MULTIROBOT EXPLORATION WITH BUBBLE SPACE BASED TOPOLOGICAL MAPS

by

Bayram Cevdet Akdeniz

B.S., Electronics and Communication Engineering, Yıldız Technical University, 2010

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University

2013

ACKNOWLEDGEMENTS

It was a pleasure for me to work with all the wonderful people in ISL lab. First of all, I would like to thank Prof. Işıl Bozma for being a great advisor. Her ideas and tremendous support had a major influence on this thesis. She has also taught me how to write an academic paper and how to be a good academician which are as valuable as my thesis for me.

Accordingly, I want to thank all ISL lab members for their kind behaviors, friendships and academic supports. I have to thank individually to Hakan Karaoğuz and Ramazan Arıkan who help me so much when I use Robot Operating System (ROS).

As a research assistant in İstanbul Bilgi University, I should show gratitude to all professors in Electrical and Electronics department. They do not only support me with their experience but also show tolerance to work in Bogazici University ISL lab when necessary.

I should mention about all of my friends. I will not write their names individually in case I forget any of them. The presence of them in my life make easier to do many things. Finally, I want to thank my family who always motivated me in spite of being hundreds of kilometers away.

ABSTRACT

MULTIROBOT EXPLORATION WITH BUBBLE SPACE BASED TOPOLOGICAL MAPS

This thesis is concerned with autonomous exploration with single and multirobot systems. In particular, the robots are assumed to be endowed with three-dimensional laser sensors. The exploration strategies are based on bubble space representation that has been previously proposed to represent nodes in topological maps. First, the exploration of an environment by a single robot is considered. There are two aspects to this problem: terrain mapping and determining where to go. Terrain mapping aims to infer the environmental surface shape - as this certainly would affect the robot in determining where to go. For this, a novel approach based on bubble space representation is proposed and experimentally evaluated. For explorative navigation, the movement direction should be such that it should point the robot to unexplored territory while being accessible. A novel approach is proposed where the generation and recognition of nodes and their associated edges are achieved simultaneously with graph exploration in a topological map based on bubble space. The validity of these approaches are demonstrated by simulations and real-time experimental results.Next, the explorative navigation strategy is extended to multirobot exploration. In this case, the robots are assumed to be communicating with each other and determine their movement directions using the bubble surface information as well as their relative position information. Experimental results with real data show that the robots are able to explore unknown territories without much overlapping.

ÖZET

BALONCUK UZAYI TABANLI TOPOLOJİK HARİTALARDA ÇOKLU ROBOTLARLA ORTAM KEŞFETME

Bu çalışma, tek robotla ve çoklu robotlarla ortam keşfetme algoritmalarını ele almaktadır. Bu amaçla, topolojik haritalarda düğüm noktalarını temsil etmek için daha önceden bulunmuş baloncuk uzayı kullanılmıştır. İlk olarak tek robotla ortam keşfetme problemini gerçekleştirme amaçlanmıştır. Bu problem yer haritalama ve keşfetme stratejisi olmak üzere iki alt probleme ayrılmıştır. Yer haritalama seçilen ortamın önceden belirlenmiş özelliklerini çıkarma işlemidir. Yer haritalama ile ilgili olarak baloncuk uzayından ortamın yükselti haritasının çıkarılmasını sağlayan bir yöntem geliştirilmiştir. Bir ortamın robot tarafından başarılı ve verimli bir şekilde haritalandırılması için, robotun akılcı bir keşif stratejisi olmalıdır. Keşif stratejisi ile ilgili olarak baloncuk uzayı imgeye dönüştürülmüş ve imge işleme ile keşif için ilginç noktalar belirlenmiştir. Bu yaklaşımımızın geçerliliği simulasyonlarla ve gerçek ortamdaki deneylerle gösterilmiştir. Tek robotla ortam keşfetme algoritması çoklu robot sistemlerine uyarlanmıştır. Çalışmanın bu kısmında robotlar arasındaki haberleşmenin kusursuz sayıldığı ortamda, ortam keşfinin zamanın azaltılmasını sağlayacak yöntemler geliştirilmesi amaçlanmıştır. Bu amaçla, tek robotla ortam keşfetme algoritması çoklu robotlara uyarlanmış ve bu yöntemin verimliliği simulasyonlarla gösterilmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

b	A point in bubble space
c	Position of the robot
$d_p(A_t)$	Distance function of the pth robot for segmented part A_t
f_1	Pan angle
f_2	Tilt angle
f_2^*	The tilt angle that corresponds to the end of the flat surface
$\bar{f}_2$	The maximum limit of the tilt angle
g	Graph of the spatial map
h	Robot height
$r(t+1)$	Node assigner function at time t+1
t_0	Starting time of the exploration
t_f	Ending time of the exploration
u	Control input
x	Base of the bubble
$\mathcal{A}_t$	Segmented parts
$B_i'(x,t)$	Binarized bubble
$E_{\delta z}(x)$	Elevation error
$I(x,t)$	Bubble Descriptor
N_1	Cardinality of discrete pan set
N_2	Cardinality of discrete tilt set
N_t	Number of nodes in graph
α	Orientation of the robot
γ	Direction function
δz	Relative elevation
θ	Slope angle of the environment
$\kappa(b,x)$	Thresholding function

λ_1	The distance between the robot and ramped/inclined surface
μ_{ρ_l, f_2}	Mean of the fixed tilt values
$\mu(A_t)$	Average surface deformations
$\nu(A_t)$	Maximum extension of the surface deformations
$\rho(b, t)$	Response of the laser
σ_{ρ_l, f_2}	Standart deviation of the fixed tilt values
$\tau_1(b)$	Binary threshold
$\varphi(A_{t_i})$	Utility function for exploration

LIST OF ACRONYMS/ABBREVIATIONS

3D	Three Dimensional
CPET3DM	Canadian Planetary Emulation Terrain 3D Mapping
ICP	Iterative Closest Point
ISL	Intelligent Systems Laboratory
ROS	Robot Operating System
SLAM	Simultaneous localization and mapping

1. INTRODUCTION

Autonomous exploration is one of the most challenging problems in robotics. The primary goal is to determine the map of an unknown environment. The maps are represented primarily using two different approaches: metric and topological. While metric maps represent the geometric structure of the environment, typically they have too high computational demands for a direct application on a mobile robot [2]. Alternatively, topological maps represent environments by graph-like structures where nodes correspond to places and edges to paths between them [3]. Although topological maps scale better to large environments, they are thought to lack the ability to represent the geometric structure of the environment [4].

In this thesis, exploration with topological maps is considered. In particular, we use bubble space as the nodes of the topological map [5]. In bubble space representation, bubble surfaces encode different sensory features in a manner that is implicitly dependent on robot pose. Each bubble surface is a deformable surface that simultaneously encodes one type of sensory feature values and their local S2-metric relations from the robot's viewpoint. For completeness, the mathematical formulation of bubble space is presented in Appendix A. In this thesis, the robots are assumed to be endowed with three-dimensional (3D) laser sensors and bubble surfaces are constructed using 3D laser data.

The exploration of an environment can be accomplished either via a single robot or multiple robots. Initially, we have focused on single robot exploration. There are two aspects to this problem: terrain mapping and determining where to go. Terrain mapping aims to infer the environmental surface shape - as this certainly would affect the robot in determining where to go. For this, a novel approach based on bubble space representation is proposed. In this approach, bubble surfaces constructed using 3D laser data are used to define local slope maps - which in turn are used to generate the local terrain map on-demand for its localization and motion planning.

For explorative navigation, the movement direction should be such that it should point the robot to unexplored territory while being accessible. As the nodes (bubble surfaces) of the topological map are actually used to determine the unexplored directions, the robot is able to incrementally extend its map via the nodes generated at new places that are reached via moving in these directions.

For correct mapping, the robots should consider pose correction. Pose correction is necessary due to odometric errors as well as environmental conditions. In our approach, pose correction is done in bubble space via comparing expected bubble surfaces with those that are actually generated.

This approach is then extended to multirobot setting. In this case, the robots are assumed to be communicating with each other and determine their movement directions using the bubble surface information as well as their relative position information with the overall strategy that they should be maximally scattered over the terrain with minimal overlapping of places explored by different robots.

1.1. Contributions

The contributions of this thesis can be summarized as follows:

Terrain mapping: A novel approach to local terrain mapping with topological maps is proposed. In this approach, bubble surfaces constructed using three-dimensional laser data are used to derive local terrain maps that encode local terrain slope and its proximity for each pan direction.

Single robot explorative navigation: A bubble space based approach to determining the movement directions is proposed. The novelty of this approach is that the generation and recognition of nodes and their associated edges are achieved simultaneously with graph exploration. As the nodes (bubble surfaces) of the topological map are actually used to determine the unexplored directions, the robot is able to incrementally extend its map via the nodes generated at new places that are reached via moving in these

directions.

Multi-robot explorative navigation: The single robot exploration strategy is extended to multirobot settings. The contribution of this approach is that the robots determine movement directions based on 3D laser sensory feedback from their current environments as well relative pose information - assuming that the robots are able to communicate with each other and exchange relative position information.

1.2. Outline

This thesis is organized as follows: In Chapter 2, the local terrain mapping approach is presented along with experimental evaluation results. Single robot exploration is explained in Chapter 3 and tested using both simulated and experimental data as well as with a real-time tracked robot. In Chapter 4, the single robot explorative navigation strategy is extended to the multirobot settings along with experimental results. The thesis concludes with a brief summary. For completeness, the mathematical formulation of bubble space representation is presented in Appendix A. The description of Jaguar robot used in single robot exploration is given in Appendix B.

2. LOCAL TERRAIN MAPPING VIA 3D LASER BASED BUBBLE SURFACES

Terrain mapping is the process by which surface shape obtained from different vantage points are accumulated into a consistent environmental model [6]. As such, it requires sensing the inclined or ramped parts of the ground surface and building a terrain representation accordingly [7]. It is a critical component of mobile robot navigation in unknown outdoor environments - as the local topography directly affects the robot's pose and imposes constraints on its motion [8–12]. In this chapter, we focus on local terrain mapping with topological maps.

2.1. Related Literature

There are two related aspects: sensing and representation. The surrounding terrain can be sensed by a variety of sensors [12]. However, due to availability and reliability, stereo vision and laser rangefinders are most commonly used. Most work on terrain mapping are based on analyzing the images captured from the robots' camera [7, 13–15]. However, these approaches are not robust against changes in illumination conditions that may dramatically alter the appearance of the terrain in natural settings [16]. While this problem may be alleviated via augmenting the intensity values with additional information [16], the strong dependency on intensity values remains nevertheless. Alternatively, distance sensors are used as these sensors are relatively robust in case of varying light and temperature [17, 18]. For example, time-of-flight measurements from two vertically placed sonar sensors having the same viewing direction are used to estimate the terrain slope [19]. Similarly, laser range finders scanning the environment vertically and horizontally are used for terrain modelling [4, 18, 20]. With the recent developments in three-dimensional (3D) lasers, it has become possible to obtain high resolution digital terrain data [1]. Although visually appealing, efficient terrain modelling structures have become more imperative as the enormous number of points make data management and reasoning both complex and time consuming [21, 22].

Hence, the second aspect pertains to representation. In most works, this is based in metric maps using a variety of tile representations such as grid-based approximations or geometric primitives [21]. For example, the map is obtained via fitting of planes to small areas in the sensed data using a variety of different methods [7,14,15,23,24]. The complexity of full three-dimensional maps is reduced by using Cartesian elevation maps which encode $2\frac{1}{2}$-dimensional height information of the terrain in a two-dimensional grid [4,6,20,25,26]. While metric maps represent the geometric structure of the environment, still, they typically have too high computational demands for a direct application on a mobile robot [2]. Alternatively, topological maps represent environments by graph-like structures where nodes correspond to places and edges to paths between them [3]. While topological maps scale better to large environments, they are thought to lack the ability to represent the geometric structure of the environment [4]. Perhaps partially due to this, to the best of authors' knowledge, there is no reported work on deriving terrain maps from the topological representation of nodes only. This is fundamentally different from adding terrain information – which can easily be done as topological representations are agnostic with respect to the type of metrical information.

2.2. General Approach

This chapter presents a novel approach to generating local terrain maps. As most work, we assume that the terrain is lightly cluttered so that there are no overwhelming obstacles in any direction that block the robot's sight of view [8]. Unlike most work that use metric maps, in our approach, local terrain maps are derived based on bubble surfaces - which have been shown to correspond to nodes in the topological maps via representing all features in a manner that is implicitly dependent on robot pose while preserving their local S^2-geometry and [5]. The contribution of this chapter is to derive local terrain maps from bubble surface representation based on 3D laser sending. We show that each bubble surface along with the robot's geometry defines a local slope map of the terrain gradient and its proximity for each pan direction - depending on the robot's base. Hence, instead of generating a global terrain map, the robot generates a local terrain map on-demand depending on its pose which can then be used for its

localization and motion planning.

2.3. Local Slope Map

Suppose the robot is at base x with a 3D laser sensor mounted at height h as shown in Figure 2.1 and constructs a laser bubble surface $B(x,t)$. The robot's viewing direction with respect to its horizon depends on the tilt viewing direction f_2. If the tilt angle $f_2 = 0$, then the robot's viewing direction is parallel to the ground. If $f_2 < 0$, then f is associated with a viewing direction that is looking downwards from its horizon by an amount that is given by the magnitude of f_2. On the other hand, if $f_2 > 0$, then the robot is looking upwards from its horizon. We consider the part of bubble surface when the robot is looking downwards with $f_2 < f_H$.

The local terrain model is based on approximating the terrain in each (pan) viewing direction f_1 by two piecewise planar surfaces as shown in Figure 2.1. Of course, this will lead to a coarse modelling of the terrain in realistic settings [21]. Nevertheless, such terrain maps will still be useful – particularly in cases when steep ramps and inclines are detected which may need to be avoided by the autonomous robots. Furthermore, as the robot computes local variations in the terrain slope as formulated in the sequel, this information can be used if crucial to the task at hand. The first plane corresponds to the terrain on which the robot is standing while the second plane corresponds to a sloped terrain. The sloped terrain may be either a ramp (slope upwards) or incline (slope downwards) as shown in Figure 2.1. The relative geometry of the two planes can be described by two parameters – relative slope θ of the second plane with respect to the first plane and its starting distance λ_1. As such an approximation is made for each pan viewing direction f_1, each bubble surface is associated with a map $s_x : \mathcal{F}_1 \rightarrow S^1 \times R$ - which we will refer to as the local slope map. This map is an egocentric representation of the gradients of the surrounding terrain and their proximity as seen from the current robot pose x as:

$$s_x(f_1) = \begin{bmatrix} \theta(f_1) & \lambda_1(f_1) \end{bmatrix}^T$$

The first component $\theta(f_1)$ denotes the gradient of the terrain in pan direction f_1 and $\lambda(f_1)$ denotes its corresponding proximity. In the sequel, we will omit the f_1 argument in order to simplify the notation.

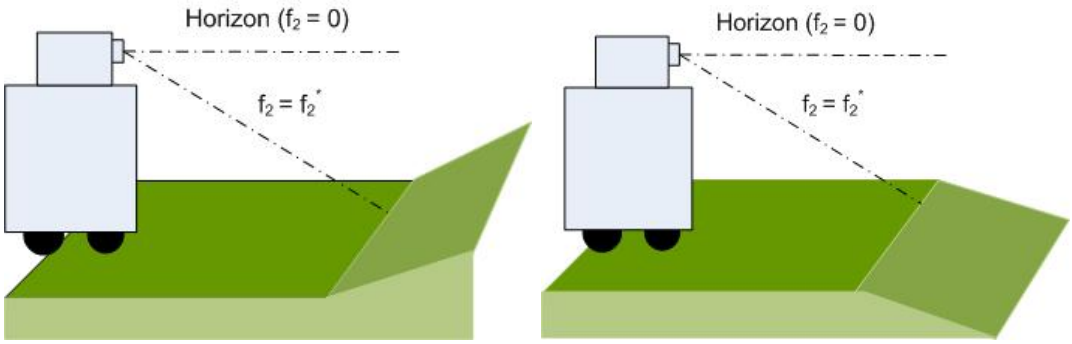

Figure 2.1. Left: Ramped slope; Right: Inclined slope.

2.3.1. Slope Type

Suppose that the robot is looking with pan viewing direction f_1. In case of flat terrain, the geometry dictates that the range reading $\rho(b,t)$ to be:

$$\rho(b',t) = \frac{h}{\sin(f_2)} \qquad f_2 \leq 0 \tag{2.1}$$

In case of sloped terrain, Equation 2.1 will not hold for $f_2^* < f_2 \leq 0$ where $f_2^* \leq 0$ is the critical tilt value associated with the start of the sloped terrain. The f_2^* value can be determined by starting from the smallest possible f_2 and keep looking upwards until the equality is satisfied or maximum upward direction is reached as explained in detail later in the sequel. Hence, the type of slope (flat, ramp or incline) can be determined as:

$$\begin{cases} \text{Ramp} & \text{if } \rho(b,t) - \frac{h}{\sin(f_2)} < 0, f_2^* < f_2 < 0, \\ \text{Incline} & \text{if } \rho(b,t) - \frac{h}{\sin(f_2)} > 0, f_2^* < f_2 < 0 \\ \text{Flat} & \text{otherwise} \end{cases} \tag{2.2}$$

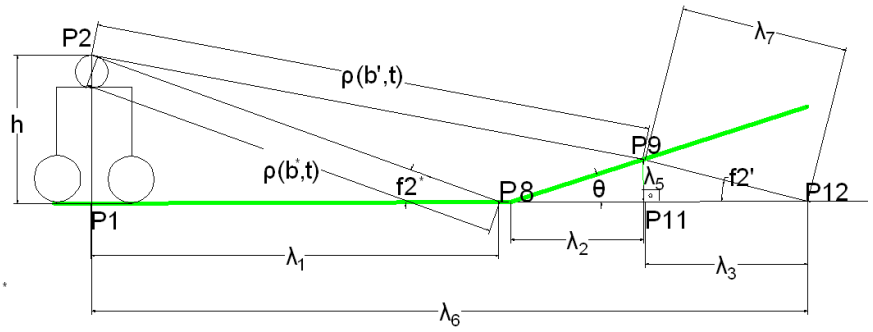

Figure 2.2. Geometry of a ramped terrain.

