Sensor network based collision-free navigation and map building for mobile robots

Author:
Li, Hang

Publication Date:
2017

DOI:
https://doi.org/10.26190/unsworks/20038

License:
https://creativecommons.org/licenses/by-nc-nd/3.0/au/
Link to license to see what you are allowed to do with this resource.

Sensor Network Based Collision-Free Navigation and Map Building for Mobile Robots

A thesis written by

Hang Li

under the supervision of

Prof. Andrey V. Savkin

in fulfilment of the requirements for the degree of

Doctor of Philosophy

School of Electrical Engineering and Telecommunications

Faculty of Engineering

The University of New South Wales

November 2017

PLEASE TYPE

THE UNIVERSITY OF NEW SOUTH WALES
Thesis/Dissertation Sheet

Surname or Family name: LI

First name: Hang

Other name/s:

Abbreviation for degree as given in the University calendar: Ph.D.

School: School of Electrical Engineering and Telecommunications

Faculty: Faculty of Engineering

Title: Sensor Network Based Collision-Free Navigation and Map Building for Mobile Robots

Abstract 350 words maximum: (PLEASE TYPE)

Safe robot navigation is a fundamental research field for autonomous robots including ground mobile robots and flying robots. The primary objective of a safe robot navigation algorithm is to guide an autonomous robot from its initial position to a target or along a desired path with obstacle avoidance. With the development of information technology and sensor technology, the implementations combining robotics with sensor network are focused on in the recent researches. One of the relevant implementations is the sensor network based robot navigation. Moreover, another important navigation problem of robotics is safe area search and map building.

In this thesis, a global collision-free path planning algorithm for ground mobile robots in dynamic environments is presented firstly. Considering the advantages of sensor network, the presented path planning algorithm is developed to a sensor network based navigation algorithm for ground mobile robots. The 2D range finder sensor network is used in the presented method to detect static and dynamic obstacles. The sensor network can guide each ground mobile robot in the detected safe area to the target. The computer simulations and experiments confirm the performance of the presented method. Furthermore, considering the implementations of small-sized flying robots in industry, the presented navigation algorithm is extended into 3D environments. In the presented method, a time-of-flight camera network is used to detect the static and moving obstacles. With the measurements of the sensor network, any flying robot in the workspace is navigated by the presented algorithm from the initial position to the target and avoids any obstacles in the workspace.

Moreover, in this thesis, another navigation problem, safe area search and map building for ground mobile robot, is studied and two algorithms are presented. In the first presented method, we consider a ground mobile robot equipped with a 2D range finder sensor searching a bounded 2D area without any collision and building a complete 2D map of the area. Furthermore, the first presented map building algorithm is extended to another algorithm for 3D map building.

FOR OFFICE USE ONLY   Date of completion of requirements for Award:

ORIGINALITY STATEMENT

'I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, or substantial proportions of material which have been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by others, with whom I have worked at UNSW or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.'

Signed   …………………………………………...............

Date   …………………………………………...............

Abstract

Safe robot navigation is a fundamental research field for autonomous robots including ground mobile robots and flying robots. The primary objective of a safe robot navigation algorithm is to guide an autonomous robot from its initial position to a target or along a desired path with obstacle avoidance. With the development of information technology and sensor technology, the implementations combining robotics with sensor network are focused on in the recent researches. One of the relevant implementations is the sensor network based robot navigation. Moreover, another important navigation problem of robotics is safe area search and map building.

In this thesis, a global collision-free path planning algorithm for ground mobile robots in dynamic environments is presented firstly. Considering the advantages of sensor network, the presented path planning algorithm is developed to a sensor network based navigation algorithm for ground mobile robots. The 2D range finder sensor network is used in the presented method to detect static and dynamic obstacles. The sensor network can guide each ground mobile robot in the detected safe area to the target. The computer simulations and experiments confirm the performance of the presented method. Furthermore, considering the implementations of small-sized flying robots in industry, the presented navigation algorithm is extended into 3D environments. In the presented method, a time-of-flight camera network is used to detect the static and moving obstacles. With the measurements of the sensor network, any flying robot in the workspace is navigated by the presented algorithm from the initial position to the target and avoids any obstacles in the workspace.

