimal navigation with respect to Euclidean distance. The GNT approach presented in [1] has been extended to a disc-shaped differential-drive robot placed into an unknown, simply connected polygonal region. The main result is a motion strategy that drives the robot to optimally navigate towards a landmark in the region. However, in [9], an exploration strategy to learn the GNT and encoding a landmark within it has not been developed. The robot's sensing capabilities and robot's motion controls are similar to the ones presented in this work.

This thesis is planned as the complementary exploration strategy that could be applied before the optimal navigation in terms of Euclidean distance developed in [9].

Chapter 3

Sensing Model

3.1 Robot's sensor and landmark

The robot considered in this work is a differential drive system shaped as a disc. The robot has a defined forward heading. The extremal left and right side robot's points are respectively called lp and rp. The robot has an omnidirectional sensor, which is used to discover the environment. The direction of the line tangent to the robot's boundary at rp is called rt. The direction of the line tangent to the robot's boundary at lp is called lt (See Figure 3.1). The omnidirectional sensor could be located at rp or lp at the beginning of the exploration task and it will remain at that position through the whole task. The omnidirectional sensor is able to track the preferential direction lt or rt depending whether the sensor is placed over lp or rp. The omnidirectional sensor reading in the particular forward robot heading direction corresponds to the preferential direction.

The omnidirectional sensor is also able to detect and track discontinuities in depth information (gaps). Hence, over the omnidirectional sensor, it is possible to build a gap detector, further refereed as the gap sensor. Additionally, the gap sensor can identify a landmark with specific characteristics detailed in Section 3.4. A large family of gap sensors is described in [50].

The gap sensor's behavior and the GNT evolution are fully described in [1], but is important to point out the most relevant details. The GNT that represents the environment encodes a different gap in each node, and it is built through the gap sensor's readings. While the robot is executing a motion primitive, the GNT is constructed incrementally. The omnidirectional gap sensor is able to identify any of four possible critical events related to the gaps. Those events represent a change in the structure of the GNT. In Subsection 3.1.1, a brief description of the GNT is presented.

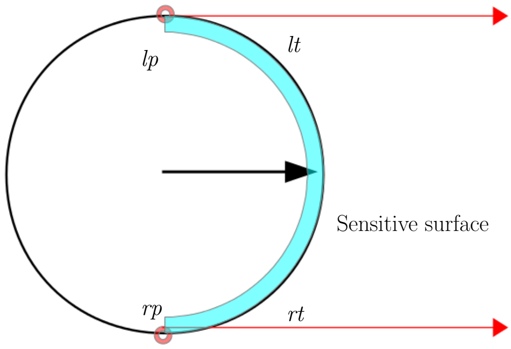


FIGURE 3.1: Representation of the robot's sensors.

The angles of the gaps are unknown due to the limited sensor's capabilities, but the sensor is able to maintain a cyclic angular order of them. Let $G(x) = [g_1, ..., g_k]$ denote the sequence of gaps as they appear in the gap sensor, when it is placed at $x \in E$. If x lies in the interior of E there is a cyclic order and statements such as $[g_1, ..., g_k] = [g_2, ..., g_k, g_1]$ can be made. If x lies in ∂E, then part of the sensor's view is obstructed by the boundary, and a linear ordering of gaps is obtained (see Figure 3.2). In summary, the gap sensor is able to detect and order the gap directions, the direction rt or lt and a visibility obstruction if the sensor is in contact with ∂E. This behavior allows the sensor to detect events such as alignments between the preferential directions rt or lt and any gap, or between one of the two preferential directions lt or rt and the wall (∂E) that is in contact with the omnidirectional sensor.

3.1.1 Gap Navigation Tree

We consider that the robot moves along any path $\tau : [0, 1] \rightarrow E$. Consider the information obtained from the gap sensor. For every $S \in [0, 1]$, a cyclic secuence $G(\tau(s))$ of gaps is observed. The GNT is a rooted tree defined as follows. Every nonroot vertex of the GNT is a gap that appears in $G(\tau(s))$ for some $s \in [0, 1]$. Every child vertex of the root is a gap in $G(\tau(1))$, and they are cyclically ordered around the root in the same way that they appear in $G(\tau(1))$. All remaining vertices (i.e., not the root and its children) in the GNT are gaps that appeared in $G(\tau(s))$ for some $s < 1$. The children of any nonroot vertex v are precisely the gaps that were merged to form v and are assumed to be ordered in the same way that they once appeared in the gap sensor. The GNT gaps are normally labeled consecutively using integers. For each new gap it is assigned the next unused integer in $\mathbb{N}$ to ensure uniqueness.

Initially, the GNT consists of a root vertex connected to one leaf vertex for every gap in $G(\tau(0))$. Each time t at which a change in $G(\tau(t))$ occurs corresponds to a *critical*

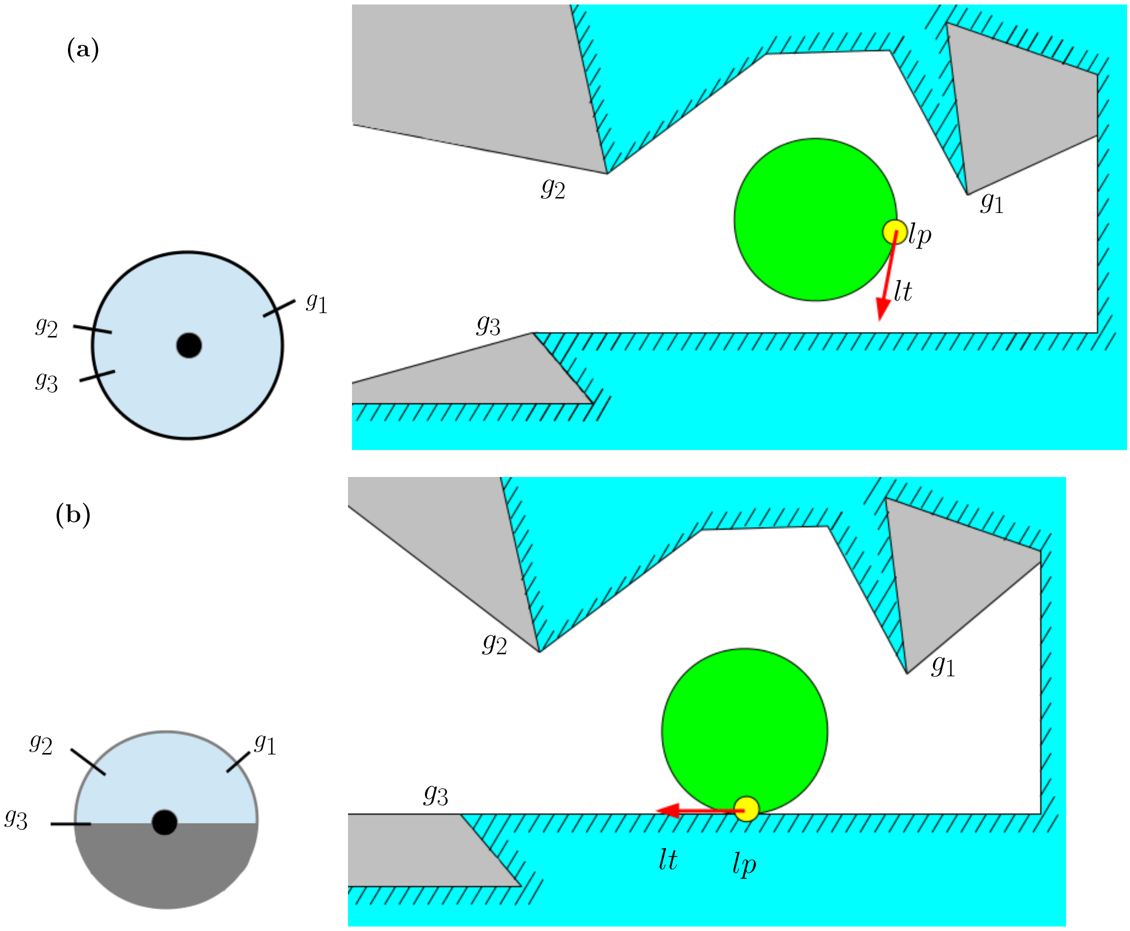
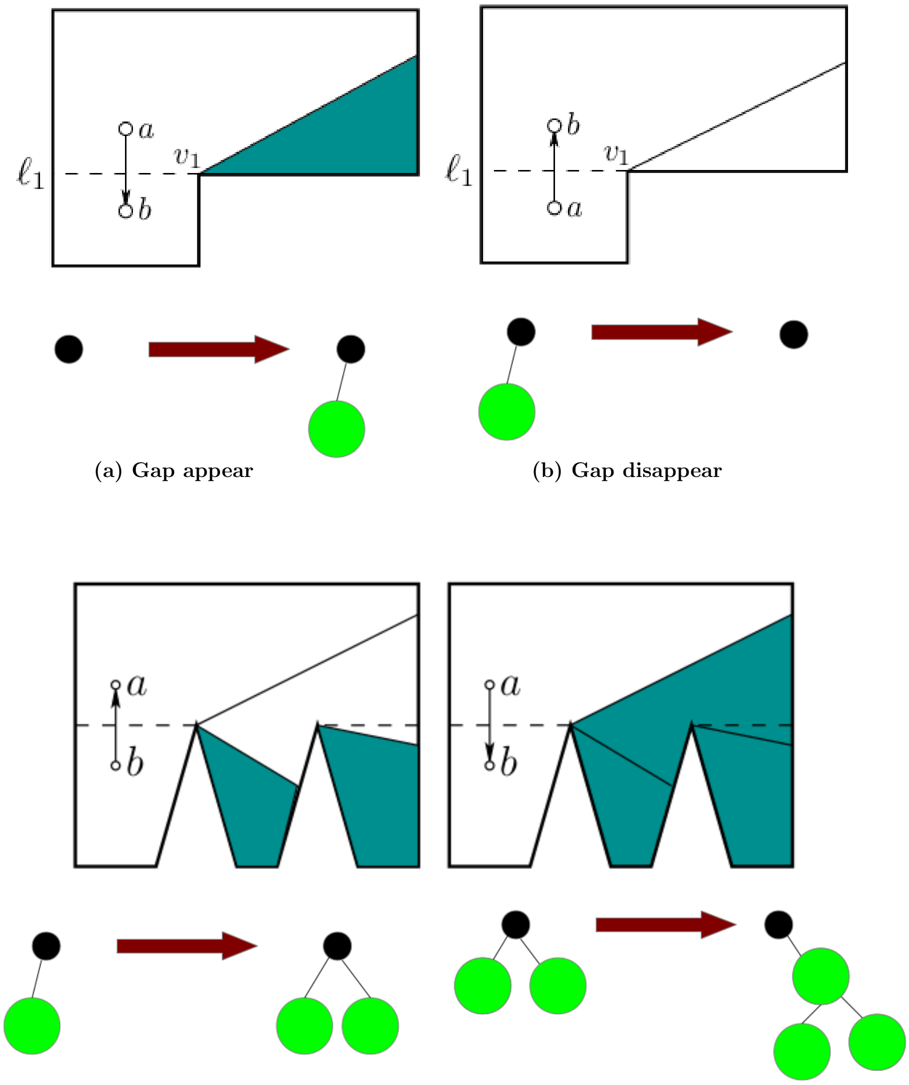