2.3.2. Ramp Estimation

Consider a ramped slope as shown in Figure 2.2. The critical tilt direction f_2^* is defined by:

$$\rho(b',t) - \frac{h}{\sin(f_2')} = \begin{cases} < 0 & f_2^* < f_2' < 0 \\ 0 & f_2'\prime \leq f_2^* \end{cases} \tag{2.3}$$

The value of $\rho(b',t)$ will decrease depending on the amount of ramp:

$$\rho(b',t) = \frac{h}{\sin(f_2')} - \lambda_7 \tag{2.4}$$

with $\lambda_7 \neq 0$. The starting distance λ_1 of the ramp is equal to:

$$\lambda_1 = \frac{h}{\tan(f_2^*)}$$

Letting λ_5 denote the relative elevation, the ramp gradient at (f_1, f_2') is equal to:

$$\theta(f_1, f_2') = atan2(\lambda_5, \lambda_2) \tag{2.5}$$

Using the similarity of triangles $P_1P_2P_{12}$ and $P_{11}P_9P_{12}$ as:

$$\frac{\lambda_5}{h} = \frac{\lambda_3}{\lambda_6} = \frac{\lambda_7}{\rho(b',t) + \lambda_7} \tag{2.6}$$

where

$$\lambda_6 = \lambda_1 + \lambda_2 + \lambda_3 = \frac{h}{\tan(f_2')} \tag{2.7}$$

$$\lambda_7 = \frac{h}{\sin(f_2')} - \rho(b', t) \tag{2.8}$$

Again based on geometry:

$$\lambda_5 = \lambda_7 \sin(f_2') \tag{2.9}$$

With some manipulation after combining Equation 2.6 and Equation 2.7,

$$\lambda_2 = \frac{h}{\tan(f_2')} - \lambda_1 - \lambda_7 \cos(f_2') \tag{2.10}$$

The overall ramp gradient is then defined as the average ramp gradient as:

$$\theta(f_1) = \frac{1}{\bar{f}_2 - f_2^*} \int_{f_2 = f_2^*}^{\bar{f}_2} \theta(f_1, f_2) df_2 \tag{2.11}$$

Of course, if the local variations in the slope are required for the task at hand, the robot will keep the individual slope values without averaging.

2.3.3. Incline Estimation

Next, we consider the case when at he robot faces an inclined terrain as seen in Figure 2.3(Top). Due to the change in geometry, the resulting formulation changes slightly. Again, for each pan viewing direction f_1', the incline associated with a tilt direction f_2^*:

$$\rho(b', t) - \frac{h}{\sin(f_2')} = \begin{cases} > 0 & f_2^* < f_2' < 0 \\ 0 & f_2' \leq f_2^* \end{cases} \tag{2.12}$$

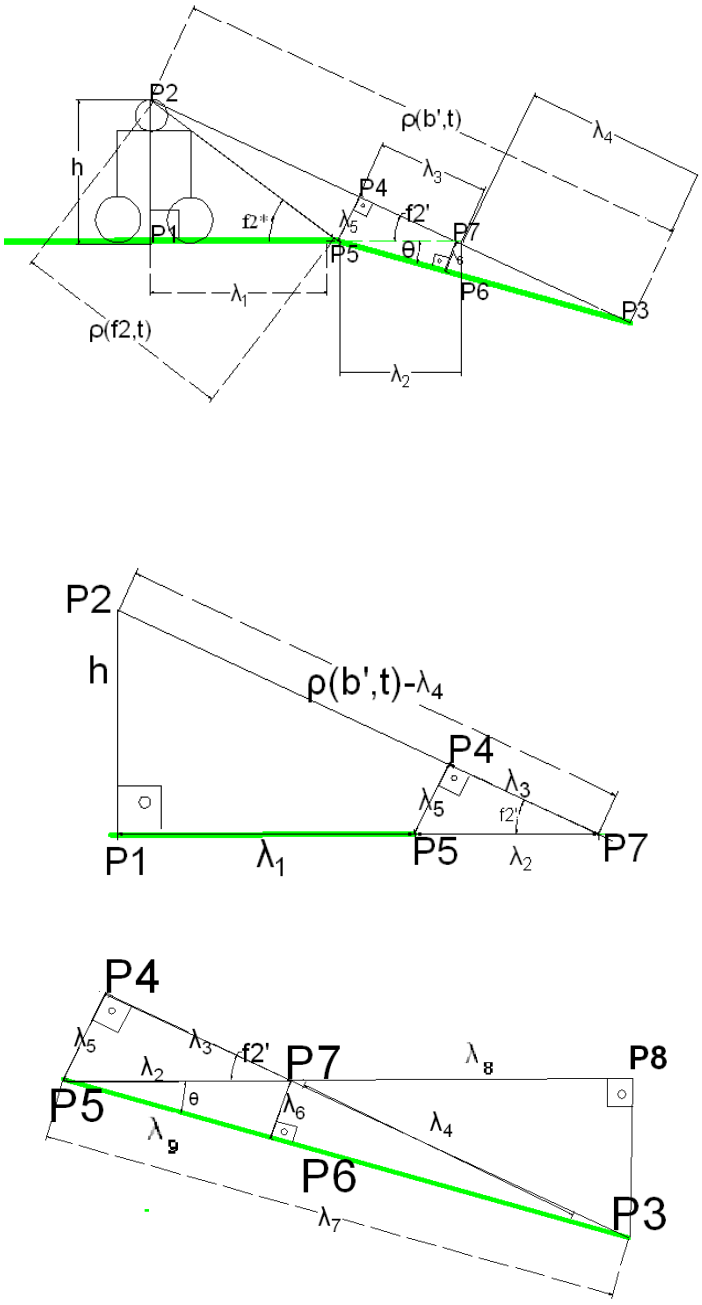

Figure 2.3. Top: General geometry. Center: Detailed geometry of incline start. Bottom: Detailed geometry of incline.

The gradient of the incline can be easily derived as seen in Figure 2.3(bottom):

$$\theta = \arcsin(\lambda_6, \lambda_2) \tag{2.13}$$

Now we need to find the values λ_6 and λ_2 respectively. From Figure 2.3

$$\lambda_2 = \frac{h}{\tan(f_2')} - \frac{h}{\tan(f_2^*)} \tag{2.14}$$

The distance to the incline λ_1 is equal to:

$$\lambda_1 = \frac{h}{tan(f_2^*)} \tag{2.15}$$

Computing

$$\lambda_5 = \lambda_2 \sin(f_2') \tag{2.16}$$

By similarity of triangles $P1P2P7$ and $P4P5P7$,

$$\frac{\lambda_5}{h} = \frac{\lambda_2}{\rho(b', t) - \lambda_4} = \frac{\lambda_3}{\lambda_1 + \lambda_2} \tag{2.17}$$

Thus,

$$\lambda_3 = \frac{\lambda_5(\lambda_1 + \lambda_2)}{h} = \lambda_2 \cos(f2') \tag{2.18}$$

Now consider the triangles $P_3P_4P_5$ and $P_3P_6P_7$ as shown in Figure 2.3(Top). By their similarity,

$$\lambda_6 = \lambda_5 \frac{\lambda_4}{\lambda_7} \tag{2.19}$$

where

$$\begin{aligned} \lambda_4 &= \rho(b', t) - \frac{h}{\sin(f_2')} \\ \lambda_7 &= \sqrt[2]{\lambda_5^2 + (\lambda_3 + \lambda_4)^2} \end{aligned}$$

Substituting, λ_6 is equal to:

$$\lambda_6 = \frac{\lambda_2 \sin(f_2')\lambda_4}{\sqrt[2]{\lambda_5^2 + (\lambda_3 + \lambda_4)^2}} \tag{2.20}$$

The gradient of the incline is equal to:

$$\theta(f_1, f_2') = \arcsin(\frac{\lambda_2 \sin(f_2')\lambda_4}{\sqrt[2]{\lambda_5^2 + (\lambda_3 + \lambda_4)^2}}, \lambda_2) \tag{2.21}$$

The overall ramp gradient is computed again using Equation 2.11.

2.4. Local Terrain Map

The robot can estimate relative elevation in its neighborhood using the bubble surface $B(x,t)$ and the local slope map s_x. Since the estimation is valid only locally, we refer to it as the local terrain map. Note that unlike metric maps, local terrain map is generated only for a neighborhood of each base point x. In bubble space, consider $\mathcal{B}' \subset \mathcal{B}$ where $\mathcal{B}' = \{b \in \mathcal{B} \mid \pi(b) = x\}$. The local terrain map is constructed via defining the map $\delta X : \mathcal{B}' \to R^3$ where

$$\delta X(b) = \begin{bmatrix} \delta c(b) \\ \delta z(b) \end{bmatrix}$$

where $\delta c(b) \in R^2$ and $\delta z(b) \in R$ are relative base displacement and relative elevation parameters. Note that

$$\delta c(b) = \|\delta c(b)\| \begin{bmatrix} sin(f_1) \\ cos(f_1) \end{bmatrix} \tag{2.22}$$

2.4.1. Ramped Slope

For the ramped slope, $\|\delta c\|$ is equal to:

$$\|\delta c(b)\| = \begin{cases} \rho(b,t)\cos(f_2) & \text{if } f_2 < f_2^* \\ (\lambda_1 + \lambda_2) & \text{otherwise} \end{cases} \tag{2.23}$$

On the other hand, the associated relative elevation $\delta z(b)$ is:

$$\delta z(b) = \begin{cases} 0 & \text{if } \|\delta c\| < \lambda_1 \\ \lambda_2 \tan\theta & \text{otherwise} \end{cases}$$

where the Equation 2.8 and Equation 2.10,

$$\lambda_2 = \frac{h}{\tan(f_2)} - \lambda_1 - \left(\frac{h}{\sin(f_2)} - \rho(b,t)\right)\cos(f_2) \tag{2.24}$$

2.4.2. Inclined Slope

In case of inclined slope, $\|\delta c\|$ corresponds to the line that passing from the points P_1, P_5, P_7 and P_8 as shown in Figure 2.3:

$$\|\delta c\| = \begin{cases} \rho(b,t)\cos(f_2) & \text{if } f_2 < f_2^* \\ (\lambda_1 + \lambda_2 + \lambda_8) & \text{otherwise} \end{cases} \tag{2.25}$$

The elevation map is equal to:

$$\delta z(b) = \begin{cases} 0 & \text{if } \|\delta c\| < \lambda_1 \\ (\lambda_2 + \lambda_8)\tan\theta & \text{otherwise} \end{cases}$$

where

$$\lambda_8 = \lambda_4 \cos(f_2) \tag{2.26}$$

2.5. Experimental Results

The proposed approach has been tested with a variety of different settings ranging from simulated terrains in Webots to Canadian Planetary Emulation Terrain 3D Mapping Dataset [1].

2.5.1. Simulated Outdoors Terrain

The first set of experiments are conducted with simulated outdoors environment with varying terrain ramp using Webots [27]. The robot is made to stand in front of a terrain with a given gradient θ^* that starts in λ_1^* meters as shown in Figure 2.4. The gradient θ^* is varied between 2.86°-11.44° while its proximity λ_1^* is varied between 0.5-1 meters. It uses the proposed approach to generate estimates θ and λ_1. The results are as shown in Table 2.1. The gradient estimates θ have an average error $E_\theta = 15\%$ while the distance estimates λ_1 have an average error of $E_{\lambda_1} = 5$ %. As expected, as both the gradient magnitude and the distance to the slope increase, estimation errors increase. Interestingly, for a fixed gradient magnitude, as robot's distance increases by 100%, the corresponding error increases by around 5%. Table 2.2 involves comparision between our method and a recent work based on slope prediction by using laser range finder [18].

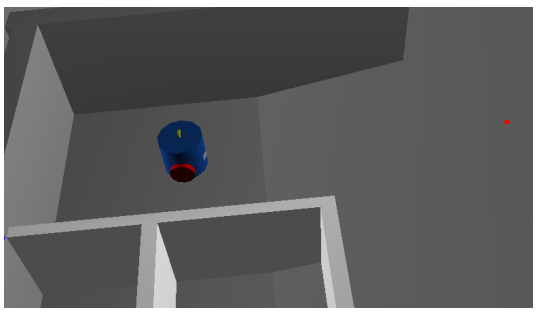

Figure 2.4. A sample terrain simulated using Webots.

2.5.2. Canadian Planetary Emulation Terrain 3D Mapping Dataset

Next, we apply our approach with Canadian Planetary Emulation Terrain 3D Mapping (CPET3DM) dataset which is a collection of 3D laser scans [1] . Note that,

Table 2.1. Webots Results.

Actual		Estimated		$E_\theta(\%)$	$E_{\lambda_1}(\%)$
θ^* (°)	λ_1^*(m)	θ (°)	λ_1(m)		
2.86	0.5	2.63	0.48	8	4
	1	2.52	0.91	12	9
5.72	0.5	4.98	0.49	13	2
	1	4.87	0.93	15	7
8.58	0.5	7.44	0.49	14	2
	1	7.16	0.92	17	8
11.44	0.5	9.74	0.495	15	1
	1	9.16	0.94	20	6

Table 2.2. Comparision of two slope prediction methods.

Slope prediction method	average prediction error (%)
Slope Prediction with Bubble Space	15
Slope Detection based on Orthogonal Assumption	4

this dataset is compatible with our assumptions and the environment that data collected can be considered as a realistic outdoor scenario which can be seen in Figure 2.6. In particular, we use `a100_dome_vo` dataset that is generated in the UTIAS indoor rover test facility. This dataset consists of 3D laser scans that are obtained from 50 different locations with a large inter-scan spacing in a gravel-filled circular workspace area 40 m in diameter. The real view of the terrain is shown in Figure 2.6. At each location c, the robot has 3D laser data with $f_1 \in [0, 360]^\circ$ having increments $\delta f_1 = 0.36^\circ$ and $f_2 \in [-30, -3]^\circ$ having increments $\delta f_2 = 0.5^\circ$ respectively. Hence, the number of scan points is $N_1 = 1000$ and $N_2 = 56$ in each direction. It generates the local terrain map s_x associated with the current base point x. The bubble surfaces, actual and estimated local terrain maps for three different base points are as shown in Figure 2.5. The black crosses in Figure 2.5 indicate the position of the robot. As expected, it is observed that the robot is able to estimate closer terrain quite accurately. However, estimation error increases as distance increases. This is because - from the perspective of the robot, the two plane approximation may not be valid. For example, if a ramped terrain is

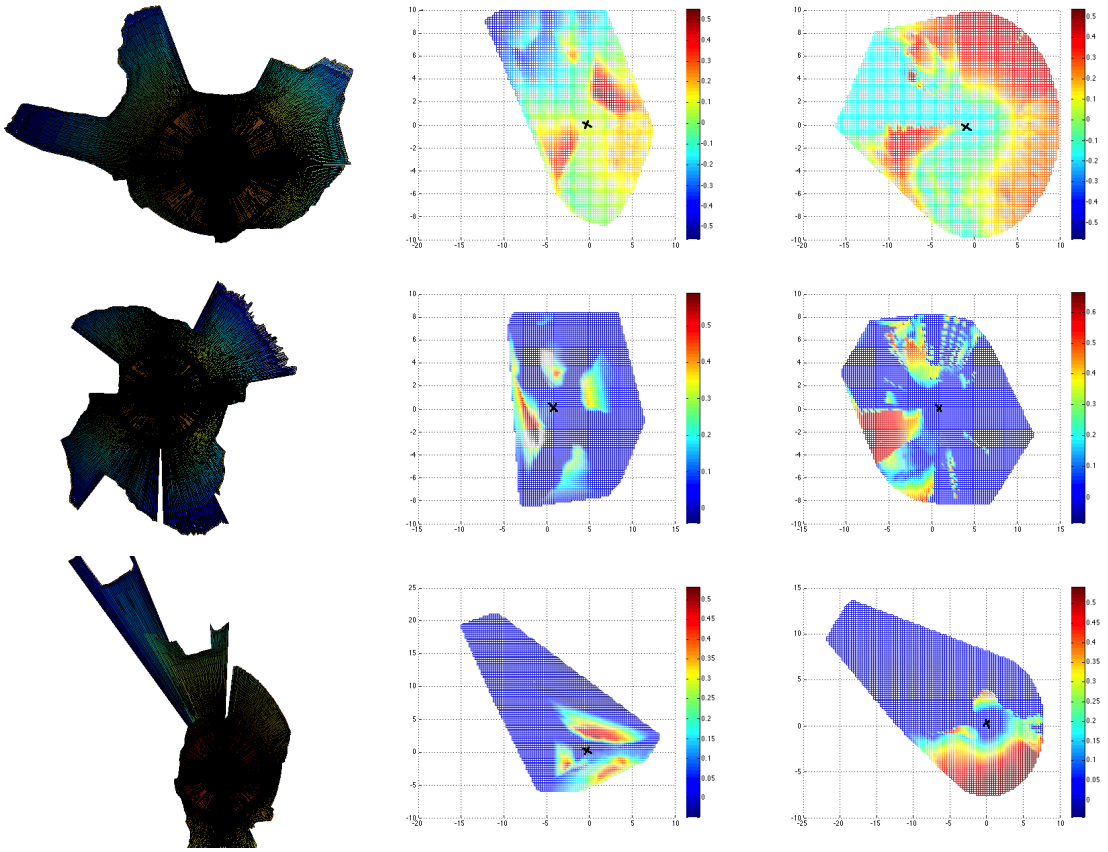

Figure 2.5. Top: Location $c = [4.9\,2.2]^T$. Left: Bubble surface; Center: Actual elevation map ; Right: Estimated terrain map. Middle: Location $c = [0\,0]^T$. Left: Bubble surface; Center: Actual elevation map; Right: Estimated terrain map. Bottom: $c = [-6.7\ -9.8]^T$: Left: Bubble surface; Center: Actual elevation map; Right: Estimated terrain map.

followed by an inclined terrain, the robot may be blocked from viewing the inclined terrain unless it moves to the end of ramped terrain. In local terrain map, this terrain will be wrongly estimated as being ramped slope. The results are compared with the actual elevation values δz^* as shown in Figure 2.7 with error defined as:

$$E_{\delta z}(x) = \int_0^{2\pi} \int_{-0.52}^{-0.052} \left| \frac{\delta z^*(b) - \delta z(b)}{\delta z^*(b)} \right| df_1 df_2$$

where $30^\circ \approx 0.52$ rad and $3^\circ \approx 0.052$ rad. It is observed that the average error is 0.033 m (around 3%) with variance is 0.0026 m. Furthermore, the data size is reduced from $3N_1N_2$ down to $2N_1$.