Moreover, in this thesis, another navigation problem, safe area search and map building for ground mobile robot, is studied and two algorithms are presented. In the first presented method, we consider a ground mobile robot equipped with a 2D range finder sensor searching a bounded 2D area without any collision and building a complete 2D map of the area. Furthermore, the first presented map building algorithm is extended to another algorithm for 3D map building.

Acknowledgements

List of Publications

The works presented in this thesis were published, accepted or submitted in the following papers:

Journal papers

1. A. V. Savkin and H. Li. A safe area search and map building algorithm for a wheeled mobile robot in complex unknown cluttered environments. *Robotica*, 2017. Advance online publication.

2. A. V. Savkin and H. Li. Simultaneous collision free area exploration and 3D map building of unknown indoor environments by a ground mobile robot. *Robotics and Autonomous Systems*. Submitted.

3. H. Li and A. V. Savkin. An algorithm for safe navigation of mobile robots by a sensor network in dynamic cluttered industrial environments. *Robotics and Computer-Integrated Manufacturing*. Submitted.

4. H. Li and A. V. Savkin. Wireless sensor network based navigation of micro flying robots in the industrial internet of things. *IEEE Transactions on Industrial Informatics*. Submitted.

Conference papers

1. H. Li. Global shortest path planning for a wheeled mobile robot navigated by a sensor network in dynamic cluttered environments. In *2016 35th Chinese*

Control Conference (CCC), pages 5525–5530, Chengdu, China, July 2016.

2. A. V. Savkin and H. Li. Collision free navigation of a non-holonomic ground robot for search and building maps of unknown areas with obstacles. In *2016 35th Chinese Control Conference (CCC)*, pages 5409–5414, Chengdu, China, July 2016.

3. H. Li and A. V. Savkin. A navigation algorithm for a non-holonomic mobile robot navigated by a sensor network in dynamic cluttered environments. In *2017 36th Chinese Control Conference (CCC)*, pages 5801–5804, Dalian, China, July 2017.

4. H. Li and A. V. Savkin. A method for collision free sensor network based navigation of flying robots among moving and steady obstacles. In *2017 36th Chinese Control Conference (CCC)*, pages 6079–6083, Dalian, China, July 2017.

5. A. V. Savkin and H. Li. A collision-free area search and 3D map building algorithm for a ground mobile robot in unknown indoor environments. In *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. Accepted.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Collision-free navigation is a fundamental problem of robotics for mobile robots. The collision-free navigation of a mobile robot is defined as the process of guiding a mobile robot to a target or along a desired path with obstacle avoidance. With the development of mathematics, electronics and communications, both the robot's computer performance and the control strategies are being improved dramatically during the previous years.

With the development of mechanical engineering, varieties of mobile robots are designed for different tasks in different environments. According to the environments where the robot works, the mobile robots are generally classified into three categories; i.e. ground mobile robots, underwater robots and flying robots. In this thesis, we focus on the navigation of ground mobile robots and flying robots.

The models we used in this thesis for both ground mobile robot and flying robot are non-holonomic models. We consider a ground mobile robot as a differential wheeled robot, which is a Dubin's car with a non-holonomic constraint [194, 87, 136, 1]. Comparing with an omnidirectional mobile robot, like [147, 28, 66], the non-holonomic ground mobile robot model is more practical. Moreover, we consider a flying mobile robot as an under-actuated non-holonomic mobile robot [101, 188].

In this chapter, the previous research works in robot navigation and relevant topics are studied and reviewed as follows.

1.1   Robot navigation problems

Robot navigation is an important technology in engineering, military and commercial applications; see [79, 153, 163, 162, 181, 107, 110, 8, 157, 188, 56]. Generally, the solutions of the robot navigation problem are classified into two classes, local and global navigation algorithms. A local navigation algorithm is to use the local information measured by the robot's sensors (e.g. range finder sensor, sonar sensor and visual sensor) to guide the robot to a target; see [133, 180, 72]. However, with the development of computer science, microelectronics and sensor network technology, global navigation algorithms have become another significant type of solution in current research works and implementations.