FIGURE 3.2: (a) The robot has its omnidirectional sensor over lp, it lies within E and gaps g_1, g_2 and g_3 are detected. On the left, the visibility region is shown with the angular position of the gaps, it is clear that statements such as $[g_1, g_2, g_3] = [g_2, g_3, g_1] = [g_3, g_1, g_2]$ can be made, (b) The omnidirectional sensor is in contact with ∂E, the same gaps are detected. On the left, the visibility region is partially obstructed by ∂E, it is shown as a gray colored region. The visibility obstruction can be used to establish a linear order of the gaps following a counterclockwise sense $[g_1, g_2, g_3]$, or a clockwise sense $[g_3, g_2, g_1]$.

event. This requires updating the GNT. There are for different kinds of critical events. In Figure 3.3 the critical events and the and the consequential changes in the GNT are illustrated. The dotted lines in the Figure are inflection rays (in the case of gap appear and gap disappear), and bitangent complements (in the case of gap merge and gap split). Inflection rays and bitangent complements represent a fundamental part of the geometric interpretation of the critical events, and they are detailed in [1].



(a) Gap appear **(b) Gap disappear**

(c) Gap split **(d) Gap merge**

FIGURE 3.3: (a) The robot moves from position a to b, when the robot crosses the inflection ray, a region of the environment is occluded, therefore a new gap appears. A new node is added to the root, (b) The opposite case, the robot moves from a to b, when the robot crosses the inflection ray the gap disappears. Its corresponding node is removed, (c) In position a the sensor detected a gap, then the robot moves to b, when the robot crosses the bitangent complement the gap is divided into two gaps. The original node is removed and replaced by two gaps, (d) In position a the sensor detected two gaps, then the robot moves to b, when the robot crosses the bitangent complement the gaps merge into a new gap. The original gaps become children of a new node that is connected to the root.

- *A new gap g* appears: A node g is added as a child of the root, while preserving the cyclic ordering from the gap sensor.

- *Gaps g_1* and *g_2 merge into g: Nodes g_1 and g_2 become children of a new node, g, which is added as a child of the root and preserving the cyclic ordering.*

- A gap g *disappears: The node g, which must be a leaf, is removed.*

- Gap g *splits into* g_1 and g_2: If g is a leaf node, then g_1 and g_2 become new nodes; otherwise, they already exist as children of g. Both g_1 and g_2 are connected to the root, preserving the cyclic ordering and removing g.

If any leaf vertex has the potential to split, then the GNT is incomplete because it could expand, some gaps split and others simply disappear. The gaps that disappear are called *primitive* (their corresponding nodes in the GNT are also called primitive and they are removed when reached). If all the leaf nodes of the GNT are primitive, then the GNT is said to be *complete*. The GNT completeness has a geometric interpretation detailed in [1].

3.2 A Tactile Bumper

The robot's frontal periphery is contact sensitive (See Figure 3.1). In a real robot it could be implemented, for instance with a piezoelectric sensor. The sensitive surface model is able to distinguish whether there exists contact on a single point or more than one, and it also distinguishes whether the point rp or lp is in contact with a wall. The particular case of both points rp and lp being simultaneously in contact is not considered, as it only would happen in a narrow corridor, of exactly the same width as the robot, and that scenario is considered a degenerated case.

3.3 The observation vector

With the sensor capabilities previously defined it is possible to define an observation vector which includes all the possible observations that are able to trigger a specific control.

Six binary sensor observations constitute the observation vector: (1: lp) the robot is touching ∂E with point lp. (2: rp) the robot is touching ∂E with point rp. (3: sc) the robot is touching ∂E with a single point within the sensitive surface (this point might be either lp, rp or any other point within the sensitive surface). (4: bc) the robot is touching ∂E with two or more points within the sensitive surface (one of them can be either lp or rp). (5: *aligned*) direction rt is aligned with the edge of the polygonal region that point rp is touching, or point rp is touching a reflex vertex and the preferential direction rt is aligned with the first polygonal edge, measured in clockwise sense starting from direction rt (see Figure 3.4); or direction lt is aligned with the edge of the polygonal region that point lp is touching, or point lp is touching a reflex vertex and the preferential direction lt is aligned with the first polygonal edge, measured in counterclockwise sense starting from direction lt; (6: o) the omnidirectional sensor is located at point lp (0) or the omnidirectional sensor is located at point rp (1) at the beginning of the exploration task (the gap sensor will remain located over that point through the whole task). Thus, the observation vector is: $ye_i = \{lp, rp, sc, bc, aligned, o\}$

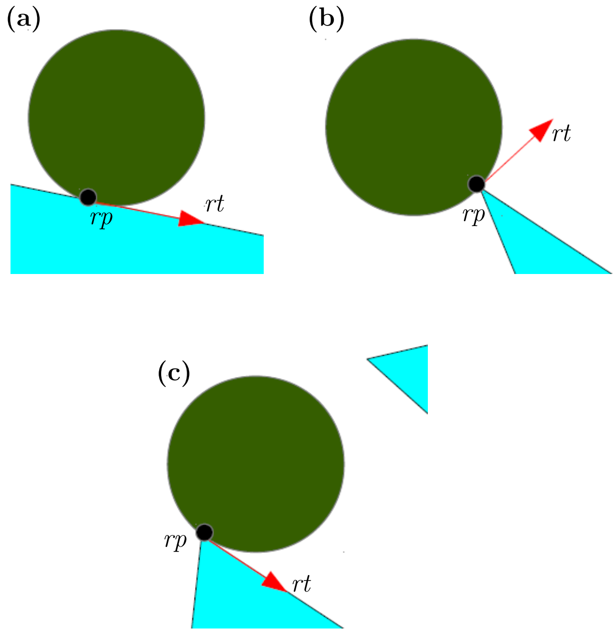


FIGURE 3.4: (a) The direction rt is aligned with it the edge of ∂E that point rp is touching. Therefore, $aligned = 1$, (b) Point rp is touching a reflex vertex, and the preferential direction rt is not aligned with the first polygonal edge in clockwise sense starting from direction rt that is in contact with rp. Therefore, $aligned = 0$, (c) Point rp is touching a reflex vertex, and the preferential direction rt is aligned with the first polygonal edge in clockwise sense starting from direction rt that is in contact with rp. Therefore, $aligned = 1$. If the omnidirectional sensor is located at lp, an alignement is just the left symmetric case.

The set of all 64 observation vectors can be partitioned by letting x denote any value to obtain:

$$
\begin{aligned}
ye_1 &= (0,0,0,0,x,x)\\
ye_2 &= (0,1,1,0,1,1)\\
ye_3 &= (1,0,1,0,1,0)\\
ye_4 &= (x,x,0,1,x,1)\\
ye_5 &= (x,x,0,1,x,0)\\
ye_6 &= (0,1,1,0,0,1)\\
ye_7 &= (1,0,1,0,0,0)\\
ye_8 &= (0,0,1,0,x,0)\\
ye_9 &= (0,0,1,0,x,1)
\end{aligned}
$$

The meaning of each observation vector is the following:

- ye_1 *No contact*: This observation might only happen at the beginning of the exploration if the robot lies completely in the interior of E, such that there is no contact sensed.
- ye_2 *Single contact* with rp: The omnidirectional sensor is positioned at rp, there is single contact detected at that point, and the preferential direction rt is aligned with the polygonal edge that point rp is touching.
- ye_3 *Single contact* with lp: This observation is analogous to Single contact rp, it is the left symmetric case.
- ye_4 *Multicontact, sensor at* rp: The omnidirectional sensor is located at point rp and there is a multicontact detected (rp might be a contact point), while the omnidirectional sensor is placed at rp. The robot's sensitive surface is touching more than one point of ∂E, the contact might be with any combination of edges or reflex vertices of E.
- ye_5 *Multicontact, sensor at* lp: This observation is analogous to Multicontact rp, it is the left symmetric case.
- ye_6 *Reflex vertex* rp: The omnidirectional sensor is located at point rp, there is single contact between point rp and a reflex vertex of the polygonal environment, and the preferential direction rt is not aligned with the first polygonal edge, measured in clockwise sense starting from direction rt.
- ye_7 *Reflex vertex* lp: The omnidirectional sensor is located at point lp, there is single contact between point lp and a reflex vertex of the polygonal environment, and the preferential direction lt is not aligned with the first polygonal edge, measured in counterclockwise sense starting from the reflex vertex.

- ye_8 *No-single contact* at lp: The omnidirectional sensor is positioned over lp and the robot is touching an edge or a reflex vertex of ∂E with a single point different to lp.

- ye_9 *No-single contact* at rp: This observation is analogous to No-single contact at lp, it is the right symmetric case with the omnidirectional sensor positioned at rp.

3.4 Landmark Encoding

The gap sensor described in this chapter is able to identify a static disc-shaped landmark Λ of the same radius r as the robot lying on the interior of E, $\Lambda \subset E$.

The landmark is said to be detected if Λ is visible at least partially from the location of the gap sensor, $\Lambda \cap V_{pol} \neq \emptyset$, where V_{pol} is the current omnidirectional sensor's visibility polygon. Once the landmark is detected, it is encoded as a special GNT node connected to the corresponding node associated with the gap generated by the reflex vertex that is occluding it[1], or if the whole landmark is visible from the current sensor's location then it is directly connected to the root. The landmark is encoded as a special node because it is not encoding a gap, therefore it cannot experience the GNT critical events.

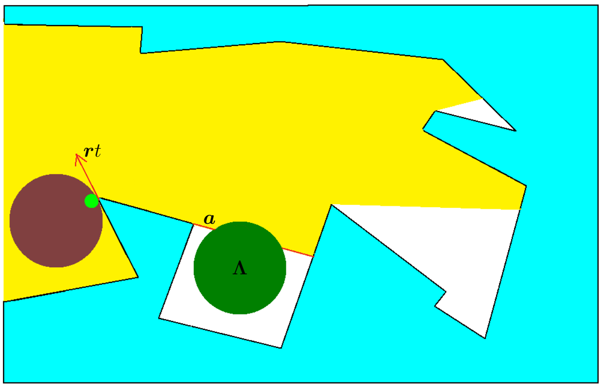


FIGURE 3.5: Λ is partially visible, it is occluded by the reflex vertex that generates gap a.