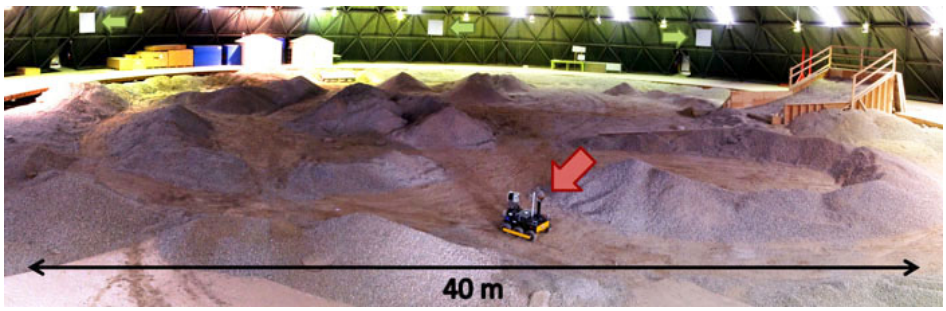

Figure 2.6. Real view of the terrain where the dataset collected [1].

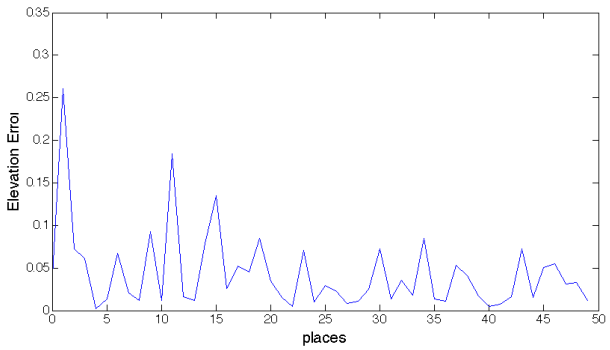

Figure 2.7. CPET3DM dataset- $E_{\delta z}$ for each robot base point.

As expected, as the resolution of the tilt angle δf_2 increases, the estimation errors will decrease accordingly. The results are as shown in Table 2.3. When the resolution goes from down $\delta f_2 = 0.5°$ to $\delta f_2 = 4°$, average $E_{\delta z}$ increases by 10%.

Table 2.3. Average $E_{\delta z}$ wrt tilt resolution δf_2.

δf_2 ($^\circ$)	$E_{\delta z}$ (%)
0.5	3
1	6
2	9
4	13

3. SINGLE ROBOT EXPLORATION VIA 3D LASER BASED BUBBLE SURFACES

One of the central questions in the autonomous exploration of unknown environments is determining where to go - given what the robot knows about the world [28,29]. Initially the robot knows nothing except what it senses from where it's standing, so that the movement directions cannot be handed to the robot in a pre-meditated manner. Rather, it operates in reflexive mode based on sensory feedback from the environment and its so far acquired knowledge - which it may simultaneously use to construct a partial map. Hence, as it starts exploring, at each location, it will determine a movement direction. The movement direction should be such that it should point the robot to unexplored territory while being accessible -namely a path must exist from the robot's current base to it [29].

The contribution of this chapter is to consider the problem of exploration with topological maps. In particular, we use bubble space representation where the nodes of the topological map are represented as bubble surfaces [5]. We assume that there is no a priori map. Node candidates are generated concurrently as the robot is exploring its surroundings. The novelty of this approach is that the nodes of the topological map (bubble surfaces) are used in generating movement directions associated with unexplored regions.

3.1. Related Literature

Exploration algirothms vary depending on whether the environment is known or unknown [30]. In a known environment, the environment model is given and motion planning becomes geometric programming [31] or graph search [32]. In an unknown environment, such knowledge is not available and the robot obtains information as it is moving around and collecting local information via its sensor. The map must be incrementally computed as new regions in the environment are explored [30]. In this

chapter, we consider the latter. Exploration of unknown environments with mobile robots have been studied intensively. In all, processing is based on local information which implies that sensing is integral to navigation [30]. Thus, the approaches must schedule the sensor operations. Most of these approaches guide the robot to the closest unexplored area. These techniques mainly differ in the way the environment is represented - which in turn affects the determination of closest unexplored areas.

Traditional approaches are based on the accumulation of accurate geometrical descriptions of the environment [33]. Most exploration strategies work with metric maps that use either two-dimensional grid-based or three-dimensional voxel representations [34]. One simple approach is to use a random selection mechanism (random walk) [35] or greedy mapping [36,37]. A competitive algorithm for the case of a polygonal room with a bounded number of obstacles in it is presented in [31]. A more efficient approach is accomplished in [29] where the robot moves to the closest frontier point which is determined based on laser-limited sonar and evidence grids. There has been many variants of the frontier based approach where the robot is made to approach the boundary between explored and unexplored space such depending on performance criteria [38,39]. Each frontier is evaluated based on the expected number of unknown cells the robot can see from the frontier as well as the distance from the robot [40]. Thus the exploring robots choose the frontier which will provide the highest utility (information gain minus driving cost) rather than simply the closest frontier. The robot moves as to maximize the number of viewable frontier cells [41], information gain using [39] or Rao-Blackwellized particle filters [42]. This idea is generalized to three-dimensional mapping where a frontier voxels are those that lie between explored and unexplored space [34]. An alternative approach to metric space maps is to use configuration space based approaches. A next-best view type algorithm is proposed to guide the robot through a series of locations with high expected amount and quality of the information [43]. The robot plans the next sensing action to maximally reduce the expected C-space entropy [44]. The exploration is defined by the incremental expansion of a sensory exploration tree in the configuration space [45] based on frontiers of explored regions - motivated by rapidly exploring random trees [46].

While grid-based and configuration space methods produce accurate metric maps, their complexity often prohibits efficient planning and problem solving in large-scale environments [47]. Alternatively, exploration is done with topological maps where the unknown environment is modelled as a graph - albeit unknown - with more efficiency [48]. In the spatial hierarchies model, the nodes correspond to distinctive places and arcs correspond to travel paths [33]. Exploration is based on developing algorithms for traversing this graph efficiently. In the semantic hierarchy of spatial representations, the exploration strategy consists of moving into an open direction, following a path with a control strategy, hill-climbing in case of detecting a distinct place until reaching a local maximum that defines being at another distinctive place [33]. The topological map is built as a side-effect of motion through this transition graph.

In most approaches, it is assumed that the robot is able to recognize a node, enumerate edges incident on the current node and traverse edges without specifying how they are achieved. These approaches can further be categorized into two groups depending on the robot's inability to distinguish vertices and edges from each other. In the former case, a common approach is to endow it with markers. The graph is explored via adding new vertices having outgoing edges that lead to unknown places using multiple markers [49, 50]. Reducing the number of markers, an unknown, undirected planar graph is learned in time linear the size of the graph by a robot equipped with one marker [51]. In the map verification problem, an efficient algorithm enables a robot - given a map of the world with its pose indicated on the map, - to find out whether this map is correct with the aid of one edge markers [32]. This idea is generalized to directed graph models in [52]. In the latter case - namely when the robot is able to distinguish visited nodes and edges, but does not know the endpoints of the unvisited edges, the robot is able to explore with a bounded efficiency when the deficiency is bounded [53]. This result has been extended for dense graphs [54] and improved in general depending on the deficiency of the graph [55, 56]. As stated, all these approaches require the robot to recognize nodes and their associated edges. In general, both are assumed to be achieved separately from the graph search process. In this chapter, we present a novel approach that integrates the two – namely the generation and/or recognition of nodes and their associated edges is achieved simultaneously with graph exploration.

Here, the nodes of the topological map are used to determine the unexplored directions. The robot expands its topological map by adding the places reached and the nodes generated after navigating in these directions.

3.2. Where to move next?

Each environment is represented by a topological map - which is a connected graph g whose nodes are represented in bubble space [5] and whose edges are labelled. Let $\mathcal{S} = \{1, \ldots, N_t\}$ be the set of bubble surfaces – each from one different base - and g denote an undirected graph defined on $\mathcal{S}$. As the environment is completely unknown a priori, initially $N_t = 0$ and $g = \emptyset$. At each location, its laser starts providing sensory data feedback. It generates a bubble surface $B(x, t)$ and the corresponding bubble descriptor $I(x, t)$ - which is then added as a node S_i. Let g_t denote the map that exists at time t. Let E denote the set of edges. Two nodes i and j have an edge – namely ij in g if and only if when the robot can navigate from one to the other, the place changes from node i to node j directly. The label of each edge includes the navigation direction between the two nodes. If where $ij \in E$, $g + ij$ denotes the graph obtained by adding the edge ij to the existing graph g and $g - ij$ denotes the graph obtained by removing the link ij from g. Each edge ij in E is also associated with the label $b(t_{k+1}) - x(t_k)$. The robot decides where to move depending on S_i and the current graph g_t - as explained in the sequel. As the robot is exploring its world, N_t increases accordingly.

Consider an exploration task starting at time t_0. Assume the robot is at base point $x(t)$ at time $t \geq t_0$ with a partial topological map g_t. Its exploration related decision-making has two aspects to it:

- Local exploration: Locally, it should decide which direction to move.
- Global exploration: Globally, it should try to navigate so that all unexplored regions in the environments are visited at least once.

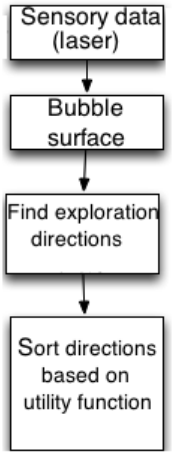

Figure 3.1. Local exploration algorithm.

3.2.1. Local Exploration - Finding Directions

At each new base, the robot finds a set of candidate directions for exploration using the 3D laser bubble surface associated with the respective node. These are then sorted with respect to their utility and the robot moves in the unexplored direction with maximal utility as shown in Figure 3.1.

Consider a particular bubble surface point $b = [x\, f]^T$ with $f = [f_1\, f_2]^T$. If the tilt angle $f_2 = 0$, then b is associated with a viewing direction that is parallel to the ground. If $f_2 < 0$, then b is associated with a viewing direction that is looking downwards from its horizon by an amount that is given by the magnitude of f_2. On the other hand, if $f_2 > 0$, then the robot is looking upwards from its horizon. As the part of the bubble surface with $f_2 > f_H$ is considered to be not accessible since we assume terrestrial navigation[1] , it considers only the part of bubble surface with viewing direction $-\frac{\pi}{2} \leq f_2 \leq f_H$.

The candidate exploration directions are found via transforming this part of the bubble surface via thresholding and applying connected component analysis on the transformed surface in order to determine regions for potential exploration. The threshold value $\tau_1 : \mathcal{B} \rightarrow R^{>0}$ plays a critical role. It is defined adaptively based on the surface

[1]For aerial robots with flying capabilities, this constraint will no longer be applicable.

via considering the lines of constant latitude f_2 or parallels as:

$$\tau_1(b) = \mu_{\rho,f_2} + \sigma_{\rho,f_2}$$

where average surface deformation μ_{ρ,f_2} along each parallel and standard deviation σ_{ρ,f_2} of this deformation are defined as:

$$\mu_{\rho,f_2} = \frac{1}{|\mathcal{F}_1|} \sum_{\substack{f_2'=f_2 \\ b' \in Im(h(x))}} \rho(b',t)$$

$$\sigma_{\rho,f_2} = \sqrt{\frac{1}{|\mathcal{F}_1|} \sum_{\substack{f_2'=f_2 \\ b' \in Im(h(x))}} (\rho(b',t) - \mu_{\rho,f_2})^2}$$

In particular, each bubble surface point b that is closer than a threshold distance $\tau_1(b)$ is considered to be explored - namely if $\rho(b,t) < \tau_1(b)$. Inaccessible regions are determined depending on the robot's physical capabilities and obstacles along the way. If there is an obstacle along the way, then the robot cannot move in this direction. This can be detected by determining if there are any bubble surface points b' having the same pan angle $f_1' = f_1$, but with lower tilt values $f_2' < f_2$ and $\rho(b',t) < \tau_1(b)$.

Thus, with this thresholding, the bubble surface is binarized as:

$$B'(x,t) = \left\{ \begin{bmatrix} f \\ \kappa(b, \rho(b,t)) \end{bmatrix} \mid \forall f \in \mathcal{F} \text{ and } b = [x\ f]^T \right\} \tag{3.1}$$

with

$$\kappa(b,x) = \begin{cases} 0 & x \leq \tau_1(b) \\ 0 & f_2 > f_H \\ 0 & \exists b' \in Im(h(x)) \text{ s.t. } f_2' < f_2 \\ & \rho(b',t) < \tau_1(b) \\ 1 & \text{otherwise} \end{cases}$$

Next, connected component analysis is performed on the binarized bubble surface $B_i'(x,t)$. Let $\mathcal{A}(t) = \{A(t,1), \ldots, A(t,M)\}$ be the set of resulting connected components. Each connected component $A(t,i) \subset \mathcal{F}$ corresponds to one set of viewing directions in which distance of objects are relatively further and is thus an area of interest for exploration. Hence, it represents a potential direction of exploration as defined by the map $u : \mathcal{A}(t) \to \mathcal{B}$:

$$u(A(t,i)) = \frac{1}{|A(t,i)|} \sum_{b \in A(t,i)} f$$

Let b^* be the bubble surface point that corresponds to a potential direction of exploration, $u(A(t,i))$, at robot's current base x :

$$b_i^* = \begin{bmatrix} x \\ u(A(t,i)) \end{bmatrix} \tag{3.2}$$

Let $\nu : \mathcal{A}(t) \to R$ be the maximal horizontal extension of each component along $f_2 = 0$:

$$\nu(A(t,i)) = |\{b \in A(t,i) \mid f_2 = 0\}|$$

If the extension $\nu(A(t,i))$ of a given $A(t,i)$, exceeds a pre-specified threshold τ_2, the pan coverage of $A(t,i)$ is extensive and can be broken into regions with smaller pan coverage of ν^*. On the other hand, if $\nu(A_{t_i})$ is less than a pre-specified threshold τ_3, then this indicates that the associated region is too narrow and the robot cannot pass through it. Hence, those components are removed from $\mathcal{A}(t)$.

If the cardinality of the resulting $\mathcal{A}(t)$ is greater than 1, this means that there are several potential movement directions. These are ordered via defining a utility function

$\varphi : \mathcal{A}(t) \rightarrow R^{>0}$. In our case, the utility function measures the distance of that region weighted by its extension in the pan direction as:

$$\varphi(A(t,i)) = \nu(A(t,i)) * \rho(b_i^*, t) \tag{3.3}$$

The robot adds an unexplored edge for each potential direction. The edge is labelled with its direction $u(A(t,i))$ and utility value $\varphi(A(t,i))$. The robot moves in the direction with maximal utility.

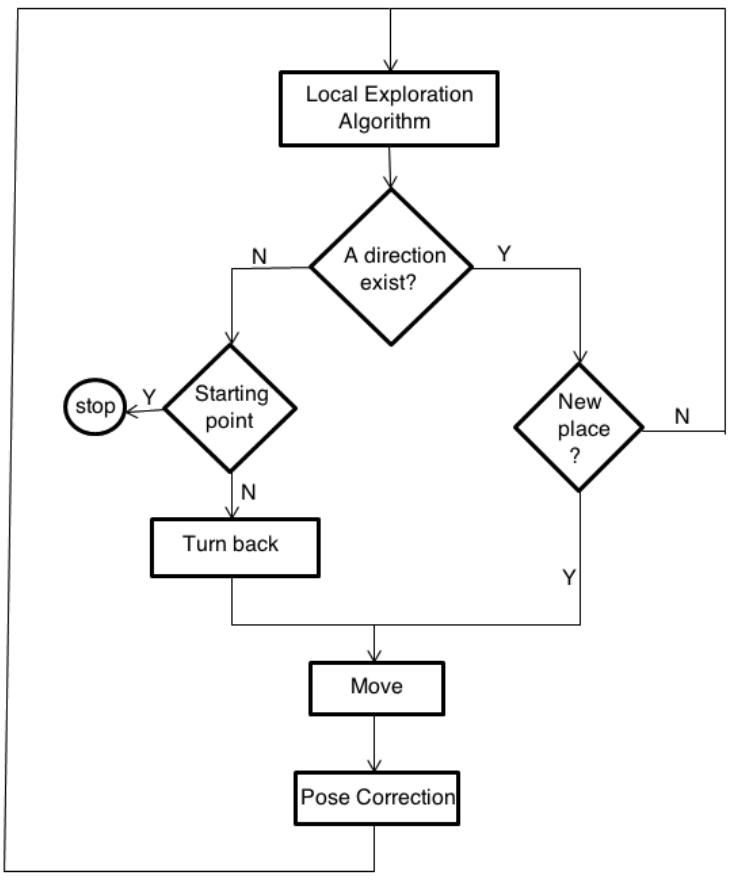

Figure 3.2. Global exploration algorithm.

3.2.2. Global Exploration

At each stage of the exploration, the robot knows a proper subgraph g_t consisting of traversed nodes and edges. In other words, the robots knows all the visited nodes and traversed edges and can recognize them when they are encountered again. However,

the robot does not know:

- How many nodes or edges are there in the graph g corresponding to the whole environment
- Where each edge that is not traversed leads to

The robot needs to explore the complete g. This problem is known as the classic graph exploration problem [53]. Global exploration requires repeatingly selecting an outgoing edge from its current node and traversing it. Our global exploration strategy is depth-first traversal [54] as shown in Figure 3.2. Each node is associated with a set of unexplored directions - namely unexplored edges. The robot chooses moving in the utility maximizing direction among the set of unexplored directions. If a robot cannot find any directions in a node, it is stuck. When it is stuck, it turns back and moves to the base point it came from. It then checks to see if there remains unexplored directions at this node. This is repeated until every place that is strongly connected is explored or a node with unexplored direction is reached. The exploration is finished when robot comes the first generated node(mothernode) and can not find any new directions that have not been moved to.