In the local navigation algorithms, the mobile robot obtains the environment information by using the mounted sensors and determines the motion planning to navigate the robot to a target or along a desired path; see [182, 184, 9, 173, 91]. In the previous works in local navigation, there are different types of sensors used to obtain the environment information. In the works [51, 77, 24], laser range finder is considered because it can provide accurate map of the local area. In other works [115, 21, 146], camera is used to capture the environment image and generate the map for the local area. It involves image process, which is more complex than laser range finder. Sonar sensor is another common sensor used for small-sized and cheap robot; see [71]. In the past few years, there are a lot of local navigation algorithms proposed. In the work [168], the authors propose a hybrid approach to finish the tasks by combining the environment information with local perceptions. In the work [179], a path planning algorithm is proposed based on the potential field, which is widely used in many other works, such as [142, 143, 167]. In the work [109], a sliding mode controller is proposed by using only distance from the robot to a moving target to drive the robot to a predefined distance to the target. In the work [99], the boundary following problem for a unicycle-like robot is addressed and a sliding mode control law is proposed to drive a unicycle-like mobile robot at a predefined distance from an obstacle's boundary while moving along the boundary.

In another work [98], a reactive navigation algorithm is proposed to guide a mobile robot to a target with obstacle avoidance. In this research work, the obstacles can be moving and deforming. The proposed control law is proved to be globally converging according to the mathematical analysis.

Another significant problem in robotics is robot formation building with multi-agent system [122, 121, 17, 196, 149]. The aim of the robot formation is to drive multiple mobile robots to achieve prescribed constraints on their states [120]. Generally the problem can be classified into shape producing problem and shape tracking problem. There are a lot of research works focusing on both of the two problems. In the work [112], the market-based coordination protocols are used to solve the dynamical task assignment problem in the distributed formation algorithm for shape producing, while another work [64] which provides an off-line task assignment strategy. In the work [197], a model is proposed to describe the dynamic characteristic of the multi-robot system and a model-based control strategy is given to solve the formation and trajectory tracking problem. In the work [97], firstly, a collision-free path tracking controller is designed for a single non-holonomic ground mobile robot. Then the proposed controller is used to solve the formation problem for multi-robot system. In another research work [165], the non-holonomic mobile robot is considered as the work [97]. In this work, a method for decentralized flocking and global formation building for multi non-holonomic mobile robot network is proposed. The robot model considered in this work has hard constraints on the robot's linear and angular velocities. The robots can be guide to move in a desired geometric pattern. In another work [198], the authors propose a formation framework to control a multi-agent robot system with a large number of robots in finite time by separating the formation information into local and global parts. This method can perform the navigation and formation control with less data exchange. In another work [37], the time-varying formation control problem is addressed for flying robots. The control method is proposed based on the Lyapunov approach combining with Riccati technique. A similar problem with [37] is mentioned and solved in another work [36] for

linear time-invariant multi-agent systems. In this work, time delay is considered. In the work [207], a distributed control algorithm for multiple non-holonomic robots is proposed. In the research work, robots only can see a part of the environment by camera and measure the range to other robots. The navigation algorithm drive the robots to encircle a given dynamic target with uniform distribution over the respective circle and collision avoidance.

In this thesis, we focus on two specific navigation problems. The first problem is sensor network based navigation problem described in Section 1.2. The second problem studied in this thesis is safe area search and map building problem which is explained in Section 1.3.

1.2   Sensor network based navigation algorithms

As mentioned above, the sensor network based navigation problem becomes a new challenge in recent years with the development of the sensor network. In the past decade, the sensor network technology was developed significantly. There are a variety of sensor networks proposed for different environments and implementations; see [52, 178, 43, 68, 199]. The main advantage of the sensor network is that it can capture the environment information in a wide area rapidly rather than single sensor in a limited coverage. This advantage promote the cooperation of sensor network and mobile robots in a lot of fields, such as development using mobile robots, data collection, mobile robot localization and navigation; see [138, 65, 47].

In the past five years, there are a lot of works proposed in the sensor network based navigation. Comparing with the local navigation algorithms, the sensor network based navigation algorithms use much more information captured by the sensor network to perform a more efficient navigation. The previously proposed research works in the sensor network based navigation can be generally classified into collision-free navigation methods and target-reaching navigation methods. The objective of a collision-free navigation algorithm is to use the measurements captured by the sensor network to guide a mobile robot to a target with obstacle avoidance.

It takes the main advantage of the sensor network to detect the static and dynamic obstacles in the environments; see an example in [193].