A simple example of the landmark encoding process is presented. The robot is represented as a dark gray (brown) disc with its corresponding omnidirectional gap sensor as a small light gray (light green) disc positioned over the robot's boundary. The preferential direction rt is presented as a small arrow over the robot. The visibility polygon

[1]A regular GNT node is considered a leaf node when it does not have children with the exception of a landmark node, which is a special kind of node. Therefore, a leaf node might have a landmark node as a child without losing its leaf status.

V_{pol} is colored in light gray (yellow). The obstacles are shown in medium gray (blue). The non-visible region at the current sensor's location (the region outside the visibility polygon) is shown in white. The landmark is a dark gray (dark green) disc labeled as Λ.

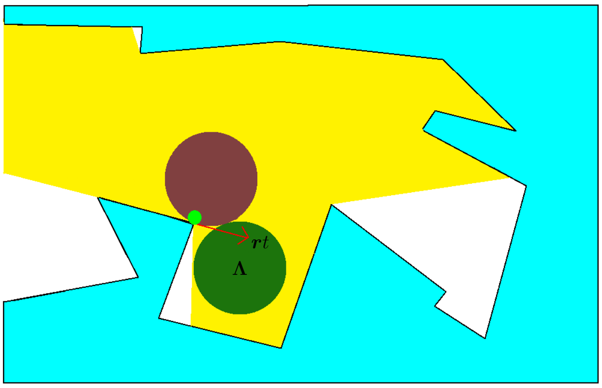


FIGURE 3.6: Λ is completely visible.

Figure 3.5 shows the case when Λ is partially visible from the gap sensor for the first time, it is encoded as a child of the corresponding GNT node. While the robot continues its movement following the environment's boundary, the whole landmark becomes visible (Figure 3.6), in this case Λ is encoded as a root's child.

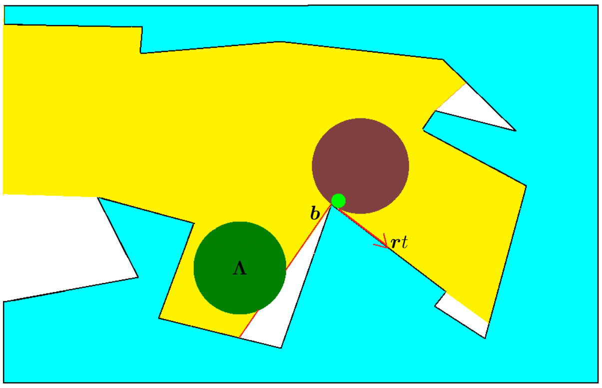


FIGURE 3.7: Λ is partially occluded by the refelex vertex that generates a new gap b.

Figure 3.7 shows a reflex vertex which is partially occluding the landmark. In that case, Λ is encoded as a child of the node associated with the new gap b generated by the occluding reflex vertex. In summary, every time Λ becomes completely visible, it is encoded as a root's child, and every time there is a partial occlusion, it becomes a child of the node associated with the gap generated by the reflex vertex that is occluding Λ.

If Λ is encoded as a child of a non-root node, its parent node might experience changes due to the GNT critical events. A merge event that includes the landmark's parent will not affect the landmark codification, Λ will remain connected to its parent. A gap disappearance related with the landmark's parent cannot happen, because the whole landmark will become visible, and the landmark node will be connected to the root before the gap disappears. Finally, if the landmark's parent is related with a gap split event, the landmark node will simply change its connection to one of the resulting split gaps (just one of the reflex vertices that generate the gaps will be occluding it). In conclusion, the GNT critical events that may affect landmark's parent node does not represent an issue for the landmark encoding process.

Chapter 4

Motion Model

4.1 Motion Model

The differential drive robot has two independent wheels, which are placed at the extremal left and right robot's points, under lp and rp respectively. Each wheel has its own motor.

The robot is arbitrarily allowed to execute five motion primitives as shown in Figure 4.1.

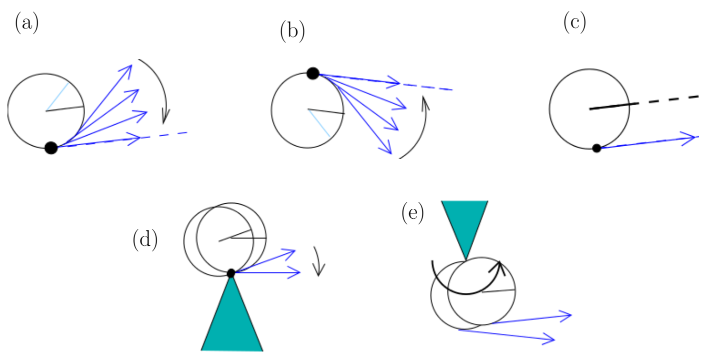


FIGURE 4.1: The motion primitives: (a) Clockwise rotation in place, (b) Counterclockwise rotation in place, (c) Straight line motion, (d) Clockwise rotation w.r.t. rp, (e) Counterclockwise rotation w.r.t. lp.

Let the angular velocity of the right and left wheels be ω_l and ω_r respectively, with saturated values ω_l, $\omega_r \in \{-1, 0, 1\}$. The robot's controls are defined by the vector $u = \{\omega_l, \omega_r\}$.

Five motion primitives are generated by the following controls:

$u_1 = (1, 1)$	forward straight line motion
$u_2 = (1, -1)$	clockwise rotation in place
$u_3 = (-1, 1)$	counterclockwise rotation in place
$u_4 = (1, 0)$	clockwise rotation w.r.t. point rp
$u_5 = (0, 1)$	counterclockwise rotation w.r.t. point lp

Executing the controls defined above, the robot explores the environment through wall following. If the omnidirectional sensor is placed at rp then the robot follows the environment's boundary ∂E in counterclockwise sense, and if the sensor is placed at lp then the robot follows ∂E in clockwise sense.

Chapter 5

The Exploration Automaton

A finite-state machine (FSM) is defined as a mathematical model of computation. It is conceived as an abstract machine that can be in one of a finite number of states. The machine is in only one state at a time, it can change from one state to another by a triggering event or condition called *transition*. A FSM is defined by a list of its states, and the triggering condition for each transition. A special kind of FSM is the Moore machine which includes outputs associated with every state, where these outputs depend exclusively of the current state and they do not take into account the input. According to the presented definition, *it is possible to represent the whole exploration strategy as a Moore machine.*

The FSM M represents the robot's planner which is a complete exploration strategy that allows the robot to discover the largest region of the environment E. Consequently the robot will find the landmark if possible, or it will declare that an exploration strategy to find it does not exist.

M includes a motion policy and manages GNT queries and updates (including the landmark encoding process as a part of the GNT evolution). The motion policy is a mapping from observations to controls (see Section 5.1). Note that the motion policy is only a part of the whole exploration strategy.

The task is not finished until a stop condition for exploration is met. This condition is not included in the motion policy because it requires topological information of the environment that is not given by the current sensor readings. This information is given by the GNT built during the robot's motion. As it is detailed in [1], the exploration task for a point robot ends when all the environment has been seen (regardless of whether the landmark Λ has been found or not), it happens when *all the leaf nodes of the GNT are labeled as primitive ones.*

The stop exploration condition for a disc robot is similar to the one for a point robot, but includes the additional issue of gaps that never disappear. Note that due to the robot's dimensions, there may be some unreachable environment's regions yielding those gaps. Consequently, an algorithm called *local exploration* has been developed for dealing with this issue. See Algorithm 1, this algorithm is part of the exploration strategy.

M is formally defined as a sextuple (Σ,S,s_0,δ,Γ,ω), where:

- Σ is the input alphabet (a finite, non-empty set of symbols). In M, Σ is defined by both the quantized observation vector ye_i and an additional query input given by the GNT, needed for determining whether the stop condition is met.
- S is a finite, non-empty set of states, where every state represents the selection and execution of a robot's motion primitive with the exception of two states: the initial state when the robot is not executing a primitive yet and the end state, in which the robot has finished the exploration task.
- s_0 is the initial state, in which the exploration task begins.
- δ is the state-transition function: $\delta : S \times \Sigma \to S$. In M, given an observation and the current state, δ defines which will be the new state. It is important to note that δ is a partial function. For example, $\delta(q, x)$ does not have to be defined for every combination of $q \in S$ and $x \in \Sigma$. Actually the set of allowed combinations is well established in the motion policy of Section 5.1.
- Γ is the output alphabet (a finite set of symbols), it is defined as the signals given to the motors for executing a given control u.
- ω is the output function: $\omega : S \to \Gamma$, each state provides a specific output signal defined on Γ.

A graphical representation of M is shown in Figure 5.1. There are seven states, one of them is the initial state when no motion primitive has been executed. There is an end state which establishes the GNT completeness. The task has been achieved, so no motion primitive is applied and the robot stops its movement. The other states represent the execution of the motion primitives defined in Chapter 4. All the links in Figure 5.1 are labeled with the corresponding observations defined in Chapter 3, with the exception of the GNT link, which represents a query to the GNT asking whether all the leaf nodes are marked as primitive ones.

GNT queries are done in states CCW Rotation in Place, CW Rotation in Place, and Straight Line Motion (see Figure 5.1). Given that, the GNT might change because the

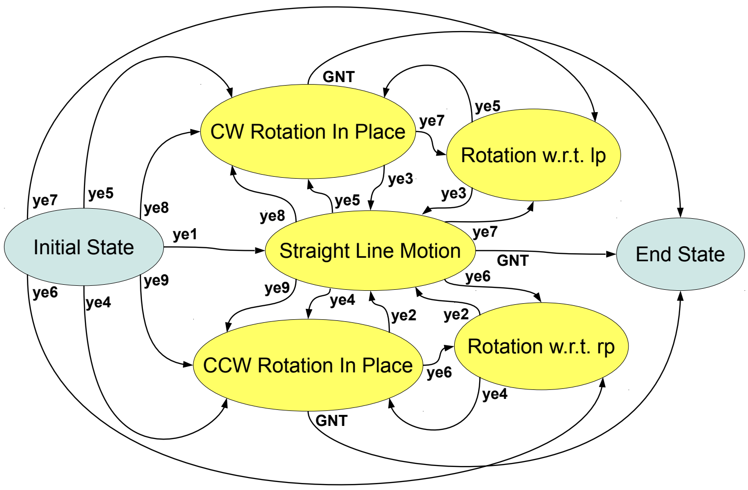


FIGURE 5.1: The finite-state machine that represents the exploration strategy.

occurrence of critical events, while the robot is execution one of these motions. The queries are required to decide whether or not the exploration is terminated (i.e. the stop condition is met). The *Local exploration* algorithm might be triggered in states CCW Rotation in Place or CW Rotation in Place. The *Local exploration* algorithm updates the labels of the gaps in the GNT.