3.2.3. Integrated Exploration

While in exploration, the robot applies both local and global strategies. A sample scenario is shown in Figure 3.3. This scenario is conducted in a Webots simulated indoor environment [27]. Terrain has been designed with a slippery surface that cause a 10% pose error. Gaussian encoder noise is also added to robot encoder in order to make simulations similar to real time. The robot starts its exploration via local exploration concept and moves as to generate nodes 1-4. At node 4, it can not find any movement direction. Hence, it turns back until coming to a node with and unexplored direction. This happens at node 2 and the robot moves move along this direction and generates nodes 5-6. At this node, again there are no unexplored direction and again it turns back. As it cannot find any unexplored directions until node 1, it comes back to where it has started. At this node, it finds an unexplored direction and moves along

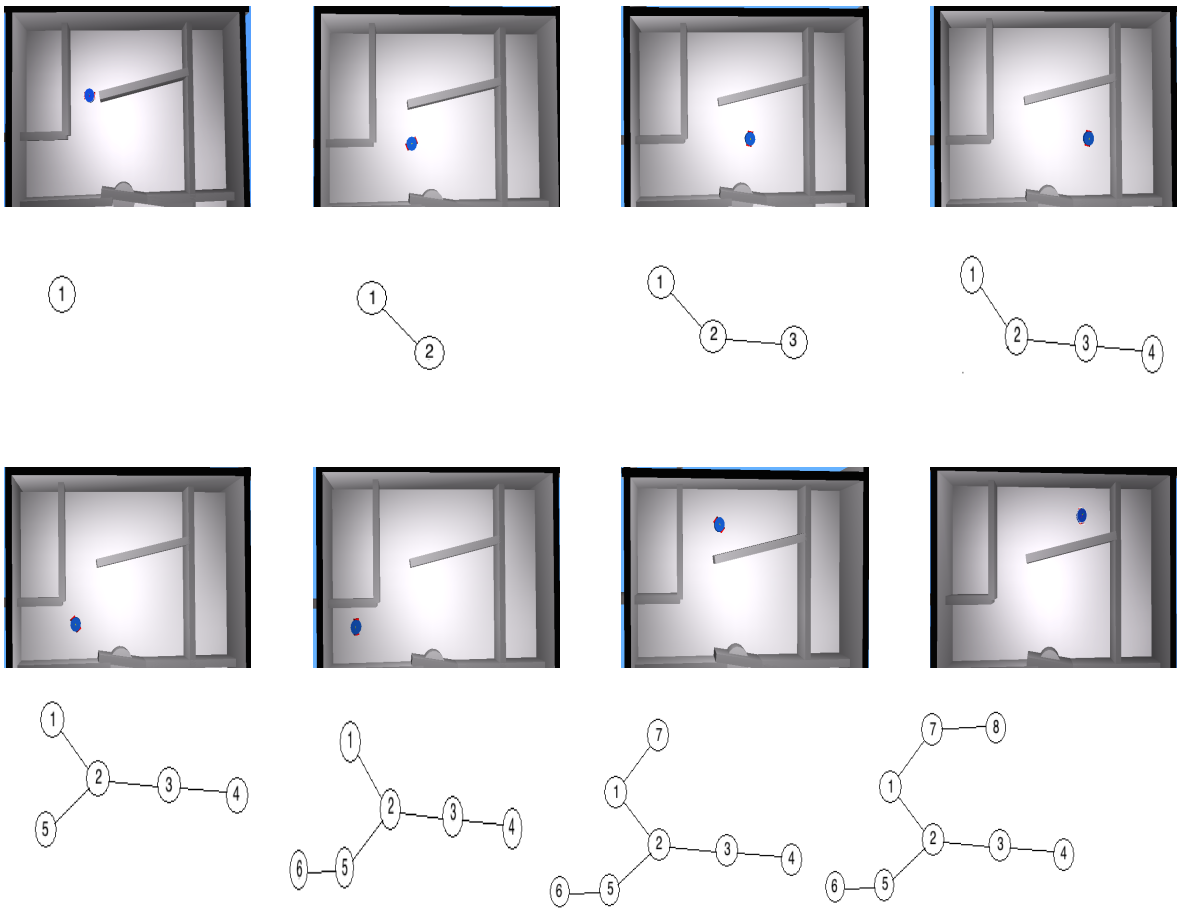

Figure 3.3. The graphs indicate the partial topological map generated so far where node numbers i indicate visit times t_i where $t_i > t_{i-1}$.

this direction until node 8. When it is stuck at this node, it turns back. In this case, it comes back to node 1. There are no remaining directions to explore. The mission is completed.

3.3. Pose Correction & Loop Closure

After the robot determines an exploration direction, it moves in this direction for a prespecified distance. However due to the environmental factors and encoder errors, robot may not reach its desired target position. Since the robot uses odometric data to determine $b \in \mathcal{B}$ and $b' \in \mathcal{B}$, it is crucial to minimize odometric error in order to prevent the accumulation of this error. When the robot navigates around, the base and the bubble surface both change accordingly. Suppose that the robot is at $b \in \mathcal{B}$ at

time t and goes to $b' \in \mathcal{B}$ at time $t + \delta t$. Let

$$b = \begin{bmatrix} x \\ f \end{bmatrix} \quad \text{and } b' = \begin{bmatrix} x' \\ f' \end{bmatrix} \tag{3.4}$$

As the robot knows how much it has planned to move, pose correction is based on comparing the expected bubble surface with the actual bubble surface generated after the completion of the motion.

Two different transformations must be applied on the bubble surface. First, feature values change their location on the bubble due to robot navigation from $o \in Im(h(x))$ to $o' \in Im(h(x'))$. Hence, the bubble surface B_i needs to be transformed to B_i' depending on the robot's movement. This is accomplished in two steps: First, it removes all the local bumps introduced previously up to $t + \delta t$ at each point $o \in Im(h(x))$. Next, the transformation map $d_{b \to b'} : Im(h(x)) \to Im(h(x'))$ between the previous and the transformed bubble surfaces need to be computed so that the corresponding bubble points are determined as $o' = d_{b \to b'}(o)$. The subscript $b \to b'$ is used to denote the dependency of the transformation on the previous and current base points b and b' respectively. Note that depending on the environment and the transformation, the map $d_{b \to b'}$ can be one-to-one, one-to-many or many-to-one. Finally, all the removed bumps are moved to their corresponding $o' \in Im(h(x'))$ as: Hence, each bubble surface B_i, $i = 1, \ldots, N_v$ is transformed as follows:

$$\begin{aligned} \rho_i(o, t + \delta t) &= \rho_i(o, 0) \\ \rho_i(o', t + \delta t) &= \rho_i(o, t) \end{aligned} \tag{3.5}$$

Secondly, new observations are made at the newly arrived viewpoint and the bubble surfaces should be deformed as defined by Equation A. Of course, this formulation of the transformation maps ignores problems related with odometry data, sensor noise, changes in the lightness of the scene and the scaling effect previously explained. In reality, the model must be revised in order to accommodate these issues. Finally, the bubble transformation is applicable only if the transformation map $d_{b \to b'}$ is determined.

Next, this is investigated for different type of robot movements.

3.3.1. Heading Correction

First, let us consider a robot movement involving only rotation by $\delta\alpha \in S^1$ as seen in Figure 3.4. Due to odometric errors, the robot may not rotate by the amount it planned. Even if the error is small, it can accumulate in every movement so the robot can not localize itself correctly and this leads to wrong mapping.

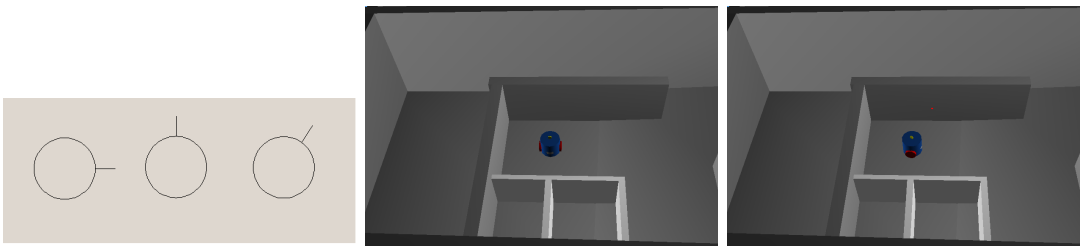

Figure 3.4. Left: Heading correction; Center: Robots' initial pose before exploration; Right: Robot determines its orientation.

Note that the base vectors x and x' will differ only in their 3^{rd} components – as this represents the robot's heading. Hence, the transformation between the base points can be expressed as:

$$x' = x + \begin{bmatrix} 0 \\ \delta\alpha \end{bmatrix}$$

Consider $o = \begin{bmatrix} x_o \\ f_o \end{bmatrix} \in Im(h(p))$. The deformation map $d_{b\to b'}$ is defined as

$$d_{b\to b'}(o) = \begin{bmatrix} x_o + \begin{bmatrix} 0 \\ \delta\alpha \end{bmatrix} \\ f_o + \delta f \end{bmatrix} \tag{3.6}$$

where $\delta f = [\delta\alpha\ 0]^T$ since a heading change is equivalent rotation of the pan coordinate system. Note that δf is identical for all the bubble surface points. Furthermore, it is

independent of the distance of the objects to the robot.

Let $B_i(x,t)$ and $B_i(x',t+\delta t)$ be the two consecutive bubbles generated before and after the rotation of the robot. It is clear that the only difference between x and x' is the rotation.

$$x - x' = \begin{bmatrix} 0 \\ 0 \\ \delta\alpha \end{bmatrix} \tag{3.7}$$

Let $B_i(x'_m, t+\delta t)$ be the bubble rotated $\delta\alpha$ amount along z-axis. The actual, $\delta\alpha$ minimizes the following expression:

$$\delta\alpha = \min_{\delta\alpha}(B_i(x,t) - B_i(x'_m, t+\delta t))^2 \tag{3.8}$$

Since $\delta\alpha$ is computed from Equation 3.8, the robot knows its actual rotation amount.

3.3.2. Translation Correction

After the the rotational correction, the robot starts to move in this direction. Again due to odometric errors, robot may move more or less than desired amount. Lets call the planned amount of translation is δc and lets call the actual translation as $\delta c'$. If we determine $||\delta c'||$, we can clearly find the amount of translation error and offset accordingly. After the robot has moved by $||\delta c'||$, it creates another bubble surface $B_i(x_2, t_2)$. Note that this bubble surface is the translated version of the $B_i(x'_m, t+\delta t)$. Assuming the robot moves on a flat surface, the difference of these bubbles at $f_1 = 0$ and $f_2 = 0$ gives the actual amount of translation. Let l_1 and l_2 are the sensor values at f_1=0 and f_2=0 for the bubbles $B_i(x'_m, t+\delta t)$ and $B_i(x_2, t_2)$ respectively. The actual

translation is equivalent to the difference of l_1 and l_2 .

$$l_1 = \rho(x'_m, t) \tag{3.9}$$

$$l_2 = \rho(x_2, t_2) \tag{3.10}$$

$$\delta c' = \rho(b', t + \delta t) - \rho(b, t) \tag{3.11}$$

As $\delta c'$ can be predicted from Equation 3.11, the robot knows its actual translation amount.

A sample case of pose correction is as shown in Figure 3.5. For slippery surface with varying amounts of encoder noise, it is observed that the actual trajectory deviates from the planned trajectory. The deviation increases with the amount of encoder noise - as expected. With pose correction, actual trajectory becomes almost identical to the planned trajectory. It is observed that this is achieved regardless of the amount of encoder noise.

3.3.3. Position Update for Ramped/Inclined terrain

Another reason for position error is moving on ramped or inclined terrain. In such regions, robot should detect and estimate the place and slope of the ramp and update its new position (in our case location of the bubble namely c). The prediction of the slope of the ramp/incline in an environment(θ) and the distance between ramped/inclined part and robot(λ_1) has been proposed in Chapter 2 and in [57]. Assume that the robot moves by δc amount in the direction of a terrain with slope θ and distance λ_1. The new location point c' is;

$$c' = \begin{cases} c + \delta c & \text{if } \|\delta c\| < \lambda_1 \\ c + \lambda_1 \frac{\delta c}{\|\delta c\|} + R(\theta)(\delta c - \lambda_1 \frac{\delta c}{\|\delta c\|}) & \text{otherwise} \end{cases}$$

where the matrix $R(\theta)$ is a 2×2 rotation matrix.

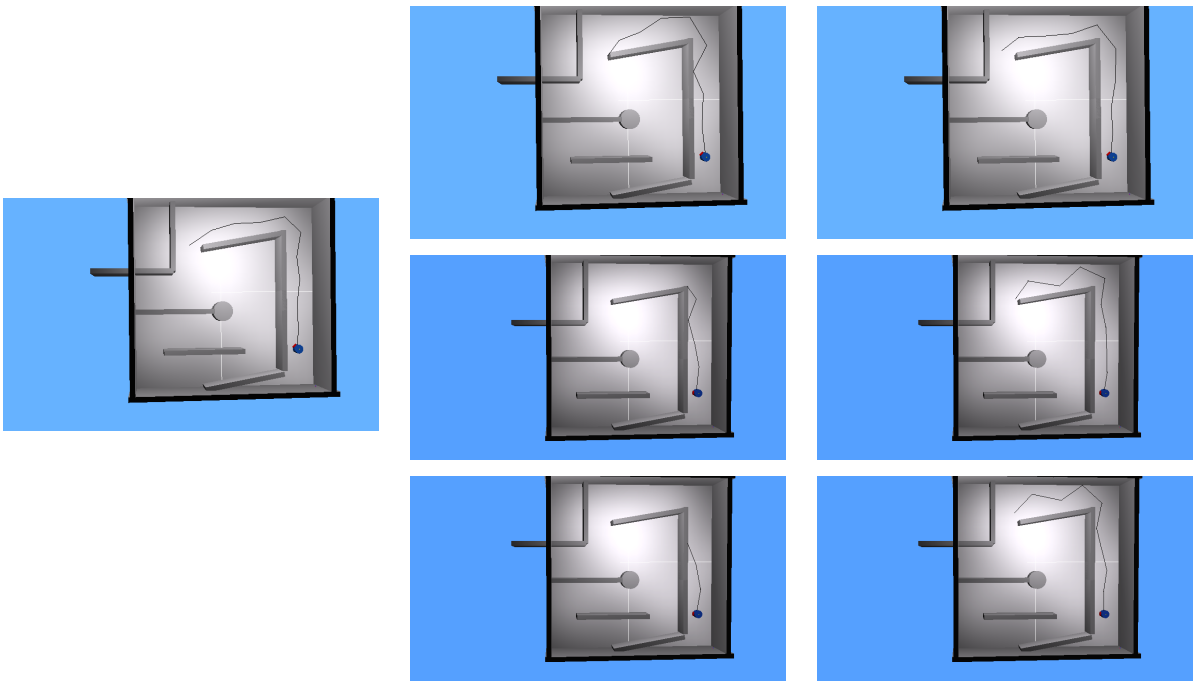

Figure 3.5. Left: Planned trajectory without noise; Top Left: Trajectory with 10% noisy encoder readings; Top right: Trajectory with pose correction; Center left: trajectory with 20% noisy encoder readings; Center Right: Trajectory with pose correction; Bottom Left: Trajectory with 20% noisy encoder readings; Bottom Right: Trajectory with pose correction.

$$R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

3.3.4. Loop Closure

In order to detect visited place, each node is associated explicitly with its base point x based on odometric information. One can readily calculate $c(t+\delta t)$ and $x(t+\delta t)$ by using the previous base vector $c(t)$ and odometry data ($\delta\alpha$ and $||\delta c||$) .

$$\alpha(t + \delta t) = \alpha(t) + \delta\alpha \tag{3.12}$$

$$c(t+\delta t) = c(t) + ||\delta c|| \, [cos(\alpha(t+\delta t)) \; sin(\alpha(t+\delta t))] \tag{3.13}$$

$$x(t+\delta t) = \begin{bmatrix} c(t+\delta t) \\ \alpha(t+\delta t) \end{bmatrix} \tag{3.14}$$

In order to detect the visited nodes, let $x(t+1)$ be the new target base which is determined by Equation 3.14. If the location component of x(t+1) namely c_{t+1} is similar with the previous location vectors, this position is regarded as a visited place. The similarity is measured by Euclidean distance with a predetermined threshold τ_p. Assume that robot finds n candidate nodes for time t+1 and base point b. Let, $r(t+1, b)$ is the node assigner function for any time t and base point b.

$$r(t+1, b) = \begin{cases} i & d(c_{t+1}, c_i) \leq \tau_p \\ N_{t+1} & \text{otherwise} \end{cases} \tag{3.15}$$

Equation 3.15 makes possible to detect the visited nodes and also enable to do a more effective exploration algorithm. In particular, the robot eliminates some movement directions that correspond to the previous visited node positions. This makes the exploration process faster and more effective.

3.4. Simulation Results

The proposed approach has been tested with a variety of different settings including Canadian Planetary Emulation Terrain 3D Mapping Dataset and Jaguar robot.

3.4.1. Canadian Planetary Emulation Terrain 3D Mapping Dataset

Next, we apply our approach on the Canadian Planetary Emulation Terrain 3D Mapping (CPET3DM) dataset [1]. In particular, we use `p2at_met` dataset that is generated in the UTIAS indoor rover test facility. This dataset consists of 3D laser scans that are obtained from 102 different locations with a grid-like structure whose dimensions are 60mx120m. At each location c, the robot has 3D laser data with viewing direction $f_1 \in [0, 360]^\circ$ and $f_2 \in [-16, -5]^\circ$ with increments $\delta f_1 = 0.3^\circ$ and $\delta f_2 = 0.5^\circ$ respectively. For generating a bubble surface, the robot choses the dataset from the most nearby data collection point to itself so a sampling error is inevitable. However, since the data collection points are almost uniformly grid-like dataset with small grid dimensions, it is a convenient dataset for exploration.