In the previous works of sensor network based collision-free navigation, camera sensor network is one of the common types of sensor networks used for obstacle detection; see [183, 23, 211]. In the work [183], a harmonic navigation algorithm for multiple ground wheelchair robots is presented by using wireless sensor nodes, which are deployed in dynamic indoor environments. The sensor network used in [183] consists of some cameras. The location of the wheelchairs and the occupied area are estimated by the image process. It can guide the wheelchairs avoiding the static obstacles and moving obstacles. In another work [23], the same camera sensor network is used to localize the robots and recognize the obstacles. The proposed algorithm in [23] allows the mobile robot or a vehicle which has less intelligence to perform a sophisticated mobility without a large number of on-board computations. The camera deployed in the indoor environment is also used in another work [211] to guide a mobile robot. However, in the proposed method in [211], only the single camera is used, not a sensor network. In addition to the camera sensor network, there are other types of sensor network used in the previous works. In the work [208], an ultrasonic sensor network is deployed on the ceiling of an indoor environment according to the square grid. Each ultrasonic sensor node measures the distance from the ceiling to the static obstacle below the sensor. According to the presented ultrasonic sensor network, a 3D map of the environment can be built approximately. Then, a D*Lite [75] path planning algorithm is used to plan the robot path. One of the disadvantages of the method in [208] is that the method requires a large number of sensor nodes densely deployed in an indoor environment. In the other two works [210, 39], the RSSI (received signal strength indicator) based sensor network is used to perform a safe navigation for a ground mobile robot. In the work [210], the authors propose an RSSI-based localization and navigation method in a static environment. Similar as [208], the sensor nodes are deployed according to the grid. Then, the A* path searching algorithm is used to search a collision-free robot

path. However, the authors do not indicate which approach the sensor network used to detect obstacles. In another RSSI-based method [39], RSSI-based sensor network combined with RFID (radio frequency identification) is used to guide a ground mobile robot in an static indoor environment and avoid obstacles. The RSSI-based sensor network indicates the reference heading for robot and the RFID tags around obstacles are used to perform the robot obstacle avoidance. The main difficulty of applying the RSSI-based navigation methods in practical implementations is that the sensor network involves a large number of sensors to cover the whole environment. The sensor nodes should be deployed manually, which is difficult in a larger area and is not economical. In addition to the sensor networks above, the infra-red (IR) sensors and 2D range finder sensors are used in the sensor network to detect obstacles; see [20] and [203], respectively. In the work [20], a sensor network measures the environment data and uses the IR signals to couple the neighbour nodes and recognize the obstacles. Then, the data collection points are determined and an optimal collision-free data collection path is generated for the ground mobile robot. In another work [203], a 2D range finder sensor network is considered in the environments. According to the measurements captured by different sensor nodes, a partly detected map can be obtained and the PRM (probabilistic roadmap) algorithm is used to generate a safe robot path. The main advantage of the work [203] is that the 2D range finder sensor network can obtain an accurate map indicating the detected obstacles and unoccupied area. It involves much less number of sensor nodes than RSSI-based sensor network to cover a same area. However, the authors of [203] do not consider the motion control of any robot model in the work.

Target-reaching navigation is another topic in the sensor network based robot navigation. The main difference of the target-reaching navigation with collision-free navigation is that the collision and obstacles are generally not considered. The objective of target-reaching navigation is to guide the mobile robot from an initial position to a target sensor node or an unknown position estimated by the sensor network. In this field, the RSSI-based sensor network is widely used for the robot

navigation. One of the disadvantages of an RSSI-based sensor network is that it cannot detect the obstacles and provide a collision-free navigation. Although there are some RSSI-based collision-free navigation algorithms discussed above, the obstacle detection is still not solved by the RSSI-based sensor network. In the collision-free navigation algorithm proposed in the work [39], the RSSI-based sensor network does not provide the obstacle avoidance. Therefore, the RFID tags are involved to indicate the possible collision with static obstacles. In another work [210], the approach of the obstacle detection in the RSSI-based sensor network is not discussed. In other works of the RSSI-based target-reaching navigation, some of the works are localization-free navigation, like [176, 32, 31], and some works requires the odometry of the mobile robot, like [209, 30]. In the work [32], the target is localized by the RSSI-based sensor network and a pseudogradient is proposed to navigate the mobile robot to the target. The robot's location is also estimated by the sensor network. In another work [31], a path generation strategy is proposed based on the pseudogradient proposed in [32] to navigate the robot with a shorter trajectory. In the work [176], a localization-free and range-free navigation algorithm for mobile robot is proposed to guide the robot along the node-to-node path.