5.1 Motion Policy

The motion policy is based on the paradigm of avoiding the state estimation to carry out two consecutive mappings: $y \to x \to u$, that is from observation y to state x and then to control u, but instead of that, there is a direct mapping $y \to u$.

Let γ be a mapping function, the motion policy can be established by $\gamma : \{0,1\}^6 \to \{-1,0,1\}^2$, then the function is expressed as $\gamma\,(ye_i) = (\omega_l, \omega_r) = u_j$. The motion policy is:

- $\gamma(ye_1 \vee ye_2 \vee ye_3) = u_1$
- $\gamma(ye_5 \vee ye_8) = u_2$
- $\gamma(ye_4 \vee ye_9) = u_3$
- $\gamma(ye_6) = u_4$
- $\gamma(ye_7) = u_5$

In which $\vee$ means "or".

The previous list summarizes the complete relationship between the controls and the observations given by the sensors.

5.2 The Local Exploration Algorithm

The configuration space restrictions for a disc robot might cause the presence of unreachable environment places. Those places might yield gaps that cannot disappear regardless of the robot motion. Once the point lp or rp lies on ∂E it is possible to identify the observations that represents the presence of gaps that do not disappear. Those observations are: $ye_4^R = (0, 1, 0, 1, x, 1)$ or $ye_5^L = (1, 0, 0, 1, x, 0)$. They are special cases of ye_4 and ye_5 observations respectively, when the corresponding observation happens (depending if the contact point is rp or lp) the algorithm is triggered. The algorithm uses information from the GNT, and the algorithm ends after the nodes encoding gaps generated by vertices within an unreachable region are labeled as primitives. The algorithm uses the property that states *the change from cyclic to linear gap ordering when the gap sensor is touching the wall according to the model detailed in [1].*

When *Local exploration* begins after ye_4^R happens, a linear list called *init-list* is created, it contains the current gaps read by the sensor and the preferential direction rt starting from the sensor's obstructed visibility region following a counterclockwise order. According to the motion policy, motion control u_3 is executed at the same time. In consequence, the gap sensor will be changing its position and critical GNT events might happen. The changes done to the GNT due to critical events are also applied to *init-list*, with the exception of the gap appearance event which does not represent an issue (the new GNT node has already the primitive label). The position of the rt direction might change while the gap sensor is moving, therefore the position of rt in *init-list* is also updated. While u_3 is executed, the gap sensor's reading gives a cyclic order of the gaps. The importance of *init-list* is that it gives a linear order of the current gaps and rt although there is not a sensor's visibility obstruction. When motion control u_3 ends (after observation ye_2 or ye_6 happens), *init-list* is a linear list of the current gaps and the current rt direction, starting from the sensor's obstructed visibility region at the beginning of the algorithm following a counterclockwise order. At this time, point rp is in contact with ∂E and a new linear list called *end-list* is created. *end-list* contains the current gaps read by the sensor and the rt direction ordered in counterclockwise sense, but it starts at the current sensor's obstructed visibility region. Linear lists *end-list* and *init-list* are different because they were created at different positions.

When *Local exploration* begins after ye_5^L happens, an analogous symmetric procedure begins. *init-list* is created in the same way that the other case, the difference is that it contains lt direction instead of rt and the list is created following a clockwise order of the gaps instead of a counterclockwise order. The motion control executed at the same time that the algorithm is u_2 and it ends after observation ye_3 or ye_7 happens. *end-list* contains the current gaps read by the sensor and lt direction following a clockwise order. Once *end-list* is created, auxiliary lists G_1 and G_2 are also created and they contain specific subsets of *init-list* and *end-list* respectively. Another list called $G_\cap$ is created and it contains only those gaps which are contained on both lists, these are the gaps generated by reflex vertices within the unreachable region, and they must propagate the primitive label to their offspring on the GNT. This process is explained with more detail in proof 5.2. In the algorithm, $init_f$ is the first element of *init-list*, $init_{rt}$ is the element containing the rt direction (if the gap sensor is placed over rp), or $init_{lt}$ is the element containing the lt direction (if the gap sensor is placed over lp). In *end-list*, end_l is the last element of *end-list*, end_{rt} is the element containing the rt direction and end_{lt} is the element containing the lt direction. As seen in the algorithm, these elements are used to establish the subsets of *init-list* and *end-list* needed to create auxiliary lists G_1 and G_2.

Lemma 5.1. *The exploration strategy modeled by the Moore machine M guarantees that all leaf gaps (i.e. gaps encoded as leaf nodes in the GNT) are labeled as primitive gaps, by executing Algorithm 1 (*local exploration *algorithm) at each time that observation y_4^R or y_5^L occurs. Algorithm 1 labels as primitive gaps, the gaps that do not disappear, given that those gaps are generated by reflex vertices located within an unreachable region.*

Proof. The gaps that do not disappear are handled by Algorithm 1. If the robot is touching ∂E with point rp then the G_1 list includes all the gaps belonging to the open interval between the rp point starting position at the beginning of Algorithm 1 and the rt direction at the end of Algorithm 1. In this interval the order of the gaps is established in counterclockwise sense. Moreover, the G_2 list includes all gaps belonging to the open interval between rt direction at the end of the algorithm and the rp position at the end of the algorithm. In this other interval the order of the gaps is also established in counterclockwise sense. The intersection between G_1 and G_2 includes only the gaps that lie between the original rp position and the current one in counterclockwise sense. Those gaps are generated by reflex vertices located within the unreachable region. Observation y_4^R or y_5^L detects an unreachable region. The region is unreachable because the robot's bumper has touched ∂E at two points. During the robot rotation in place, the omnidirectional sensor moves from a point touching ∂E to the other, hence all gaps within the unreachable region are considered. Due to the possible split and merge critical events

Algorithm 1 Local Exploration Algorithm

Input: GNT, current observation: ye_i.
Output: updated GNT.
if $ye_i = ye_4^R$ (u_3 is executed) **then**
  1. *init-list* $\leftarrow$ Current gaps and rt direction starting from the sensor's obstructed visibility region following a counterclockwise order;
  while $(ye_i \neq ye_2)$ **or** $(ye_i \neq ye_6)$ **do**
    if $GNT\text{-}event$ = **true then**
      if *critical-event* $\neq$ *gap-appear* **then**
        2. Apply the update suffered by the root's children nodes of the GNT to the corresponding gaps in *init-list*;
      end if
    end if
    3. Update the position of the rt direction in *init-list* according to the current angular counterclockwise order in the sensor reading;
  end while
  4. *end-list* $\leftarrow$ Current gaps and rt direction starting from the sensor's obstructed visibility region following a counterclockwise order;
  5. $G_1 \leftarrow \{x \in \textit{init-list} \mid init_f \leq x < init_{rt}\}$;
  6. $G_2 \leftarrow \{x \in \textit{end-list} \mid end_{rt} < x \leq end_l\}$;
else if $ye_i = ye_5^L$ (u_2 is executed) **then**
  7. *init-list* $\leftarrow$ Current gaps and lt direction starting from the sensor's obstructed visibility region following a clockwise order;
  while $(ye_i \neq ye_3)$ **or** $(ye_i \neq ye_7)$ **do**
    if $GNT\text{-}event$ = **true then**
      if *critical-event* $\neq$ *gap-appear* **then**
        8. Apply the update experienced by the root's children nodes of the GNT to the corresponding gaps in *init-list*;
      end if
    end if
    9. Update the position of the lt direction in *init-list* according to the current angular clockwise order in the sensor reading;
  end while
  10. *end-list* $\leftarrow$ Current gaps and lt direction starting from the sensor's obstructed visibility region following a clockwise order;
  11. $G_1 \leftarrow \{x \in \textit{init-list} \mid init_f \leq x < init_{lt}\}$;
  12. $G_2 \leftarrow \{x \in \textit{end-list} \mid end_{lt} < x \leq end_l\}$;
end if
13. $G_\cap \leftarrow G_1 \wedge G_2$;
for every gap $g_i \in G_\cap$ **do**
  14. Label node g_i in the GNT as a primitive node;
  15. Propagate the primitive label to the offspring of g_i;
end for
16. return updated GNT;

between these gaps, the primitive label of such gaps is propagated to all the offspring of them in $G_{\cap}$. Each time that observation y_4^R or y_5^L occurs, the *local exploration* algorithm is executed. Hence, all gaps encoded as leaf nodes (called leaf gaps) in the GNT are labeled as primitive gaps. □

When the robot is touching ∂E with point lp and the omnidirectional sensor is placed at lp the proof is analogous, it is just the symmetric case.

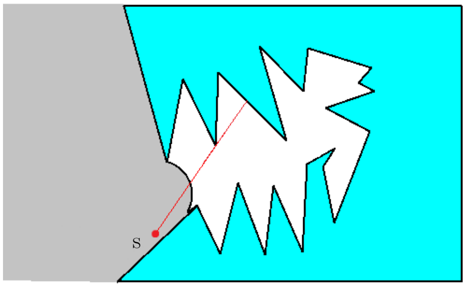


FIGURE 5.2: This figure shows region R_{na} in white and region R_a in dark grey. The regions are divided by the arc of circle trajectory followed by the omnidirection sensor during a robot's rotation in place. The figure also shows a light source s with a ray of light which goes from R_a to R_{na}.