The robot is made to start at varying starting points and a sample exploration path is shown in Figure 3.6. The starting positions are labelled as a cross. As an example, in the left part of the Figure 3.6, the robot starts exploration at [-17.4 10.5]. This point is labelled as node 1 in corresponding map as shown left bottom of the Figure 3.6. The robot uses proposed method to find its new direction and moves in this direction for 20 meters. The nodes associated with all the places visited are numbeted in an increasing manner. Thus, the robot moves incrementally from node 1 to node 16. At this node, it can not find any unexplored direction so it turns back until it is able to do so. At node 15, robot finds such a direction and moves to nodes 17 and 18 respectively. At node 18 robot can not find any unexplored direction to move and then it turns back again. This continues until node 1. At node 1, the robot finds an unexplored direction and move there which is labelled as 19. At node 19, there are no unexplored directions and the robot turns back to node 1. At this initial node, as, there are no unexplored directions and it finishes its exploration task. A similar reasoning is followed even if the robot starts at different position as shown in Figure 3.6. In each case, the exploration path is different but in all cases robot cover the whole environment. These paths are different while covering roughly the same regions. This can be attributed to the fact that bubble surfaces at initial positions are different. Therefore, the robot moves in different directions which leads to different paths.

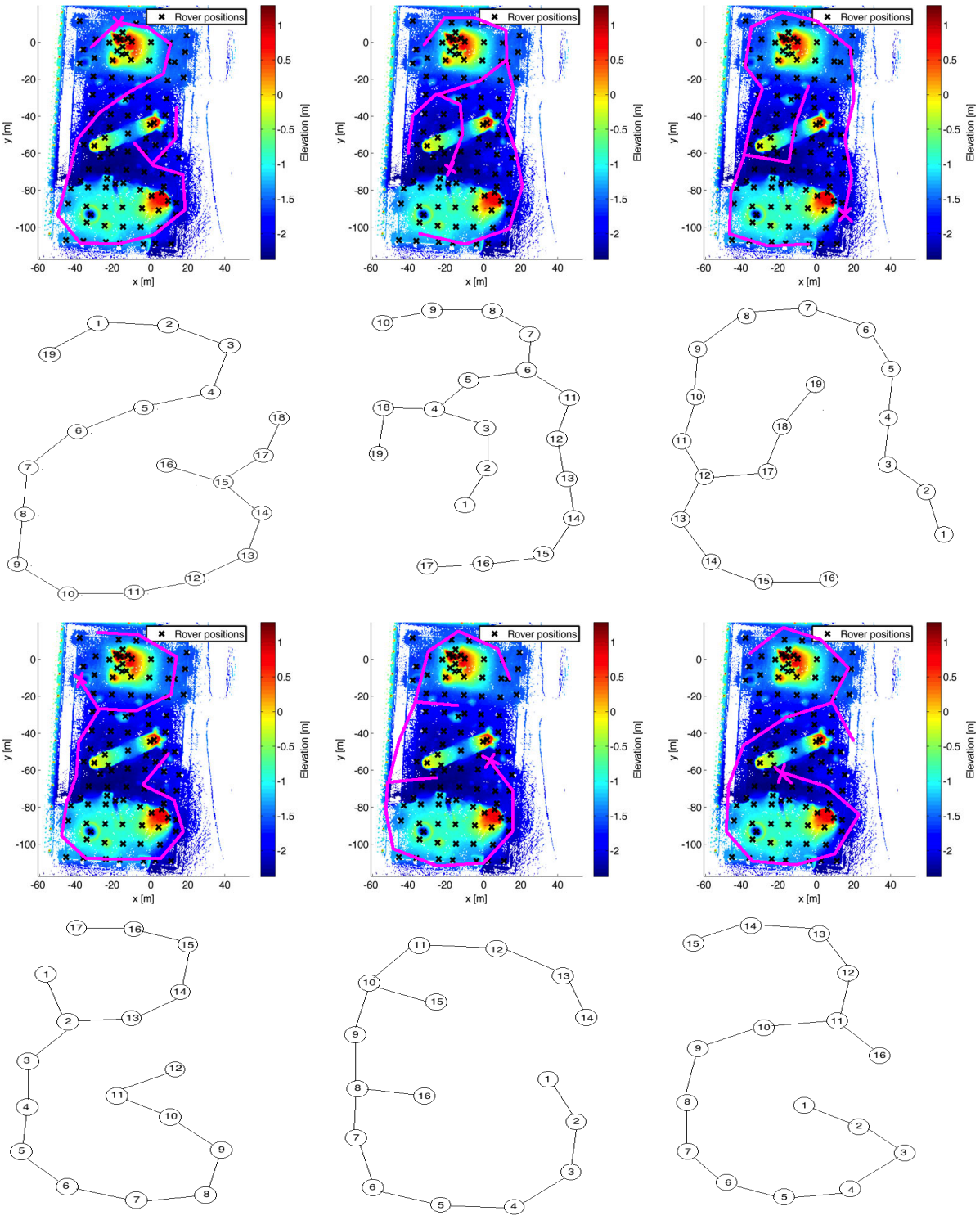

Figure 3.6. Travelled paths and graphs Top Left: Initial $c = [-17.410.5]^T$; Top Center: Initial $c = [-19.2 - 76.4]^T$; Top Right: Initial $c = [13.4 - 93.4]^T$; Bottom Left: Initial $c = [-38.411.0]^T$; Bottom Center: Initial $c = [-0.4 - 44.5]^T$; Bottom Right: Initial $c = [-14.9 - 76.0]^T$.

Table 3.1. CPET3DM results with different starting points.

Starting position c	Area coverage (m^2)	# Places	Path length (m)
(-17.4 10.5)	4850	36	720
(-19.2 -76.4)	4620	36	720
(13.4 -93.4)	4460	36	720
(-38.4 -11.0)	4070	32	640
(-0.4 -44.5)	4670	30	600
(-14.9 -76.0)	5120	30	600

Table 3.1 presents the total covered area and the length of the travelled path for each exploration mission. In some parts, the dataset does not involve the sensor values at higher than $f_2 = -5$. Due to this limitation, the area is found by using the bubble values at $f_2 = -5$. This leads to find a smaller value than the actual covered area. It is observed that changing the starting position affects area coverage slightly while travelled path lengths also vary by 10% . The variability in area coverage is attributed to the varying number of places that are explored as seen in Figure 3.6. Furthermore, the actual locations of the nodes are important. For example, if the robot is near to a ramp, the corresponding area coverage decreases accordingly.

3.4.2. Jaguar Robot

The last set of experiments are done with the Jaguar robot as shown in Figure 3.7(top left). The Jaguar robot is a tracked robot endowed with two cameras, 2D laser scanner, encoders and IMU. A special user interface based on QT and ROS has been designed for operating the robot in teleoperation and autonomous modes. For the interested reader, the technical details of the robot are presented in Appendix B.

First, an extensive dataset is collected via teleoperating the robot around the floor as shown in Figure 3.7 (top right) and have it collect data at 75 different locations. The dataset consists of camera, laser, gps, encoder and imu data. At each location c, the robot has 2D laser data with viewing direction $f_1 \in [0, 360]^\circ$ with increments $\delta f_1 = 0.166^\circ$ and and $f_2 = 0$. In fact, robot's laser scanner is capable to pan between $f_1 \in$

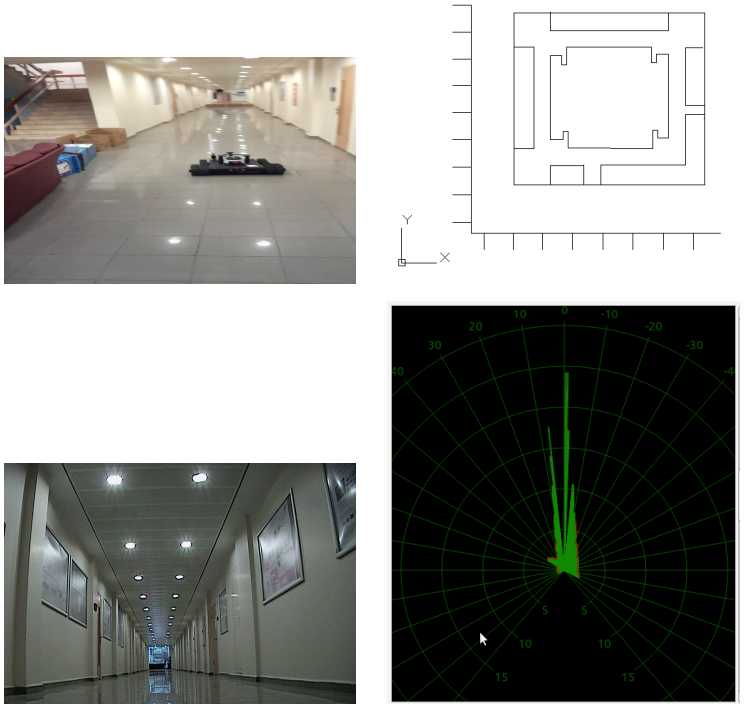

Figure 3.7. Top Left: Jaguar robot in the floor; Top Right: Floor plan; Bottom Left: A sample front camera image; Bottom right: A sample laser data.

$[-90, 90]^{\circ}$, we turned robot 180 degrees to make laser scan omnidirectional. Sample camera and laser data are as shown in Figure 3.7(bottom left) and Figure 3.7(bottom right) respectively.

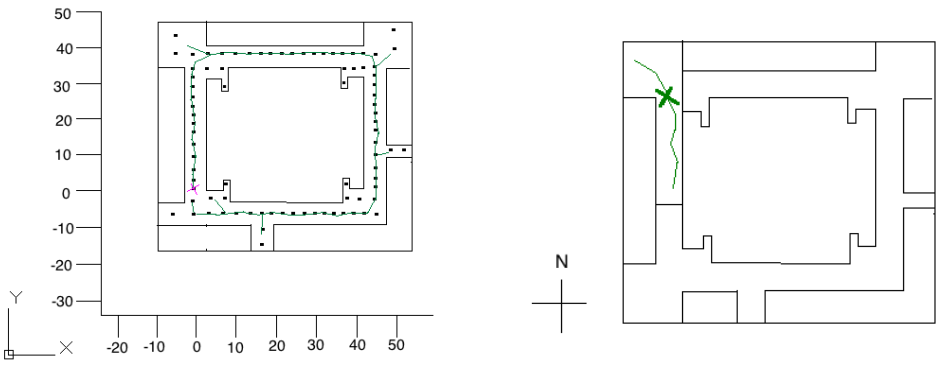

Figure 3.8. Left: Exploration path with dataset; Right:Real-time exploration path under autonomous mode.

The algorithm is then applied with this dataset and the resulting exploration path is as shown in Figure 3.8(left). The robot starts exploration at a base as indicated with the pink cross. It then navigates along the corridor until coming back to its starting

point. Then, it turns back to explore interesting places that are detected, but not explored. While turning back, robot goes to some of the offices and finishes exploration by turning back to the initial position. the details of the exploration stages are shown in Figure 3.9 .

Finally, this approach is implemented in real-time on the Jaguar robot and is put to test. In this mode, the robot is completely autonomous in determining its exploration path – using the approach as explained. The given environment and the robot path is given in Figure 3.8(right). The robot goes to the northern part of the environment. When it realizes that there are no unexplored directions, it turns back and moves to the southern part and covers the whole environment.

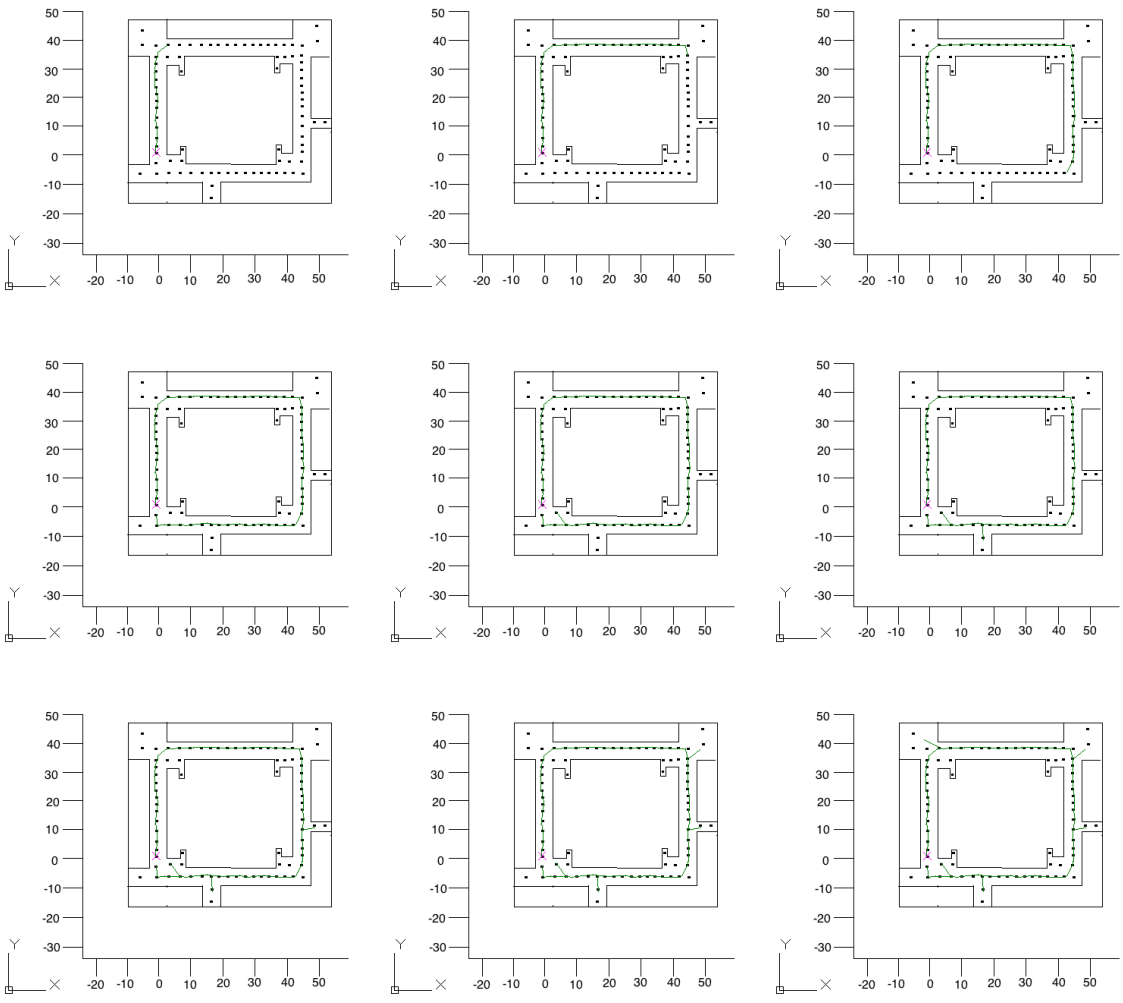

Figure 3.9. Stages of the Exploration paths with Jaguar robot.

4. MULTIROBOT EXPLORATION VIA 3D LASER BASED BUBBLE SURFACES

This chapter is about autonomous exploration of a priori unknown environments with multiple robots. Each robot builds a topological map about the environment that it covers which are merged to construct the map of the environment. Initially, each robot knows nothing except what it can sense from where it's at and the initial relative positions of the other robots. Each robot starts exploration in a manner similar to that described for a single robot. However, here, robots exchange information at each base. In particular, they inform each other of their current positions. By using this data each robot knows where the other robots are roughly and makes movement decisions accordingly. Thus, the robots are able to cover the whole area to be explored with minimal overlap.

4.1. Related Literature

More recent work in exploration has started considering exploration missions with multiple robots. It is expected that using multi-robot systems instead of a single one would improve exploration performance [50, 58, 59]. First, the amount of coverage in a given amount of time will increase. Similarly multi-robots cover the given environment faster than a robot. Furthermore, there is a possibilty of verifying information collected by each robot.

Approaches to multirobot exploration can be categorized depending on two factors : maps used and the nature of decision making. As discussed for the single robot case, maps can be metric or topological. The second dimension refers to how the robots decide where to go. This decision-making can be centralized or decentralized. In centralized strategies, the robots' motions and coordinations are determined by a central agent or a base station. In decentralized approaches, each robot makes its own decision for exploration. These decisions can depend on various criteria including the

cooperative information gathering from other robots. Even though the decision of the robot may be related to the information coming from other robots, robot makes its own decision independently. In all these work, communication among the robot is critical to task performance.

As a metric and centralized strategies, a central station determines the frontier cells based on a utility value for each location in [60] where the utility value depends on the expected travel cost and the information gain. The information gain is the estimated number of unknown map cells within a radius at the location. When one robot is assigned to some location, the information gain of the location cells is decreased. A method using a Monte Carlo localizer and maximum likelihood on grid maps is based on using maximum likelihood function determines the best alignment of laser data [61]. A team leader merges the maps and shares the grid map with the rest of the team.

As a metric and decentralized strategies, frontier-based exploration has been proposed [62]. The definition of frontiers is basically the boundaries between the explored and unexplored areas. An utility function is derived for less cost and much exploration and robots utilize that function. Robots exchange their grid maps and continuously update their own map by merging the map received with their local maps. Frontier concepts have been used in many projects such as [60, 63, 64]. In [65], when two or more robots come in each other's communication range, they merge their local maps and choose a leader which is responsible for building a complete map that represents the data collected by all robots in the communication range, and broadcasts the map frequently to all the robots in that range [65] .

One of the earlier work in centralized and topological strategies [66] includes one moving and two stationary robots, robots behave according to a centrally agreed plan and triangle formed by two stationary and one moving robot is considered as free space. The formed triangles are connected to graph and map is formed. It is proved that this approach is more successful than single robot exploration but in that approach many robots remain stationary [66].

In [67], a topological and decentralized approach which is a randomized strategy for cooperative exploration based on SRT(Sensor based Random Tree) concept is proposed. The method entails two decentralized cooperation mechanisms at different levels. The first simply consists in an appropriate definition of the local frontier, by which each robot plans its motion towards areas that appear to be unexplored by the rest of the team on the basis of the availible information. The second allows a robot that has completed its individual exploration phase to support the others in their task. Another example of decentralized and topological strategies is "Sensor Based Random Graph" (SGR) [68], the nodes of the SRG represent view configurations that have been visited by at least one robot and these nodes are connected by arcs that represent safe paths which is equivalent to edges. In [69], the robots cooperatively explore the whole environment and generate its topological map. The robots independently generate local topological maps and by transferring them to each other, they are able to integrate these maps and generate a whole global map.

In summary, centralized systems obtain solutions close to the optimal but are computationally intensive and inefficient for large number of robots, these approach also have a single point of failure; on the other hand, decentralized systems are flexible and robust, but frequently achieve considerably sub-optimal solutions compared to centralized systems [60].

There are also some works on this subject which includes different concepts. For assigning the next optimal target for exploration Hungarian method [70] has been used in some recent projects [71, 72]. In an active approach, RFID tags are dropped along the way to some suitable parts of the environment [73]. These tags store the relative locations of frontier cells and visited cells and they are helpful for determining the explored and unexplored part. On the other hand it is an active approach which is not a desired situation. There are also many works in robot formation problem. Yang et al focus on a different problem in multi robot exploration. In many cases, formation of robots is very important. For instance, robots may need to form a formation such as line, triangle etc. for effective exploration. They derive a mathematical foundation for robots formation and find the optimal path for robots to obtain a line [74].

4.2. General Approach

The contribution of this chapter is to consider the problem of decentralized multi-robot exploration with topological maps - assuming a perfect communication among robots and propose a new approach based on bubble space representation [5]. The bubble space representation is used for not only mapping but motion planning which determines where to go next and localization that means where I am.

The multirobot system consists of a set of $\mathcal{R} = \{1, \ldots, r\}$ robots. Each robot is associated time-varying bubble space point $b_i = \begin{bmatrix} x_i & f_i \end{bmatrix}^T$ with base $x_i \in \mathcal{X}$ and viewing direction $f_i \in \mathcal{F}$. Initially the robots do not know anything about the environment except the relative positions of the other robots. During the course of the mission, as each robot comes to a new place, it sends its new position to other robots. As the robots all take this information into considertion while deciding where to go next, they are able to explore the environment with minimal overlap.

4.3. Where to Explore Next?

The robots use the same strategy as has been developed for the single robot case. At each place, each robot uses a local exploration algorithm followed by global exploration algorithm. The local exploration algorithm is identical to that one used in the single robot case with the following exception. The utility function used is modified in order to take other robots into account. Let $\mathcal{A}_i(t)$ denote the set of bubble surface segments at time t. Consider robot i and its utility function $\varphi_i : \mathcal{A}_t \rightarrow R^{>0}$.

With the single robot case, for each segment $A_i(t,k)$ on the bubble surface, the function φ_i is a measure of the horizontal extent ν_i and average depth μ_i of that segment. In this case, it is modified to incorporate relative distance to other robots' current and previous positions. For this define the index set of robot pairs $Q_{\mathcal{R},2} = \{ij \,|i,j \in \mathcal{R}, i < j\}$. For each pair $ij \in Q_{\mathcal{R},2}$ of robots, let $\delta_{ij}(A_i(t,k),t') = \| \; c_i(t) + \frac{u(A_i(t,k)}{\|u(A_i(t,k)\|} - c_j(t') \; \|$ denote the pairwise distance between a potential new place location and that of robot j at time t'. The distance of this potential new place location to all

the robots is defined as:

$$d_i(A_i(t,k),t) = \sum_{\substack{j\in\mathcal{R}\\ j\neq i}} \delta_{ij}(A_i(t,k),t)^2$$

Furthermore, A measure of how distant this potential new place location to all previous positions of the robots is defined as:

$$d_i(A_i(t,k),t') = \prod_{t'<t} \sum_{\substack{j\in\mathcal{R}\\ j\neq i}} \delta_{ij}(A_i(t,k),t')^2$$

The utility function is constructed via incorporating

$$\varphi_i(A_i(t,k)) = (\nu_i(A_i(t,k))\rho(b_i^*,t) + d_p(A_i(t,k)))\, d_i(A_i(t,k),t') \tag{4.1}$$

The general concept of the local algorithm for multi robot is given in Figure 4.1. The global algorithm is completely same as with the single robot exploration case.

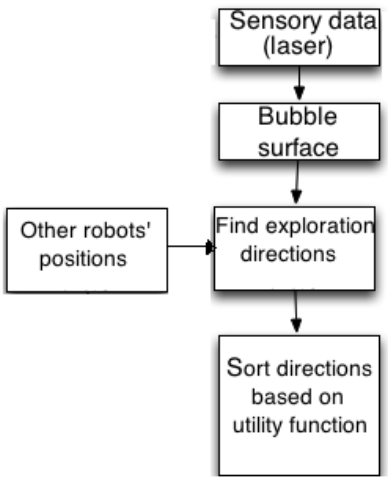

Figure 4.1. Local exploration algorithm.

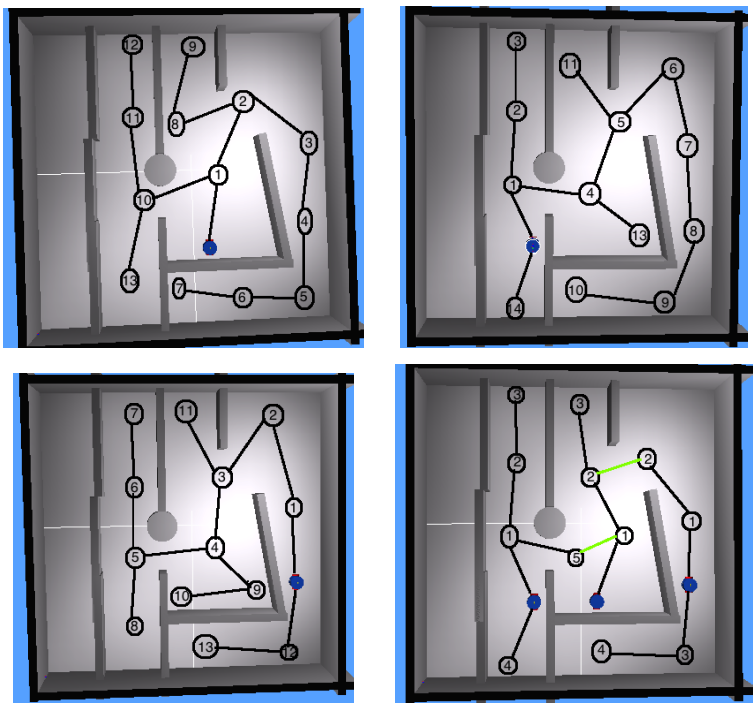

Figure 4.2. Exploration paths: Top left: Robot 1 operating only; Top Right: Robot 2 operating only; Bottom Left: Robot 3 operating only; Bottom Right: All robots operating concurrently.

4.4. Simulation Results

In this section, simulation and experimental results with multirobot exploration tasks are presented.

4.4.1. Simulated Environments in Webots

The first set of experiments are conducted in a Webots simulated outdoors environment with three robots. Each of the robots is placed at different location as shown in Figure 4.2. Gaussian encoder noise is also added to robot encoder in order to make simulations similar to real time. The ground is slippery floor so that there is 10% error pose error. Operating only, the exploration paths are as shown in the first three figures in Figure 4.2. When the three robots do the task together, the resulting exploratory path is as shown in Figure 4.2(bottom right).

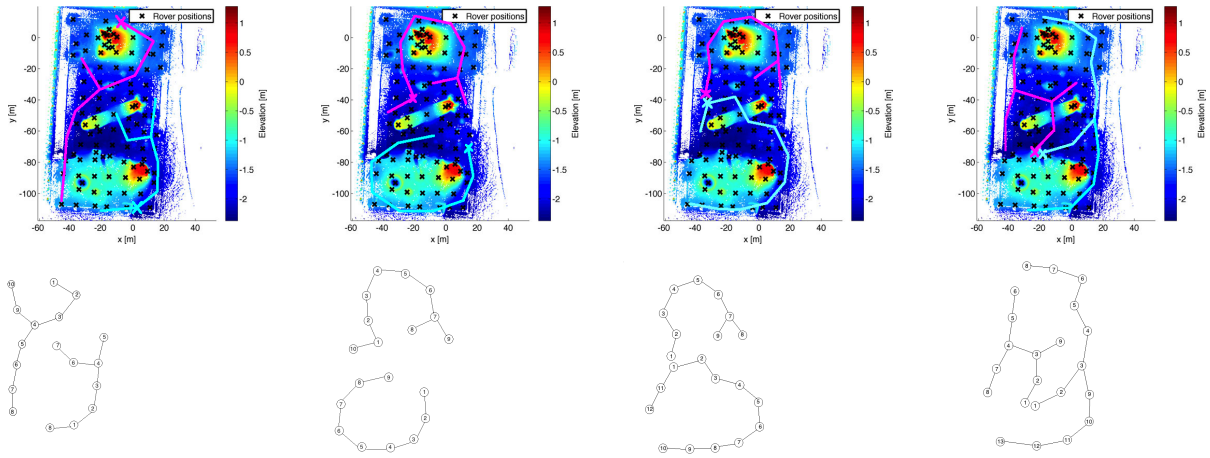

Figure 4.3. Exploration paths with varying initial positions with CPET3DM. The initial positions are indicated by the big colored crosses. Top: 2-robot paths; Bottom: Corresponding graphs.

4.4.2. Canadian Planetary Emulation Terrain 3D Mapping Dataset

Next, we apply our approach with CPET3DM [1] . In particular, we use `p2at_met` dataset that is generated in the UTIAS indoor rover test facility. This dataset consists of 3D laser scans that are obtained from 102 different locations with a grid-like structure whose dimensions are $60m \times 120m$. At each location c, the robot has 3D laser data with viewing direction $f_1 \in [0, 360]^\circ$ and $f_2 \in [-16, -5]^\circ$ with increments $\delta f_1 = 0.3^\circ$ and $\delta f_2 = 0.5^\circ$ respectively. For generating bubble surfaces, each robot chooses the dataset nearest to itself - which leads to odometric errors. The exploration mission is finished when all the robots decide to stop. For example, in Figure 4.3, the exploration is finished when the robot (whose trajectory as shown by the pink color is longer than the other robots) stops.

A 2-robot team is made to explore this terrain starting at different initial positions varying from nearby to far away. The results are as shown in Figure 4.3. It is observed that different initial positions lead to different exploration paths - as expected. It is observed the robots go in different directions - even when they start within each other's vicinity. In all, area coverage is about the same and is seemingly balanced.

The same is repeated for 3 and 4 robot teams as seen in Figure 4.4. Due to

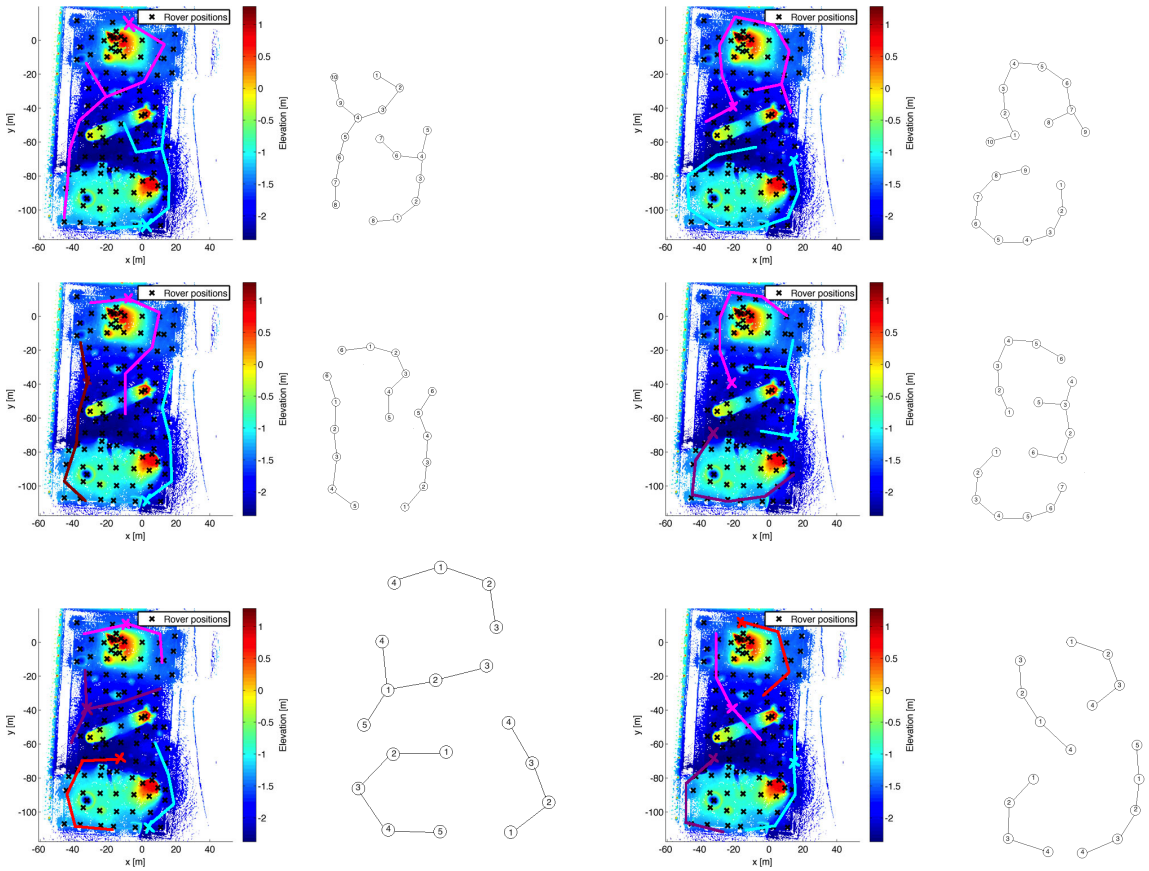

Figure 4.4. Exploration paths with varying initial positions with CPET3DM. The initial positions are indicated by the big colored crosses. Top: 2-robot paths; Center: 3-robot paths; Bottom: 4-robot paths.

variations in the initial positions, the trajectory of each robot changes accordingly. The statistical evaluation of this performace is presented in Table 4.1. It is observed that both average distance travelled as well as area explored by each robot decreases as the number of robots increase with a corresponding decrease in their respective standard deviations. As explained earlier, the area explored by each robot is computed bu using the bubble surface restricted to $f_2 = -5$. This actually leads a smaller value than the actual covered area. Furthermore, if a base is close to a ramp, then the covered area is relatively smaller than a flat terrain. However, in all area coverage is about the same.

Since all the robots move for a fixed amount of time with the same speed at each exploration step, distance values in Table 4.1 are strongly related with the time. Next, we study the effect of team size on the total exploration time as shown in Figure 4.5. It

Table 4.1. Canadian Dataset exploration results for multiple robots.

# Robots	Distance/Robot (m)		Area /Robot (Km^2)		Total area (Km^2)
	Mean	Std. Dev.	Mean	Std. Dev.	
2	14.50	3.42	2.13	0.58	4.26
3	10.33	0.82	1.48	0.11	4.44
4	6.75	1.04	1.09	0.10	4. 37

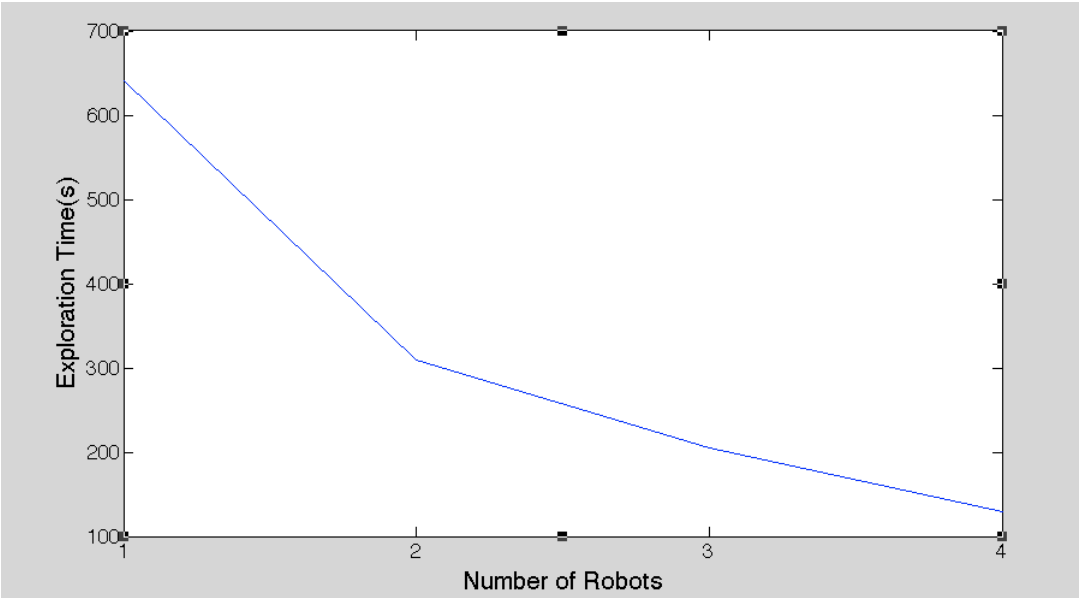

Figure 4.5. The relation between number of robots and exploration time.

is observed that as the number of robots is increased from one to four, the exploration time decreases in an exponential manner.

5. CONCLUSION

This thesis has presented novel approaches to autonomous exploration with single and multiple robots. In particular, the robots are assumed to be endowed with three-dimensional laser sensors. The exploration strategies work with topological maps. Although topological maps are less accurate than metric maps, they are computationally much more efficient - which is really important in real time applications. In particular, the nodes in the topological map are based on bubble space representation. In the bubble space representation, bubble surfaces encode different sensory features and their local S2-metric relations in a manner that is implicitly dependent on robot pose. Hence, exploration is considered with bubble space based topological maps. First, the exploration of an environment by a single robot is studied. There are two aspects to this problem: terrain mapping and determining where to go.