Lemma 5.2. *The robot is capable of covering (observing) the largest possible portion of the environment, by executing* local exploration *algorithm at each time that observation y_4^R or y_5^L occurs.*

Proof. The omnidirectional sensor trajectory during the rotation in place motion is an arc of circle, which divides the environment's interior in two regions, named accessible region R_a and unaccessible region R_{na}, such that $R_a \cap R_{na} = \emptyset$. The boundary between those regions is the arc of circle described by the rotation, which depends of the robot's radius. The omnidirectional sensor is unable of penetrating deeper in the unreachable region due to the space configuration restrictions, therefore, the arc of circle is clearly the boundary between both regions. Refer to Figure 5.2. It is clear that every ray of light emerging from any source $s \in R_a$ which is able to intersect R_{na} must cross the regions' boundary as seen in Figure 5.2. If the visibility polygon of s includes a portion of R_{na} then every ray of light emerging from R_a to R_{na} must cross the regions' boundary. Therefore every single ray of light traveling from any point $x \in R_a$ to R_{na} must cross the regions' boundary. Hence, an omnidirectional sensor following the arc of circle trajectory guarantees observing the largest possible region of R_{na}. Each time that observation y_4^R or y_5^L occurs the *local exploration* algorithm is executed. The result follows. □

Theorem 5.3. *The exploration strategy modeled as a Moore machine M guarantees exploring all the environment or the largest possible region of it, and it also guarantees that the exploration task terminates. Additionally, the robot is able to find the landmark or to declare that an exploration strategy to find the landmark does not exist, for the connected component of the collision-free subset of the configuration space $\mathcal{C}$ where the robot lies.*

Proof. Since the environment is simply connected, a wall following strategy is enough for exploring all the environment for a point robot due to the absence of internal obstacles (generating more than one class of homotopic paths). For a disc robot, the gaps that are generated by reflex vertices located in reachable regions are labeled as primitive gaps, since the robot is able to reach the reflex vertices generating those gaps, then these gaps disappear. If there are unreachable regions, where some gaps do not disappear regardless the sensor's motions, then *local exploration* algorithm is executed. Lemma 5.1 guarantees that all leaf gaps are labeled as primitive ones, that is the stop condition for the exploration task. Hence, the exploration task terminates. Lemma 5.2 guarantees that the robot discovers the largest possible region of the environment. Hence, if the collision free sub-set of the configuration space $\mathcal{C}$ is simply connected then the landmark is found. If the collision free sub-set of the configuration space $\mathcal{C}$ has several connected components then the landmark might or might not be found. Again, by Lemma 5.2 the robot observes (discovers) the largest possible part of the environment, therefore when the landmark is not found, there does not exist a robot exploration strategy to find the landmark, for the connected component of the configuration space where the robot lies. $\square$

Chapter 6

Implementation

6.1 Environment 1

The whole method has been implemented and simulations' results are included. Simulations were programed using the C++ programming language. The most important libraries used are the Standard C++ Library and the 5.2 version of LEDA library http://www.algorithmic-solutions.com/leda/, which is used due to its internal computational geometry algorithms, data structures and its friendly graphical functions for animations and simulations. The GNT was performed using linked lists and our own GNT node class called *node*.

In all the following figures, the already explored environment is shown in white. The current visibility robot's region is shown in light gray (yellow), the environment regions which have not been discovered yet are shown in dark gray. The obstacles are shown in medium gray (blue). The robot is represented with a black disc, the omnidirectional sensor is a point over the robot's boundary. A small arrow over the robot is used to show the preferential sensor direction rt. The landmark is represented by a medium gray disc (green). In the GNT, the primitive leaf nodes are shown as squares (yellow), the landmark node is a triangle (blue), and the non-primitive nodes are shown as circles (green). The gap id for every GNT node is also shown (as consecutive integers), the id for the root node is 0.

Figure 6.1 shows the robot executing control u_1 that yields a straight line motion primitive, the robot initial position lies in the interior of E, it moves forward until a contact with ∂E is detected, it is the only case where the robot does not follow the environment's boundary.

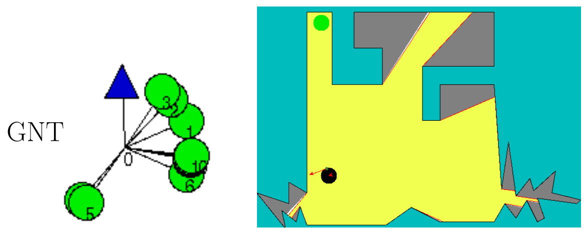


FIGURE 6.1: The robot is executing the straight line motion primitive. The corresponding GNT is shown, the circles represent non primitive nodes (for illustrative purposes, they the same orientation that the gaps they represent) and the triangle is the encoded landmark.

Figure 6.2 shows a partial landmark occlusion. It occurs when the robot is following an environment's edge, then the landmark is encoded to the corresponding gap in the GNT. Before that, the landmark was connected directly to the GNT root because it was completely visible as described in Section 3.4.

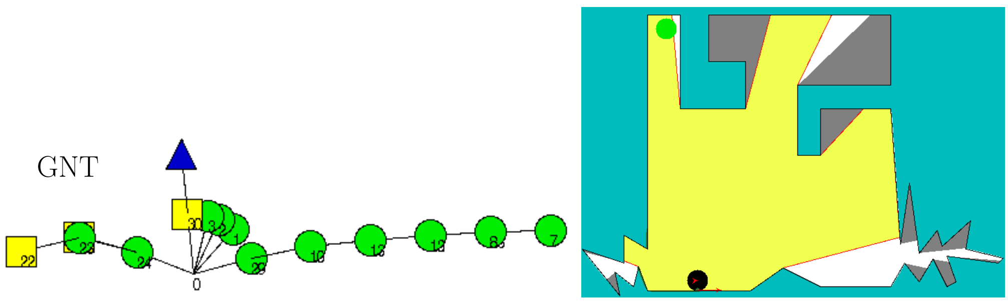


FIGURE 6.2: The landmark is partially occluded, the current GNT is presented and it shows that the landmark is encoded in the GNT (the primitive leaf nodes are shown like squares).

Figure 6.3 shows the execution of control u_4 that yields a clockwise rotation w.r.t. point rp motion primitive, the robot is touching a reflex vertex of ∂E. In order to mantain contact with the vertex, the robot must rotate w.r.t rp until rt is aligned with the environment's edge.

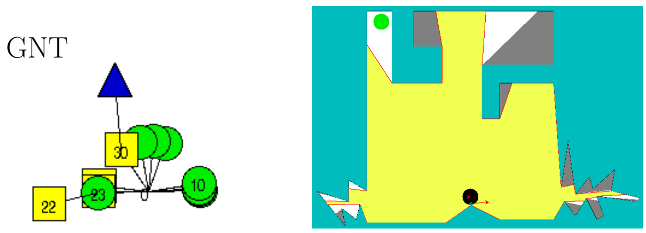


FIGURE 6.3: The robot is performing rotation w.r.t.rp because it is touching an environment's reflex vertex.

Figure 6.4 shows one case where *local exploration* algorithm is applied while the control u_3 that yields a counterclockwise rotation in place is executed. The robot is facing a narrow gate impossible to cross. There are gaps that do not disappear and their corresponding nodes must be properly labeled as primitive nodes at the end of *local exploration*. In Appendix A, a case in which *local exploration* algorithm is applied with the omnidirectional sensor positioned over rp is presented in detail.

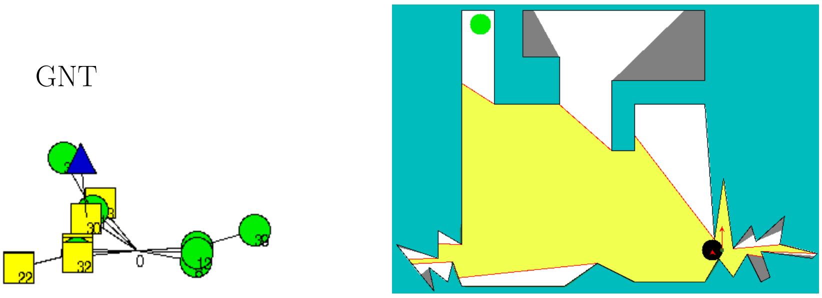


FIGURE 6.4: The robot has found a narrow gate, the control applied is u_3, while *local exploration* algorithm is executed. The GNT is shown and the nodes corresponding to the gaps within the unreachable region are not labeled as primitive nodes yet, because *local exploration* has not ended.

6.2 Environment 2

This example presents a more complex environment, where the landmark encoding process is presented. The GNT at the start of the exploration task is shown and also the GNT after the task has ended.

Figure 6.5 shows the beginning of the simulation in a more difficult environment, the meaning of the gray scale colors and the geometric shapes is the same used in Section 6.1. The starting GNT is also shown, there are four gaps detected, and none of them is primitive because neither of them comes from a gap appearance event.

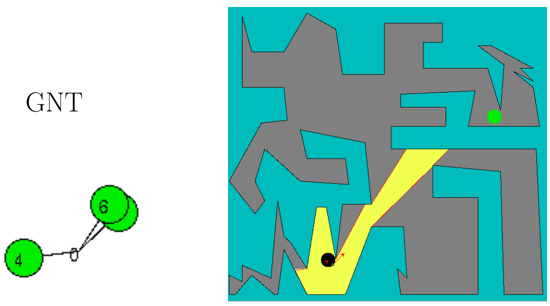


FIGURE 6.5: The robot and the environment at the beginning of the exploration task. The starting GNT is also shown.

Figure 6.6 shows the instant when Λ is (partially) visible for the first time. It is encoded in the GNT as a child of the node associated with the gap generated by the reflex vertex that is occluding Λ. Figure 6.7 illustrates an instant where Λ is fully visible. Therefore Λ is encoded as a root's child. The respective GNT with the new position of the landmark node is also shown.

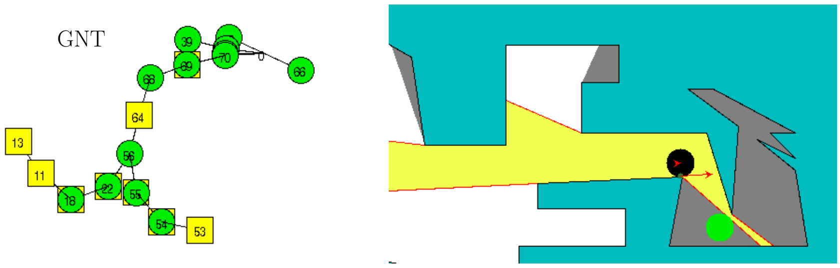


FIGURE 6.6: Λ is partially visible for the first time.

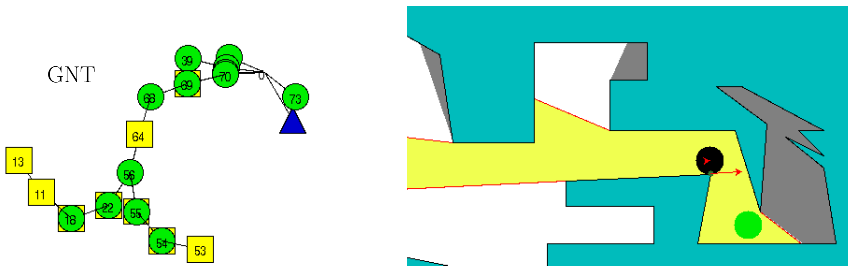


FIGURE 6.7: The whole area of Λ is visible, the corresponding GNT is shown with the landmark node connected to the root.

Figure 6.8 shows a new landmark occlusion. While the robot was executing control u_1, a gap partially occluded Λ after the whole landmark was visible. The landmark node becomes a child of the corresponding GNT node associated with the gap generated by the reflex vertex that is occluding Λ. The robot moves following ∂E using the defined Moore's machine structure and the whole landmark becomes visible again as shown in Figure 6.9. According to the landmark encoding process detailed in Section 3.4, the landmark node becomes a root's child.

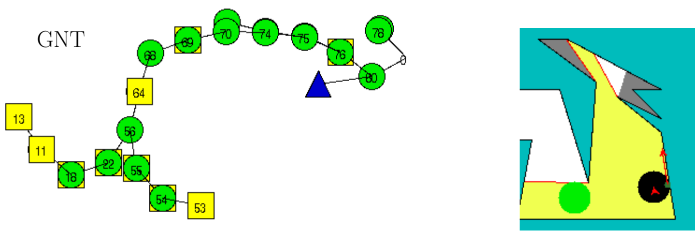


FIGURE 6.8: Λ becomes partially occluded and the landmark node is now a child of the corresponding GNT node.