Terrain mapping aims to infer the environmental surface shape - as this certainly would affect the robot in determinig where to go. For this, a novel approach based on bubble space representation is proposed and experimentally evaluated. For terrain mapping, we show that each constructed bubble surface can be associated with a slope map that defines local terrain slope and its proximity for each different pan direction. Thus, instead of constructing a global terrain map a priori, local terrain map is estimated on demand in real-time as the robot navigates to a new place and constructs a bubble surface. We present experimental results with simulated and real datasets to validate this approach.

For explorative navigation, the movement direction should be such that it should point the robot to unexplored territory while being accessible. In this approach, bubble surfaces are transformed via a thresholding and are processed in order to determine candidate directions. A depth-first search algorithm is used to ensure that all the candidate directions are explored. The novelty of this approach is that the generation and recognition of nodes and their associated edges are achieved simultaneously with graph exploration. We have used simulated data as well as real experimental datasets

in order to evaluate this approach. The approach is finally applied with the Jaguar robots for autonomous explorative navigation. The results reveal that our proposed method is useful for exploring the environments.

Next, the explorative navigation strategy is extended to multirobot exploration. In this case, the robots are assumed to be communicating with each other and determine their movement directions using the bubble surface information as well as their relative position information. The developed method is independent from the number of robots and initial positions of the robots. This method reduces the exploration time as the number of robots are increased while ensuring that maximal area is covered with minimal overlap.

In order to ensure that bubble surfaces have correct base points, pose correction is considered. This is based on comparing the expected bubble surfaces with those that are actually realized. This approach is implemented in both single and multi robot exploration applications.

APPENDIX A: BUBBLE SPACE

In this Section, we briefly review bubble space. The interested reader is referred to [5] for details. Consider a robot positioned at location $c \in R^2$ with a heading $\alpha \in S^1$. Its base is defined as $x = [c\,,\ \alpha]^T$ and the base space is defined to be the set of all possible viewpoints $\mathcal{X} = R^2 \times S^1$. Let the set of (pan and tilt) viewing directions be denoted by $\mathcal{F} \subset S^2$ where $f = [f_1\, f_2]^T$. The bubble space $\mathcal{B} = \mathcal{X} \times \mathcal{F}$ is an abstract representation of the robot's base along with its viewing directions. Each point $b \in \mathcal{B}$ is defined as $b = [x\, f]^T$ where $x \in \mathcal{X}$ and $f \in \mathcal{F}$. The robot's base point is given by $\pi : \mathcal{B} \to \mathcal{X}$ - defined as the projection of b onto $\mathcal{X}$ as $\pi(b) = x$. The section is a continuous map $h : \mathcal{X} \to \mathcal{B}$ such that $\forall x \in \mathcal{X}$, $\pi(h(x)) = x$. The image of a section h – namely $Im(h(x))$ – is the set of viewing directions from a given base position x.

Assume that at time t, the robot is at base $x \in \mathcal{X}$. For each viewing direction $f \in \mathcal{F}$, it obtains 3D laser data $q(b, t)$. Now, visualize the robot to be surrounded by an hypothetical spherical surface that is deformed at each f by an amount that is dependent on the sensed data value. This surface is referred to as bubble surface $B(x, t)$. As the robot's sensor moves through a sequence of pan and tilt viewing directions, the sensed data is encoded by the bubble surface $B(x, t)$. It is an egocentric representation of its surroundings. Mathematically, the bubble surface is a deformed sphere embedded in R^3 with an intrinsic parametrization:

$$B(x,t) = \left\{ \begin{bmatrix} f \\ \rho(b,t) \end{bmatrix} \mid \forall f \in \mathcal{F} \text{ and } b = [x\, f]^T \right\} \tag{A.1}$$

where $\rho : \mathcal{B} \times R^{\geq 0} \to R^{\geq 0}$ is a Riemannian metric that encodes the 3D laser data. It is initialized to be a S^2 sphere with radius $\rho_0 \in R^{\geq 0}$ – namely $\rho(b, 0) = \rho_0$. As the robot looks around, for each fixation direction, an observation $q(b, t)$ is made, each bubble surface is also deformed in $N_\epsilon(b)$ - the ϵ-neighborhood of b - via introducing a local

bump at b

$$\rho(o, t^+) = \rho(o, t^-) + g_b(o)q(b, t)$$

where the bump function $g_b : \mathcal{B} \to [0, 1]$ is any continuous decreasing function that satisfies the following two conditions:

$$g_b(o) \quad = \quad \begin{cases} 1 & \text{if} \quad \|b - o\| = 0 \\ 0 & \text{if} \quad \|b - o\| = \epsilon \end{cases}$$

The bubble surface can be explicitly represented by the double Fourier series as:

$$\rho(b, t) = \sum_{m=0}^{H_1} \sum_{n=0}^{H_2} \varsigma_{mn} z_{x,mn}^T(t) e_{mn}(f)$$

For each (m, n), the vector $e_{mn}(f) \in R^4$ consists of an orthonormal set of trigonometric basis functions as:

$$e_{mn}(f) = \begin{bmatrix} cos(mf_1)cos(nf_2) \\ sin(mf_1)cos(nf_2) \\ cos(mf_1)sin(nf_2) \\ sin(mf_1)sin(nf_2) \end{bmatrix}$$

The corresponding vector $z_{x,mn}(t) \in R^4$ of double Fourier series coefficients associated with base point x at time t is:

$$z_{x,mn}(t) = \begin{bmatrix} \eta_{x,mn}(t) & \beta_{x,mn}(t) & \mu_{x,mn}(t) & \nu_{x,mn}(t) \end{bmatrix}^T$$

APPENDIX B: JAGUAR MOBILE ROBOT

B.1. Robot System

Jaguar robot is a robot that is designed for both indoor and outdoor applications as shown in Figure B.1. It can operate in extreme terrains and is capable of climbing up stairs (up to 200mm step). Its technical details and components are as given in

Figure B.1. Left: Jaguar robot; Right: Robot components.

Table B.1. The technical specifications of Jaguar robot are as shown in Table B.2.

B.2. Teleoperation User Guide

In this part, teleoperation user guide of Jaguar Robots is explained. This is a software interface designed together with Hakan Karaoğuz and Ramazan Arıkan. The interface is designed in Ubuntu 12.04 operating system. Although not tested, more recent versions of Ubuntu are thought to be suitable for the interface. The interface is QT and ROS based. Hence, QT and ROS need to be installed. In particular, the software is developed with ROS Fuerte, so it is highly recommended to install this version of the ROS.

Table B.1. Jaguar robot components.

No	Component	Unit	Properties
1	Length	mm	820
2	Height	mm	176
3	Width	mm	700
4	Portable Weight	kg	25
5	Battery	1	22.2V (Li-Po)
6	Motors	3	1 arm unit, 2 track-wheel unit
7	Encoders		JAGUAR-ME (1227.4 per revolution)
8	Camera 1		30fps, 640x480 resolution
9	Camera 2		30fps 640x480 resolution
10	2D Laser		Scanning Angle:240° (Resolution:0.36°)
11	GPS		OGPS501
12	IMU		IMU9000
13	Wireless		WRT802G

Table B.2. Jaguar robot - Technical Data.

Property	Unit	Theory	Experimental
Axis No	3	3	
Maximum Linear Speed	m/s	1.52	1
Maximum Rotational Speed	rad/sec	0.57	0.4
Maximum Arm Speed	rad/sec	1.52	1
Drive Method		tracked	
Maximum Slope to Climb Up	degree	higher than 45°	higher than 45°
External Interface		Wireless Connection	
Working Temperature	Celcius	Up to approx. 24°	

B.2.1. Operating the Robot

In this section we present how to activate the robot, connect to it with interface and have it operating.

(i) **Opening the Jaguar Robot:** Install the battery as shown in Figure B.2(left). Make sure that the red connector is connected with red and black with black. The robot is made operational by turning the switch on as shown in Figure B.2(right). A simple check to see if the robot is activated is to observe whether the red light of the laser scanner is on or not.

(ii) **Using the interface:**

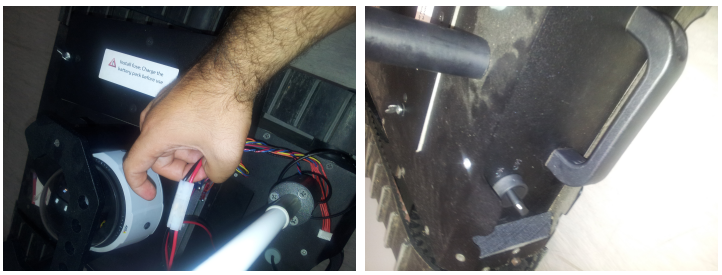

Figure B.2. Left: Battery connection; Right: Turning the power on.

- Open the computer running Ubuntu and connect to the Jaguar Robot with wireless as shown in Figure B.3.

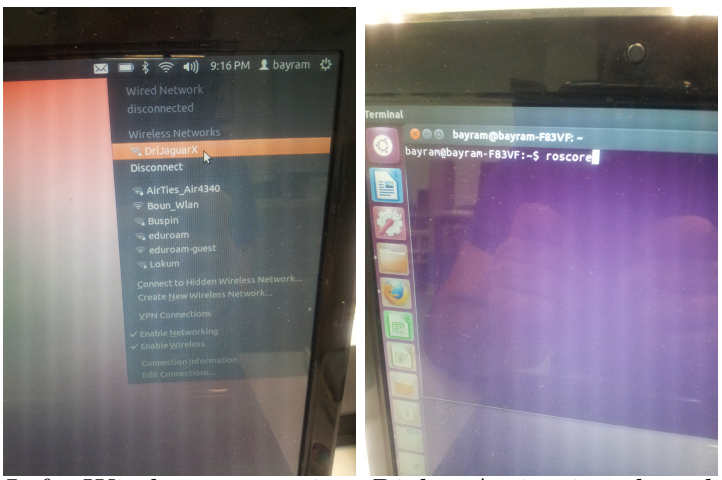

Figure B.3. Left: Wireless connection; Right: Activating the robot software.

- Open a terminal and write "'`roscore`"' as shown in Figure B.3(left-right). Roscore is the collection of nodes and programs that you will run. A series of text will appear on the screen as shown in Figure B.4.
- Jaguar robots has its own topic that enable to communicate with computer that need to be started. Note that you need to go to the directory of the terminal to run the topic as shown in Figure B.5 and Figure B.6 respectively.
- Robot status information will be listed as seen in Figure B.7.

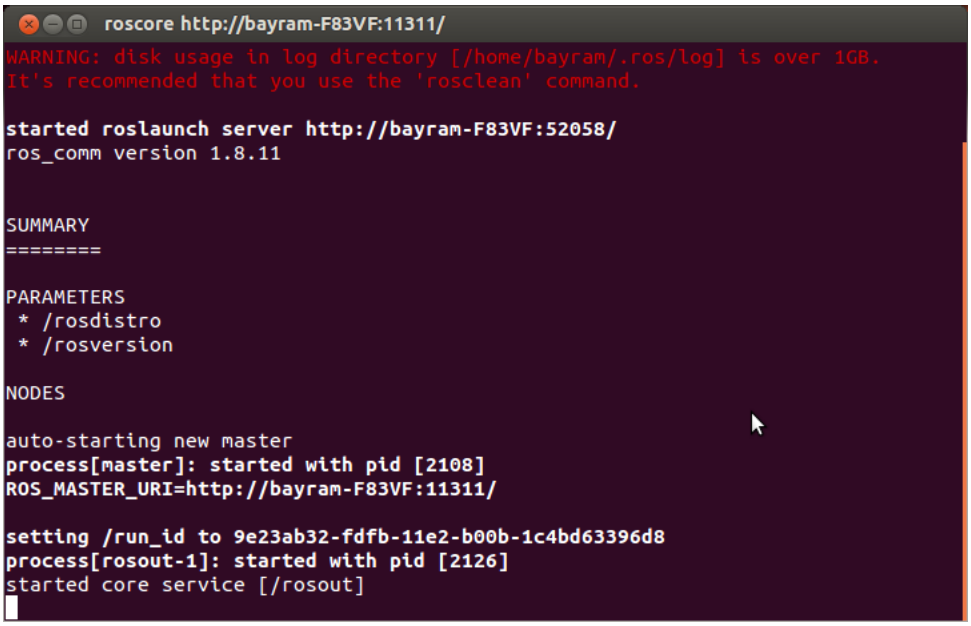

Figure B.4. Running roscore.

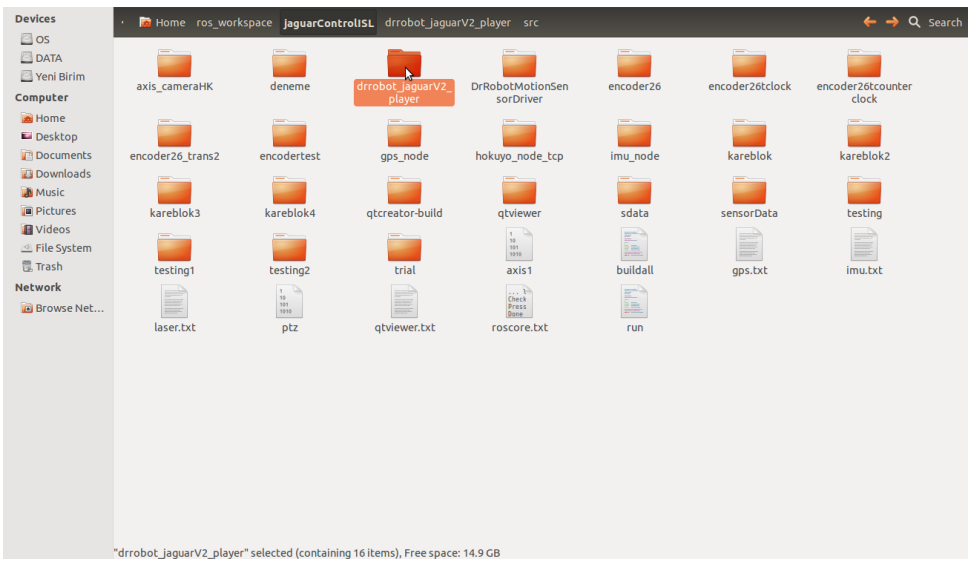

Figure B.5. The directory of Jaguar Robots communication topic.

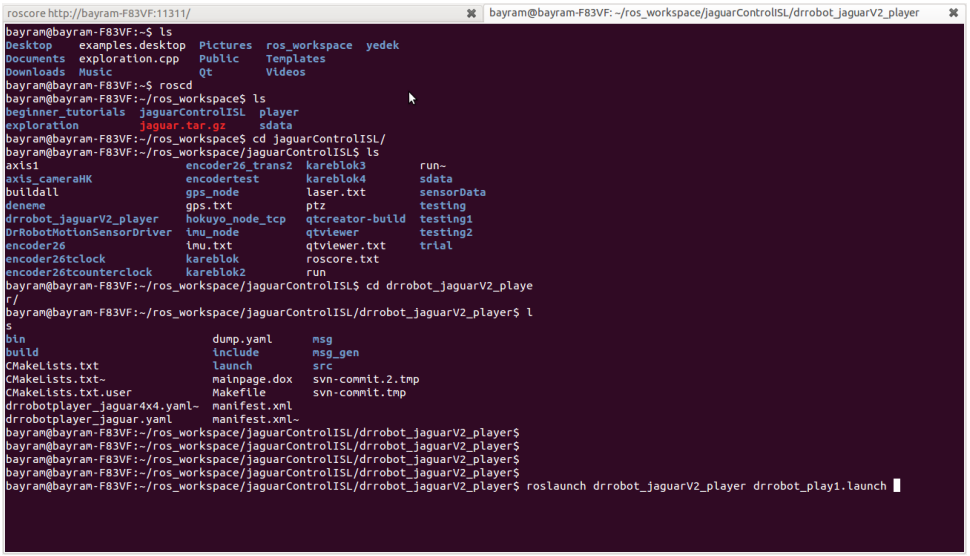

Figure B.6. Launching the communication topic.

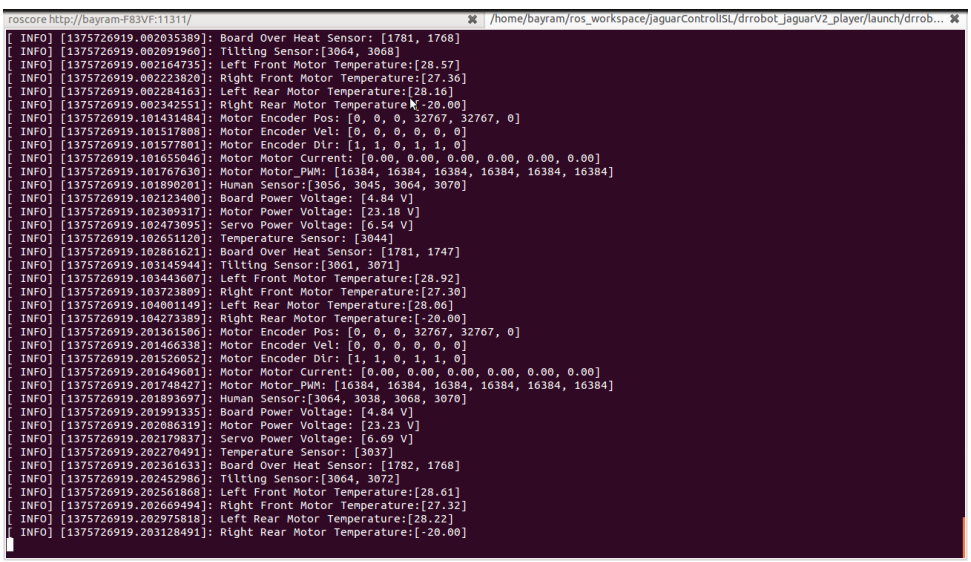

Figure B.7. The robot's data collected by communication topic.

- Now, the robot is ready to run. The interface can be invoked via going to the bin directory and running the application as shown in Figure B.8. Some sample applications are as shown in Figure B.9.

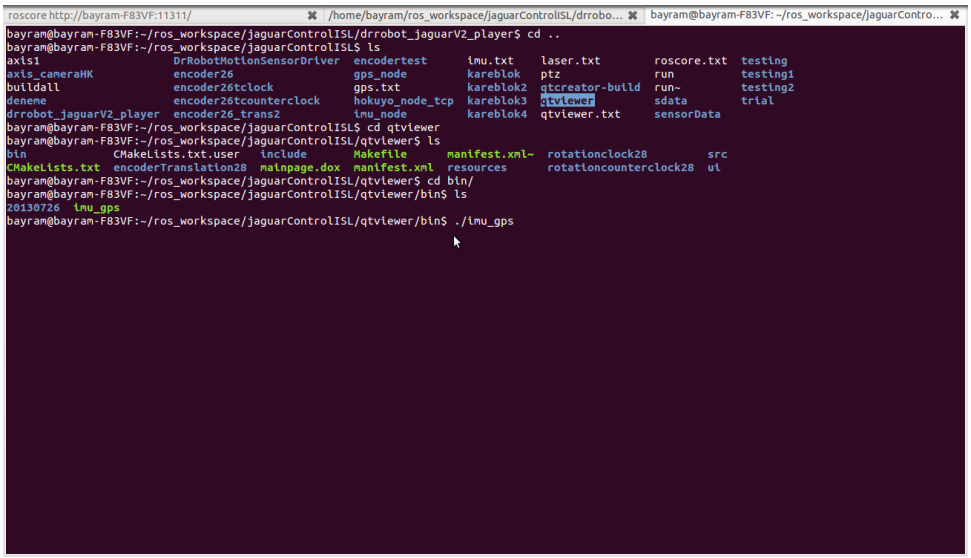

Figure B.8. Jaguar gui interface.

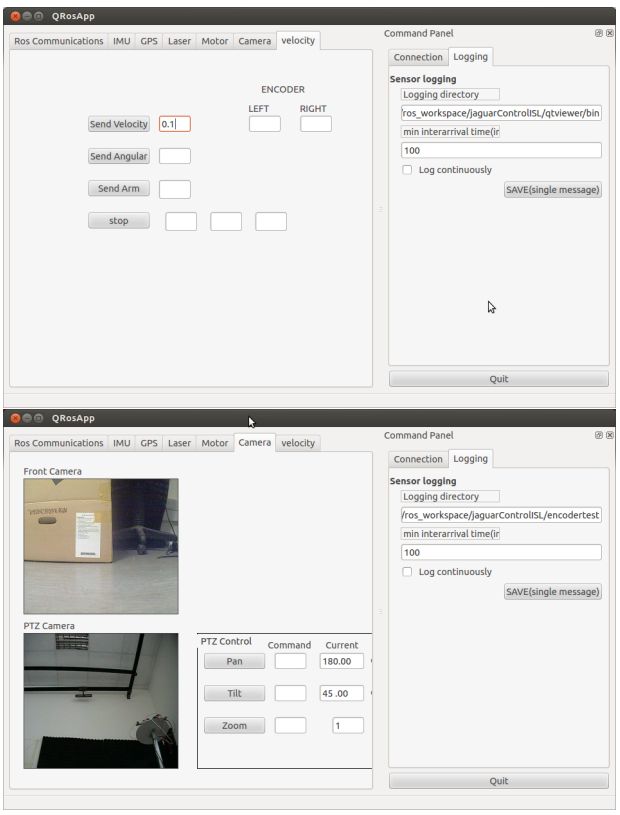

Figure B.9. Top: Teleoperated navigation; Bottom: Teleoperated data collection.

REFERENCES

1. Tong, C., D. Gingras, K. Larose, T. Barfoot and E. Dupuis, *The Canadian Planetary Emulation Terrain 3D Mapping Dataset*, International Journal of Robotics Research, 2013.

2. Brunskill, E., Y. Kobayashi, Y. Hoshino and T. Emaru, "Topological Mapping Using Spectral Clustering and Classification", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3491–3496, 2007.

3. Remolina, E. and B. Kuipers, "Towards a General Theory of Topological Maps", *Artificial Intelligence*, Vol. 152, pp. 47–104, 2004.

4. Pfaff, P., R. Triebel and W. Burgard, "An Efficient Extension to Elevation Maps for Outdoor Terrain Mapping and Loop Closing", *International Journal of Robotics Research*, Vol. 26, pp. 217–230, 2007.

5. Erkent, O. and I. Bozma, "Bubble Space and Place Representation in Topological Maps", *International Journal of Robotics Research*, pp. 22–24, 2013.

6. Hebert, M. and T. Kanade, "Analysis and Interpretation of Range", R. Jain and A. Jain (Editors), *3-D Vision Techniques for Autonomous Vehicles*, pp. 273–337, Springer, 1990.

7. Howard, A. and H. Seraji, "Vision-Based Terrain Characterization and Traversability Assessment", *Journal of Robotics Systems*, Vol. 18, No. 10, pp. 577–587, 2001.

8. Lacroix, S., A. Mallet, D. Bonnafous, G. Bauzil, S. Fleury, M. Herrb and R. Chatila, "International Journal of Robotics Research", *Autonomous Rover Navigation on Unknown Terrains: Functions and Integration*, Vol. 21, pp. 917–942, 2002.

9. Ye, C. and J. Borenstein, "T-transformation: Traversability Analysis for Navi-

gation on Rugged Terrain", *Proceedings of the Defense and Security Symposium, Unmanned Ground Vehicle Technology VI (OR54)*, pp. 12–16, 2004.

10. Howard, T. M. and A. Kelly, "Optimal Rough Terrain Trajectory Generation for Wheeled Mobile Robots", *International Journal Robotics Research*, Vol. 26, No. 2, pp. 141–166, 2007.

11. Pereira, G., "Robot Navigation in Multi-terrain Outdoor Environments", *International Journal of Robotics Research*, Vol. 28, pp. 685–700, 2009.

12. Karumanchi, S., T. Allen, T. Bailey and S. Scheding, "Non-parametric Learning to Aid Path Planning over Slopes", *The International Journal of Robotics Research*, Vol. 29, No. 8, pp. 997–1018, 2010.

13. Ye, C. and J. Borenstein, "T-transformation: Traversability Analysis for Navigation on Rugged Terrain", *IEEE Transections on Robotics*, Vol. 20, pp. 913–921, 2004.

14. Gennery, D., "Traversability Analysis and Path Planning for a Planetary Rover", *Autonomous Robots*, Vol. 6, pp. 131–146, 1999.

15. Howard, A. and H. Seraji, "An Intelligent Terrain-Based Navigation System for Planetary Rovers", *IEEE Robotics and Automation Magazine*, Vol. 8(4), pp. 9–17, 2001.

16. Zhu, J., Y. Wang, H. Yu, W. Wang and Y. Wen, "Sensing Incline Terrain for Mobile Robot Autonomous Navigation Under Unknown Environment", *IEEE International Conference on Information and Automation*, pp. 2296–2301, 2010.

17. Singh, S. and A. Kelly, "Robot Planning in the Space of Feasible Actions: Two Examples", *IEEE International Conference on Robotics*, pp. 3309–3316, 1996.

18. Lv, J., Y. Kobayashi, Y. Hoshino and T. Emaru, *Slope Detection Based on Or-*

thogonal Assumption, IEEE/SICE International Symposium on System Integration (SII), 2012.

19. Wei, B., J. Gao and K. Li, "Navigation and Slope Detection System Design for Autonomous Mobile Robot", *The Ninth International Conference on Electronic Measurement and Instruments*, pp. 46–58, 2009.

20. Wolf, D. F., G. Sukhatme, D. Fox and W.Burgard, "Autonomous Terrain Mapping and Classification Using Hidden Markov Models", *IEEE International Conference on Robotics and Automation*, 2005.

21. Elmqvist, M., E. Jungert, F. Lantz, A. Persson and U. Soderman, "Terrain Modelling and Analysis Using Laser Scanner Data", *International Archives of Photogrammetry and Remote Sensing*, pp. 22–24, 2001.

22. Smith, M., I. Posner and P. Newman, "Robot Map Verification of a Graph World", *International Journal Robotics Research.*, Vol. 5, No. 4, pp. 383–395, 2001.

23. Xu, L., L. Cao and H. Ju, "Traversability-Based Lunar Rover Autonomous Navigation in Three-dimensional Terrain", *Journal of System Simulation*, Vol. 19, No. 12, pp. 2852–2856, 2007.

24. Williams, S. and A. M. Howard, "A Single Camera Terrain Slope Estimation Technique for Natural Arctic Environments", *2008 IEEE International Conference on Robotics and Automation*, pp. 2729–2734, 2008.

25. Bares, J., M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons and W. L. Whittaker, "Ambler: An Autonomous Rover for Planetary Exploration", *IEEE Computer*, Vol. 22, pp. 18–26, 1989.

26. Kelly, A. and A. Stent, "Rough Terrain Autonomous Mobility-Part 2: An Active Vision, Predictive Control Approach", *Autonomous Robots*, Vol. 5, pp. 163–198, 1998.

27. Webots, *Commercial Mobile Robot Simulation Software*, `http://www.cyberbotics.com`, May 2013.

28. Lee, D. and M. Recce, "Quantitative Evaluation of the Exploration Strategies of a Mobile Robot", *International Journal of Robotics Research*, pp. 413–447, 1997.

29. Yamauchi, B., "A Frontier-Based Approach for Autonomous Exploration", *Symposium on Computational Intelligence in Robotics and Automation*, pp. 146–151, 1997.

30. Rao, N., S. Kareti, W. Shi and S. S. Iyengar, "Robot Navigation in Unknown Terrains: Introductory Survey of Non-Heuristic Algorithms", *Oak Ridge National Laboratories*, Vol. 12410, 1993.

31. Deng, X., T. Kamade and C. Papadimitriou, "How to Learn in an Unknown Environment", *32nd Symposium Foundations Computational Science*, pp. 298–303, 1991.

32. Deng, X., E. Milios and A. Mirzaian, "Robot Map Verification of a Graph World", *Journal of Combinatorial Optimization*, Vol. 5, No. 4, pp. 383–395, 2001.

33. Kuipers, B. J. and Y. T. Byun., "A Robot Exploration and Mapping Strategy Based on a Semantic Hierarchy of Spatial Representations", *Journal of Robotics and Autonomous Systems 8*, pp. 47–63, 1991.

34. Shade, R. and P. Newman, "Choosing Where to Go: Complete 3D Exploration with Stereo", *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2806–2811, 2011.

35. Thrun, S., "Exploration in Active Learning", M. A. Arbib (Editor), *The Handbook of Brain Theory and Neural Networks*, pp. 381–384, MIT Press, 1995.

36. Moorehead, S., R. Simmons and W. Whittaker, "Autonomous Exploration Us-

ing Multiple Sources of Information", *International Conference on Robotics and Automation*, pp. 3098–3103, 2001.

37. Koenig, S. and C. Tovey, "Improved Analysis of Greedy Mapping", *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3251–3257, Las Vegas, 2003.

38. Oriolo, G., G. Ulivi and M. Vendittelli, "Real-Time Map Building and Navigation for Autonomous Robots in Unknown Environments", *IEEE Transections on Systems, Management and Cybernetics - Part B: Cybernetics*, Vol. 28, No. 3, pp. 316–333, 1998.

39. Bourgoult, F., A. A. Makarenko, S. B. Williams, B. Grocholsky and F. Durrant-Whyte, "Information-Based Adaptive Robotics Exploration", *In Proceedings of the International Conference on Intelligent Robots and Systems*, pp. 540–545, 2002.

40. Simmons, R. G., D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun and H. L. S. Younes, "Coordination for Multi-Robot Exploration and Mapping", *In Proceedings of the 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence*, pp. 852–858, 2000.

41. Freda, L. and G. Oriolo, "Frontier-Based Probabilistic Strategies for Sensor-Based Exploration", *IEEE International Conference on Robotics and Automation*, pp.38-81, 2005.

42. Stachniss, C., G. Grisetti and W. Burgard, "Information Gain-Based Exploration Using Rao-Blackwellized Particle Filters", *Robotics: Science and Systems Conference*, pp. 65–72, 2005.

43. Gonzalez-Banos, H. H. and J.-C. Latombe, "Navigation Strategies for Exploring Indoor Environments", *International Journal of Robotics Research*, Vol. 21, No. 10, pp. 829–848, 2002.

44. Suppa, M., P. Wang, K. Gupta and G. Hirzinger, "C-space Exploration Using Noisy Sensor Models", *International Conference on Robotics and Automation*, pp. 4777–4782, 2004.

45. Freda, L. and G. V. Oriolo, "Sensor-Based Exploration for General Robotics Systems", *International Conference on Intelligent Robots and Systems*, pp. 2157–2164, 2008.

46. LaValle, S. M. and J. J. Kuffner, "Rapidly Exploring Random Trees: Progress and Prospects", B. R. Donald, K. M. Lynch and D. Rus (Editors), *Algorithmic and Computational Robotics: New Directions*, pp. 293–308, A K Peters, 2001.

47. Thrun, S. and A. Biicken, "Integrating Grid-Based and Topological Maps for Mobile Robot Navigation", *Thirteenth National Conference on Artificial Intelligence*, pp. 944–950, 1996.

48. Jia, S., H. Shen, X. Li, W. Cui and K. Wang, "Autonomous Robot Exploration Based on Hybrid Environment Model", *Proceeding of the IEEE International Conference on Information and Automation*, 2012.

49. Dudek, G., M. Jenkin, E. Milios and D. Wilkes, "Robotics Exploration as Graph Construction", *IEEE Transection of Robotics and Automotion*, Vol. 7, No. 6, pp. 859–865, 1991.

50. Rekleitis, I., G. Dudek and E. Milios, "Multi-Robot Exploration of an Unknown Environment, Efficiently Reducing the Odometry Error", *In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1340–1345, Nagoya, 1997.

51. Rekleitis, I. M., V. Dujmovic and G. Dudek, "Efficient Topological Exploration", *International Conference on Robotics and Automation*, pp. 676–681, 1999.

52. Bender, M. A., A. Fernandez, D. Ron, A. Sahai and S. Vadhan, "The Power of a

Pebble: Exploring and Mapping Directed Graphs", *30th Annual ACM Symposium*, pp. 269–278, 1998.

53. Deng, X. and C. H. Papadimitriou, "Exploring an Unknown Graph", *Proceedings of the 31st Annual IEEE Symposium on FOCS*, pp. 355–361, 1990.

54. Kwek, S., "On a Simple Depth-First Search Strategy for Exploring Unknown Graphs", *International Workshop on Algorithms and Data Structures*, pp. 345–353, 1997.

55. Albers, S. and M. Henzinger, "Exploring Unknown Environments", *SIAM Journal on Computing*, Vol. 29, pp. 1164–1188, 2000.

56. Fleischer, R. and G. Trippen, "Exploring an Unknown Graph Efficiently", *13th Annual European Symposium Algorithms*, pp. 11–22, Las Vegas, 2005.

57. Akdeniz, B. and I. Bozma, "Local Terrain Mapping via 3D Laser Based Bubble Surfaces", *In Proceedings of ECMR 6th European Conference on Mobile Robots*, 2013.

58. Dudek, G., M. Jenkin, E. Milios and D. Wilkes, "A Taxonomy for Multi-agent Robotics", *Autonomous Robots*, pp. 375–397, 1996.

59. Cao, Y. U., A. S. Fukunaga and A. Kahng, "Cooperative Mobile Robotics: Antecedents and Directions", *Autonomous Robots*, Vol. 4, No. 1, pp. 7–27, 1997.

60. Burgard, W., M. Moors and F. Schneider, "Collaborative Exploration of Unknown Environments with Teams of Mobile Robots", *Plan-Based Control of Robotics Agents*, pp. 52–70, 2002.

61. Thrun, S. and W. Burgard, "A Real-time Algorithm for Mobile Robot Mapping with Applications to Multi-Robot and 3D Mapping", *In Proceedings of the IEEE International Conference on Robotics and Automation*, 2000.

62. Yamauchi, B., "Frontier-Based Exploration Using Multiple Robots", *In Proceedings of the International Conference on Autonomous Agents (AGENTS)*, pp. 47–53, New York, 1998.

63. Fox, D., "Distributed Multirobot Exploration and Mapping", *Proceedings of IEEE*, Vol. 94, 2006.

64. Renzaglia, A. and A. Martinelli, "Distributed Multirobot Exploration and Mapping", *INRIA Rhone-Alpes*, 2010.

65. Konolige, K., D. Fox, C. Ortiz, A. Agno, M. Eriksen, B. Limketkai, J. Ko, J. M., D. Schulz, B. Stewart and R. Vincent, "Centibots: Very Large Scale Distributed Robotics Teams", *In Proceedings of the International Symposium on Experimental Robotics*, ISER, 2004.

66. Rekleitis, I., G. Dudek and E. Milios., "Multi-Robot Exploration of an Unknown Environment", *In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1340–1345, Nagoya, 1997.

67. Franchi, A., L. Freda, G. Oriolo and M. Vendittelli, "A Randomized Strategy for Cooperative Robot Exploration", *International Conference on Robotics and Automation*, pp. 768–774, 2007.

68. Franchi, A., L. Freda, G. Oriolo and M. Vendittelli., "The Sensor-Based Random Graph Method for Cooperative Robot Exploration", *IEEE/ASME Transactions on Mechatronics*, Vol. 14, No. 2, pp. 163–175, 2009.

69. Marjovi, A. and L. Marques, "Multi-Robot Topological Exploration Using Olfactory Cues", *Distributed Autonomous Robotics Systems Springer Tracts in Advanced Robotics*, Vol. 83, No. 8, pp. 47–60, 2010.

70. Kuhn, H. W., "The Hungarian Method for the Assignment Problem", *Naval Research Logistics Quarterly*, Vol. 2, No. 1, pp. 83–97, 1955.

71. Wurm, K. M., C. Stachniss and W. Burgard., "Coordinated Multi-Robot Exploration Using a Segmentation of the Environment", *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1160–1165, 2008.

72. Yan, Z., N. Joundaeu and A. A. Cherif., *Sampling-Based Multi-Robot Exploration*, International Symposium on Robotics, 2010.

73. Kleiner, A., J. Prediger and B. Nebel., "RFID Technology-Based Exploration and SLAM for Search and Rescue", *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4054–4059, 2006.

74. Jia, Z., H. Ma and M. W. C. Yang, "Three-Robot Minimax Travel-Distance Optimal Formation", *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pp. 12–15, Orlando, 2011.