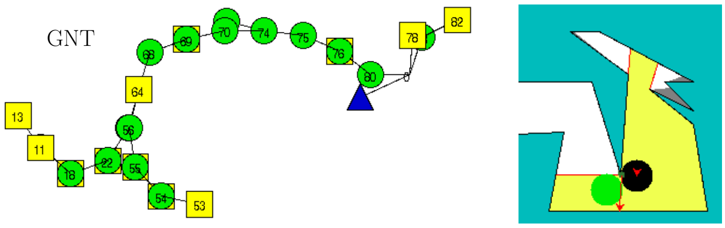


FIGURE 6.9: The whole landmark is visible again, and it becomes a root's child node as shown in the GNT.

Finally, Figure 6.10 shows the environment and the robot when the exploration task is over. The exploration task ends when the largest possible region of the environment has been discovered. The complete environment's representation is encoded in the GNT including the nodes corresponding to the unreachable gaps that did not disappeared, and the landmark node. The remaining gray spots correspond to those unreachable places which could not been discovered by the omnidirectional sensor regardless of the robot motion. *Local exploration* algorithm labeled those unreachable gaps as primitive nodes.

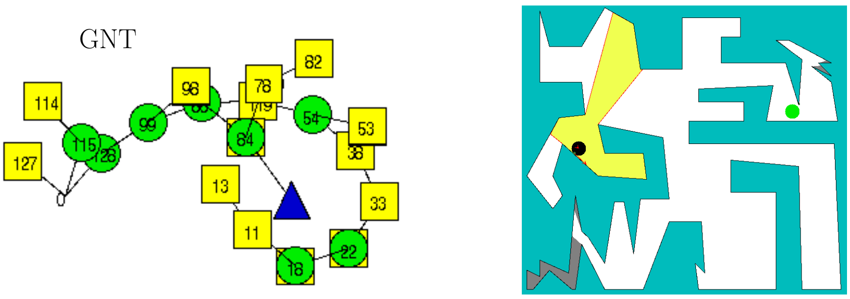


FIGURE 6.10: The exploration task ends and the largest possible region of the environment has been discovered.

6.3 Environment 3

This example presents an environment where the collision-free subset of the robot's configuration space $\mathcal{C}$ has two connected components.

Figure 6.11 shows the beginning of the simulation. The starting GNT is also shown, there are four gaps detected.

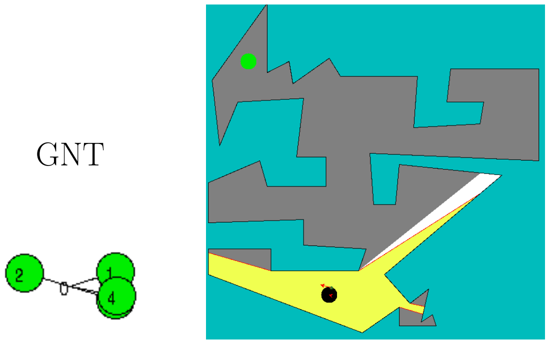


FIGURE 6.11: The robot and the environment at the beginning of the exploration task in an environment where $\mathcal{C}$ has two connected components. The starting GNT is also shown.

Figure 6.12 shows a case where *local exploration* algorithm is applied while the control u_3 that yields a counterclockwise rotation in place is executed after observation ye_4^R is detected. Λ lies inside the unreachable region which is a connected component of $\mathcal{C}$ different that the connected component where the robot lies. The landmark encoding process is the same and it is encoded as a child of the corresponding gap.

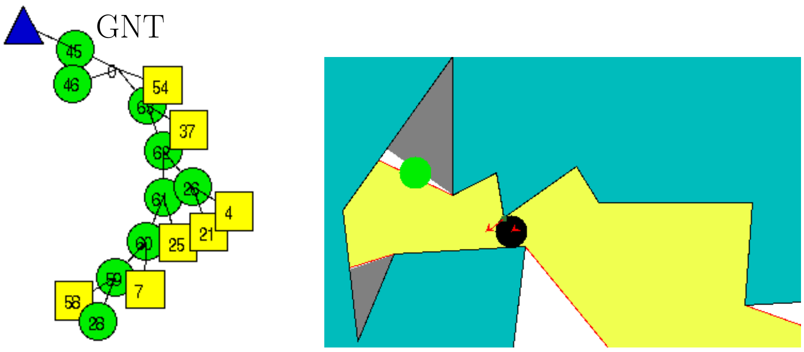


FIGURE 6.12: The robot faces the gate between two components of connected component of $\mathcal{C}$. *Local exploration* algorithm is applied while the control u_3 is executed. The GNT is also shown.

Figure 6.13 shows the robot and the GNT after *local exploration* labeled all the gaps within the unreachable region as primitive nodes.

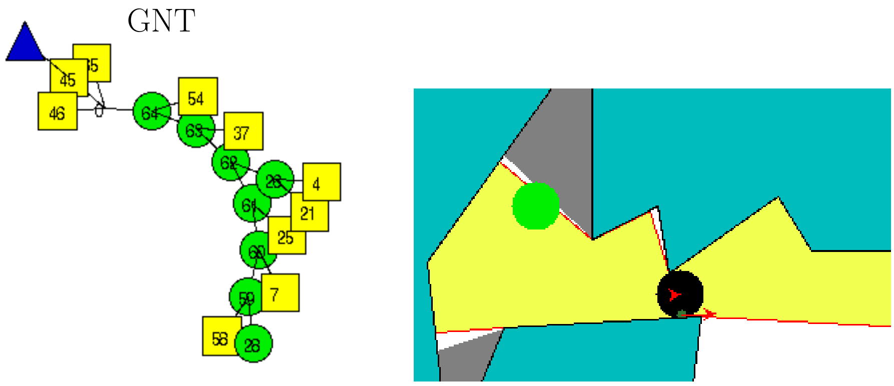


FIGURE 6.13: *Local exploration* ends and the gaps within the unreachable region (the other connected component of $\mathcal{C}$) are labeled as primitive nodes.

Finally, Figure 6.14 shows the environment, the robot and the complete GNT when the exploration task ends. The largest possible region of the environment has been discovered including the unreachable region that represents another connected component of $\mathcal{C}$. The landmark was encoded the same way that in the case of an environment where $\mathcal{C}$ is simply connected. The remaining gray spots correspond to those unreachable places which could not been discovered by the omnidirectional sensor from the connected component of $\mathcal{C}$ where the robot lies.

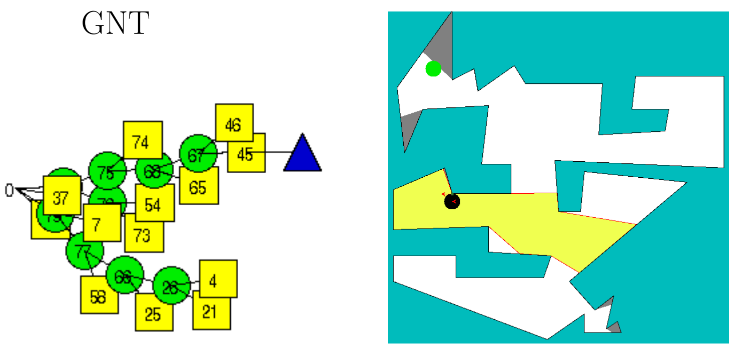


FIGURE 6.14: The exploration task ends and the largest possible region of the environment has been discovered.

Chapter 7

Conclusion

7.1 Conclusion

This thesis work addressed the problem of exploring an unknown environment, using a differential drive robot with the shape of a disc. The robot is equipped with simple sensors and it is unable to build precise geometric maps or localize itself in any Euclidean frame. The exploration problem addressed in this thesis is more challenging than the case of a point robot because visibility information does not provide collision free paths in the configuration space.

In this work, an exploration strategy is proposed. This exploration strategy is modeled as a Moore machine, and it guarantees exploring all the environment or the largest possible region of it while the GNT that represents the environment is built.

The robot is able to find a landmark or declare than an exploration strategy for this objective does not exist. A motion policy based on sensor feedback is also proposed. All the proposed algorithms have been implemented and simulation results were presented. We wrote a conference paper based on this work, and it is currently under review for 2014 IEEE International conference on Robotics and Automation (ICRA 2014).

7.2 Future work

The work presented in this thesis can be expanded, there are two important improvements that can be done in future research works.

The complete GNT resulting after the exploration task might be used for optimal navigation strategies (in terms of Euclidean distance) to reach the landmark Λ. Therefore, the main future work is to integrate this exploration strategy with the navigation strategy proposed by Rigoberto López [9]. That strategy has a similar robot's model, and uses the information provided by the GNT for optimal navigation for a differential drive robot with the shape of a disc.

This work can be also expanded by considering uncertainty in the motion model. Future work is to establish a probability measure of reaching a landmark even though the robot motion is not perfect.

Appendix A

Local Exploration Example

To clarify the *local exploration* algorithm, the method is illustrated with an example. Figure A.1 shows the time when the *local exploration* algorithm begins after observation ye_4^R is met. The gap sensor is placed over the rp point.

For illustration purposes the gaps are labeled using consecutive characters in alphabetic order. First, the initial linear list is created, *init-list*$= \{a, b, c, d, \uparrow, e, f, g, h, i, j, k\}$, where the characters represent the current gaps and $\uparrow$ represents the preferential direction rt.

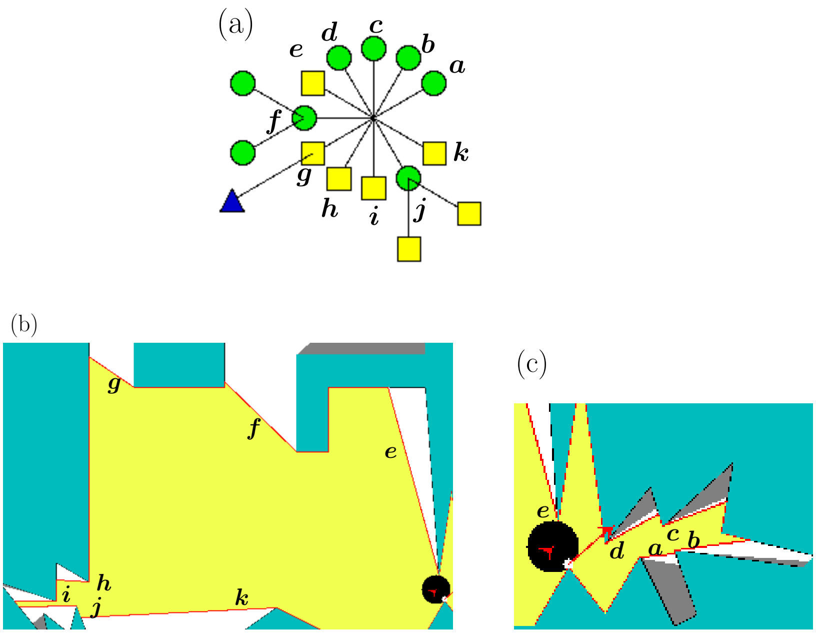


FIGURE A.1: (a) Current GNT at the beginning of *local exploration*, (b) Gaps out of the unreachable region, (c) Gaps within the unreachable region, those gaps must be labeled as primitive ones, after executing *local exploration* algorithm.

Figure A.2 shows the first critical event during the motion primitive execution, it is a merge between gaps c and d yielding the new gap l[1]. The linear list is then, *init-list*= $\{a, b, l, \uparrow, e, f, g, h, i, j, k\}$.

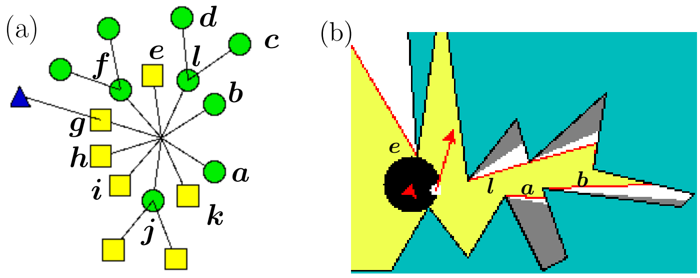


FIGURE A.2: (a) GNT with c and d as the children nodes of l due to the merge event, (b) The new gap l.

Figure A.3 shows the next critical event, a merge between gaps b and l yielding the new gap m, the current linear list is now *init-list*= $\{a, m, \uparrow, e, f, g, h, i, j, k\}$. Figure A.4 shows the next critical event, it is a merge between the gaps e and f generated by reflex vertices within the current non reachable region yields the new gap n. The linear list is now *init-list*= $\{a, m, \uparrow, n, g, h, i, j, k\}$.

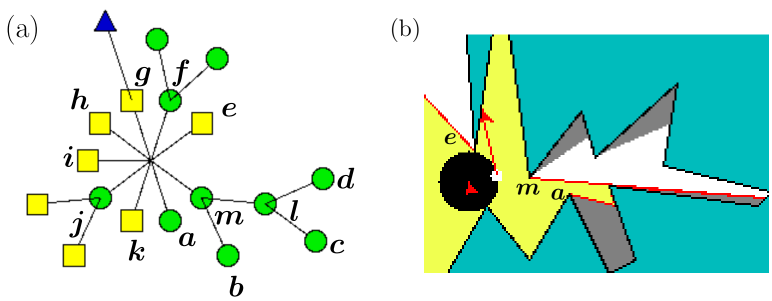


FIGURE A.3: (a) Current GNT with b and l as the children nodes of m due to the merge event, (b) The new gap m.

[1] According to the GNT evolution, nodes c and d become children nodes of the new node l which is connected to the root, all merge events have the same behavior.

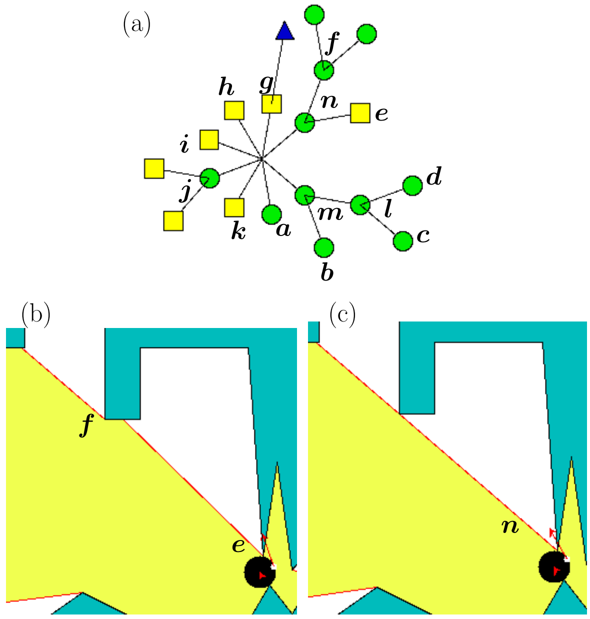


FIGURE A.4: (a) Current GNT with e and f as the children nodes of n due to the merge event, (b) Gaps e and f right before the merge event, (c) The new gap n.

Figure A.5 shows a merge event between gaps a and m yielding the new gap o, the current linear list is then *init-list*$= \{o, \uparrow, n, g, h, i, j, k\}$. Figure A.6 shows the next critical event, it is a split of gap n [2], the current linear list is *init-list*$= \{o, e, f, g, \uparrow, h, i, j, k\}$. It is important to notice that the linear order between the preferential direction and the gaps also changed.

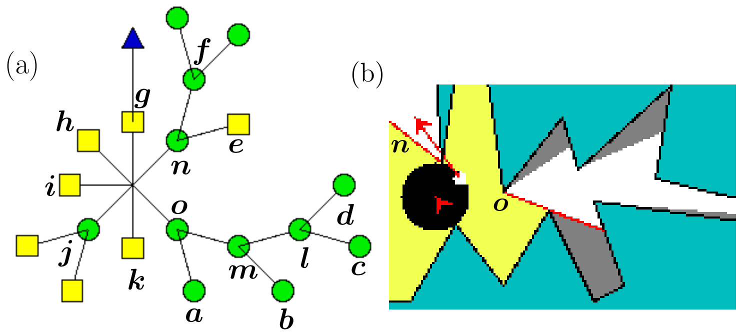


FIGURE A.5: (a) Current GNT with a and m as the children nodes of o due to the merge event, (b) The new gap o.

Figure A.7 shows when gap e disappears, the current linear list is *init-list*$= \{o, f, g, \uparrow, h, i, j, k\}$. Figure A.8 shows when a new gap p appears, *init-list* remains unchanged

[2] When a split event happens to a no leaf GNT node, the resulting gaps are its children nodes, therefore the gaps obtained are e and f.

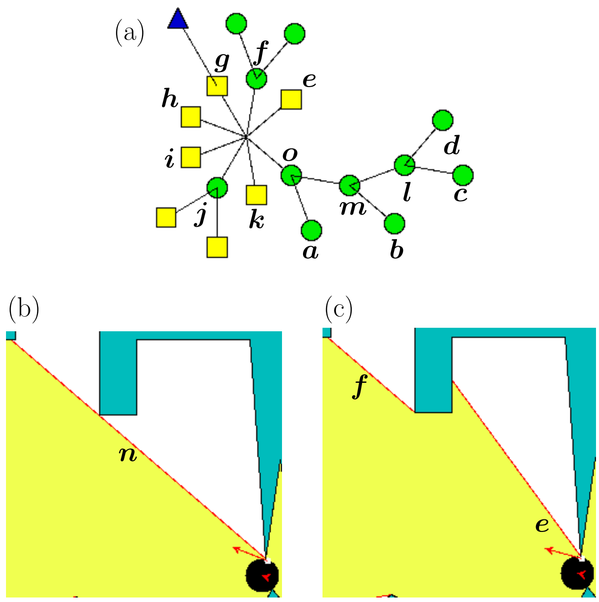


FIGURE A.6: (a) GNT with resulting gaps e and f after gap n splits (they were its children nodes), (b) The instant before n splits, (c) The resulting gaps e and f after the split event.

during gap appearance events (recall that appearances of gaps are not considered because they already have the primitive label, therefore they do not represent an issue).

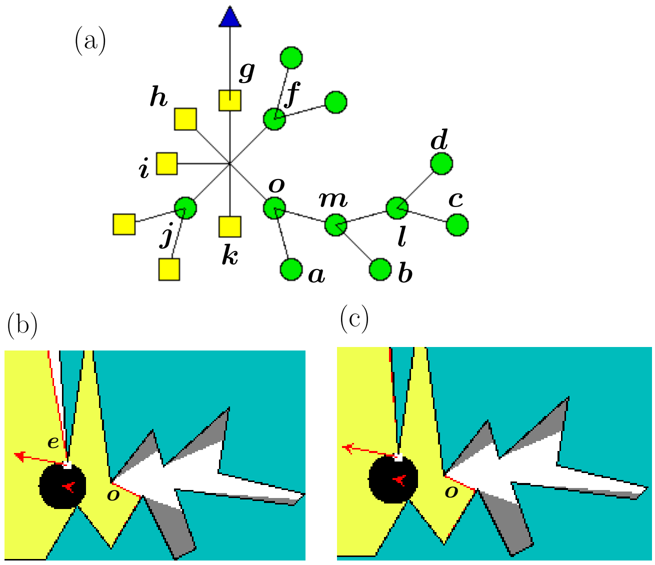


FIGURE A.7: (a) The GNT after gap e has disappeared, (b) The gaps within the unreachable region and gap e before disappearing, (c) The gaps after the disappearance event.

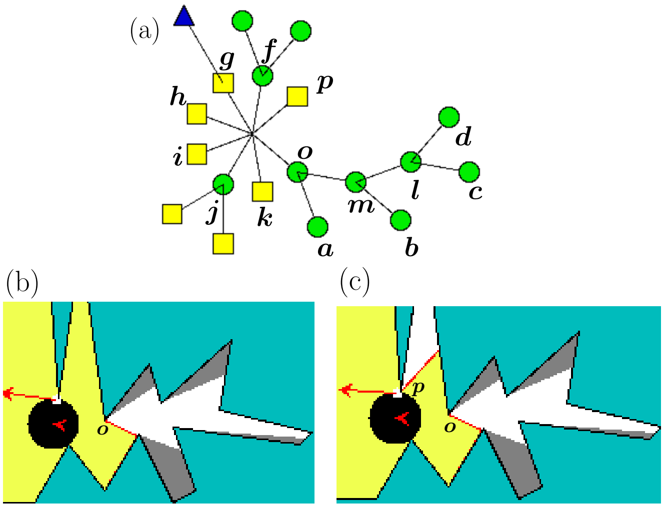


FIGURE A.8: (a) Current GNT with the new gap p, (b) The instant before the gap appear event, (c) The new gap p due to the gap appear event.

Finally, Figure A.9 shows the gaps when the algorithm 1 ends and the GNT leaves are correctly labeled, the final list is *end-list*$= \{f, g, \uparrow, h, i, j, k, o, p\}$. According to *local exploration* algorithm the auxiliar lists are: $G_1 = \{o, f, g\}$ and $G_2 = \{h, i, j, k, o, p\}$, the intersection list is then $G_\cap = \{o\}$, so gap o receives the *primitive* label and propagates it to its offspring (leaf nodes a,b,c and d).

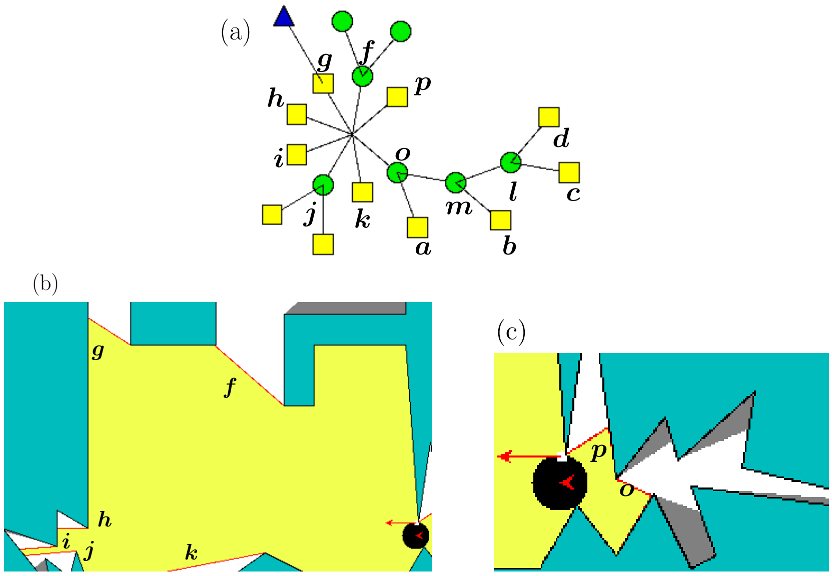


FIGURE A.9: (a) The GNT at the end of *local exploration* when the primitive label has been propagated to the leaf nodes representing unreachable gaps, (b) The gaps outside of the unreachable region, (c) The gaps within the unreachable region that have been correctly labeled as primitive ones.

Bibliography

[1] B. Tovar, R. Murrieta-Cid, and S. M. LaValle. Distance-optimal navigation in an unknown environment without sensing distances. *IEEE Transactions on Robotics*, 23(3):506–518, 2007.

[2] R. Murrieta-Cid, C. Parra, and M. Devy. Visual navigation in natural environments: From range and color data to a landmark-based model. *Journal on Autonomous Robots*, 13(2):143–168, September 2002.

[3] H. Bulata and M. Devy. Incremental construction of landmark-based and topological model of indoor environments by a mobile robot. In *Proc. Conf. IEEE Int Robotics and Automation*, volume 2, pages 1054–1060, 1996.

[4] R. Chatila and J. Laumond. Position referencing and consistent world modeling for mobile robots. In *Proc. IEEE Int. Conf. Robotics and Automation*, volume 2, pages 138–145, 1985.

[5] A. Elfes. Sonar-based real world mapping and navigation. *IEEE Transactions on Robotics and Automation*, 3(3):249–264, 1987.

[6] S. Thrun, D. Fox, and W. Burgard. Probabilistic mapping of an environment by a mobile robot. In *Proc. IEEE Int Robotics and Automation Conf*, volume 2, pages 1546–1551, 1998.

[7] M. Katsev, A. Yershova, B. Tovar, R. Ghrist, and S. M. LaValle. Mapping and pursuit-evasion strategies for a simple wall-following robot. *IEEE Transactions on Robotics*, 27(1):113–128, 2011.

[8] C.J. Taylor and D. Kriegman. Vision-based motion planning and exploration algorithms for mobile robots. *IEEE Transactions on Robotics and Automation*, 14(3): 417–426, 1998.

[9] R. Lopez-Padilla, R. Murrieta-Cid, and S.M. LaValle. Optimal gap navigation for a disc robot. In E. Frazzoli et al., editor, *Proc. of the Tenth Workshop on the Algorithmic Foundations of Robotics: Springer Tracts in Advanced Robotics*, pages 123–138. Springer-Verlag, Berlin, Heidelberg, 2013.

[10] S. Thrun. *A survey. In Exploring Artificial Intelligence in the New Millenium.* Morgan Kaufmann, 2002.

[11] F. Amigoni, S. Gasparini, and M. Gini. Building segment-based maps without pose information. *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 94(7):1340–1359, 2006.

[12] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics.* MIT Press, Cambridge, MA, 2005.

[13] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: Part I. *IEEE Robotics and Automation Magazine*, 13(2):99–110, 2006.

[14] B. Tovar, L. Munoz, R. Murrieta-Cid, M. Alencastre, R. Monroy, and S. Hutchinson. Planning exploration strategies for simultaneous localization and mapping. *Journal of Robotics and Autonomous Systems*, 54(4):314–331, April 2006.

[15] H. Choset and J. Burdick. Sensor based motion planning: The hierarchical generalized voronoi diagram. In J. Laumond and M. Overmars, editors, *Algorithms for Robotic Motion and Manipulation.* Springer-Verlag, 1997.

[16] B. Kuipers, R. Froom, W.-Y. Lee, and D. Pierce. The semantic hierarchy in robot learning. In *Robot Learning*, pages 141–170. Kluwer Academic Publishers, 1993.

[17] S. Teller. Automated urban model acquisition: Project rationale and status. In *DARPA Image Understanding Workshop*, pages 455–462, 1998.

[18] J. L. Crowley. World modeling and position estimation for a mobile robot using ultrasonic ranging. In *Proc. Conf. IEEE Int Robotics and Automation*, pages 674–680, 1989.

[19] J. Gonzalez, A. Ollero, and A. Reina. Map building for a mobile robot equipped with a 2d laser rangefinder. In *Proc. Conf. IEEE Int Robotics and Automation*, pages 1904–1909, 1994.

[20] N. Ayache and O. Faugeras. Building, registrating, and fusing noisy visual maps. *Int. J. Rob. Res.*, 7(6):45–65, 1988.

[21] S. Hutchinson. Exploiting visual constraints in robot motion planning. In *Proc. Conf. IEEE Int Robotics and Automation*, pages 1722–1727, 1991.

[22] C. Stachniss, O. M. Mozos, and W. Burgard. Speeding-up multi-robot exploration by considering semantic place information. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA 2006*, pages 1692–1697, 2006.

[23] H. Gonzalez Banos, E. Mao, J.-C. Latombe, T. Murali, and A. Efrat. Planning robot motion strategies for efficient model construction. In *Robotics ResearchThe 9th International Symposium*, pages 345–352, 1999.

[24] R. Sim and G. Dudek. Effective exploration strategies for the construction of visual maps. In *IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 69–76, 2003.

[25] R. Sim and G. Dudek. Online control policy optimization for minimizing map uncertainty during exploration. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1758–1763, 2004.

[26] J. Castellanos, J. Montiel, J. Neira, and J. Tard' s. A probabilistic framework for simultaneous localization and map building. *IEEE Transactions on Robotics and Automation*, 15(5):948–953, 1999.

[27] G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on Robotics and Automation*, 2001.

[28] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. Fastslam: A factored solution to the simultaneous localization and mapping problem. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, 2002.

[29] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In I. Cox and G. Wilfong, editors, *Autonomous Robot Vehicles*. Springer-Verlag, 1997.

[30] L. Murphy and P. Newman. Using incomplete online metric maps for topological exploration with the gap navigation tree. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA 2008*, pages 2792–2797, 2008.

[31] Y. Landa and R. Tsai. Visibility of point clouds and exploratory path planning in unknown environments. *Communications in Mathematical Sciences*, 6(4):881–913, 2008.

[32] F. Amigoni. Experimental evaluation of some exploration strategies for mobile robots. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA 2008*, pages 2818–2823, 2008.

[33] H. H. Gonzalez Banos and J.-C. Latombe. Navigation strategies for exploring indoor environments. *The International Journal of Robotics Research*, 21(10-11):829–848, 2002.

[34] J. O'Rourke. *Visibility.* CRC Press, Inc., 1997.

[35] T. Shermer. Recent results in art galleries. *Proceedings of the IEEE*, 80(9), 1992.

[36] A. A. Makarenko, S. B. Williams, F. Bourgault, and H. F. Durrant-Whyte. An experiment in integrated exploration. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems IROS*, pages 534–539, 2002.

[37] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proc. Symp. IEEE Int Computational Intelligence in Robotics and Automation CIRA97*, pages 146–151, 1997.

[38] J. C. Latombe. *Robot Motion Planning.* Kluwer Academic Publishers, Boston, 1991.

[39] F. Amigoni and V. Caglioti. An information-based exploration strategy for environment mapping with mobile robots. *Robotics and Autonomous Systems*, 58(5): 684–699, 2010.

[40] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. Younes. Coordination for multi-robot exploration and mapping. In *Proceedings of the AAAI National Conference on Artificial Intelligence.* AAAI, Austin, TX, 2000.

[41] B. Tovar, R. Murrieta-Cid, and C. Esteves. Robot motion planning for map building. In *Proc. IEEE/RSJ Int Intelligent Robots and Systems Conf*, volume 1, pages 673–680, 2002.

[42] B. Tovar, R. Murrieta-Cid, and C. Esteves. Robot motion planning for model building under perception constraints. In *International Symposium on Intelligent Robotic Systems*, pages 447–456, 2001.

[43] P. M. Newman, M. Bosse, and J. J. Leonard. Autonomous feature-based exploration. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA '03*, volume 1, pages 1234–1240, 2003.

[44] H. J. S. Feder, J. J. Leonard, and C. Smith. Adaptive mobile robot navigation and mapping. *The International Journal of Robotics Research*, 18(7):650–668, 1999.

[45] D. Calisi, A. Farinelli, L. Iocchi, and D. Nardi. Multi-objective exploration and search for autonomous rescue robots: Research articles. *J. Field Robot*, 24(8-9): 763–777, 2007.

[46] W. Burgard, M. Moors, C. Stachniss, and F. Schneider. Coordinated multi-robot exploration. In *IEEE Transactions on Robotics*, volume 21, pages 376–386, 2005.

[47] K. M. Wurm, C. Stachniss, and W. Burgard. Coordinated multi-robot exploration using a segmentation of the environment. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems IROS 2008*, pages 1160–1165, 2008.

[48] K. Singh and K. Fujimura. Map making by cooperating mobile robots. In *Proc. Conf.IEEE Int Robotics and Automation*, pages 254–259, 1993.

[49] W. Cohen, M. Moors, C. Stachniss, and F. Schneider. Adaptive mapping and navigation by teams of simple robots. In *Journal of Robotics and Autonomous Systems*, volume 18, pages 411–434, 1996.

[50] S. M. LaValle. Sensing and filtering: A fresh perspective based on preimages and information spaces. In *Foundations and Trends in Robotics Series.* Now Publishers, Delft, The Netherlands, 2012.