1.3 Safe area search and map building algorithms

Area search and map building is another important topic in robot navigation; [131, 11, 137, 29, 41, 49, 129]. The objective of an are search and map building algorithm is to guide the mobile robot along a desired collision-free path while searching and mapping the environment by the robot's local sensors.

One of the primary requirements in this problem is collision-free robot navigation for area exploration on the bounded flat ground. In the past few years, there are a lot of local navigation algorithms which are presented. Generally there are two mathematical models used to describe the mobile robot, onmidirectional mobile robot and non-holonomic mobile robot. The work [205] considers the obstacle avoidance with an onmidirectional mobile robot and the work [107] considers the navigation

of a non-holonomic mobile robot. In the work [200], an extended state observer and a nonlinear controller are designed and analysed for obstacle avoidance of a non-holonomic ground mobile robot. In another work [67], a switch-system approach to obstacle avoidance is proposed. In the works [200, 67], the authors consider the control of both the linear velocity and angular velocity of a non-holonomic ground mobile robot with complex computation. In the work [45], the authors consider the particle swarm optimization to search the robot's collision-free path and in the work [201], Q-learning is considered for robot obstacle avoidance navigation. The main disadvantage of the particle swarm optimization and Q-learning is that the performance of the algorithms cannot be proved mathematically. In another work [125], the authors design an mobile robot obstacle avoidance algorithm by using fuzzy potential field and sonar sensors.

Map building is another fundamental requirement of the topic. There are different types of on-board sensors used to capture the environment information. The 2D range finder sensor is a common type used to scan the obstacles in the environments. It performs a more accurate map than a monocular camera or other vision sensors, such as [33]. Although there are a number of researchers using time-of-flight (ToF) camera to capture depth image of the environment for map building like [89, 166, 78], a 2D range finder sensor is much more economical than a ToF camera and has a larger scanning angle. It can be noticed that in the work [16], the authors used the same type of sensor, 2D range finder sensor, to scan a 3D environment with an accurate map, unlike the rough 2D map in [170]. However, in the work [16], the 2D range finder sensor is connected with a motor, which can be controlled to adjust the pitch of the sensor. It makes the map building control complicate.

Moreover, varieties of area search and map building problems for mobile robots have attracted a lot of attention in the robotics community; see e.g. [110, 70, 118, 139, 57, 94, 190, 6, 185, 195, 119]. Recent publications in this field present many achievements in both single robot mapping and multi-robot mapping. The frontier-based exploration is the most common method in single robot exploration

and mapping; see e.g. [4, 124, 42, 190]. For multi-robot systems, not only frontier-based algorithms (see e.g. [2, 25]) but other approaches (see e.g. [53, 93]) are proposed. For single robot exploration, there are a lot of achievements. The work [4] provides an efficient and simple algorithm to explore an closed environment by using incremental triangulation. However, this algorithm requires the obstacles in the environments are polygons. Another work [124] presented a fast exploration algorithm based on a simple prior topological map of the environment. One of the disadvantages is that the algorithm of [124] cannot be implemented in a completely unknown environment. The works [42, 190] provides two similar frontier-based exploration algorithms with search trees. The main difference of these two algorithms is the algorithm of [42] selects a candidate randomly and the algorithm of [190] determines the candidate by a multi criteria decision method.

1.4 Contributions

In this thesis, we present three algorithms of sensor network based navigation in networked control systems; see examples [103, 104, 150, 105]. Moreover, two algorithms of robot area search and map building are presented based on switched control system, such as [102, 152, 159, 175].

In Chapter 2, 3 and 4, we study the navigation problem with micro-sized robot. The robot is unable to take an environment detection sensor for local navigation; therefore, a sensor network is proposed to help navigate the robot. Firstly, we solve the global path planning problem by an artificial potential field based algorithm in Chapter 2. Then, the algorithm is developed into a real-time navigation method with practical sensor network in Chapter 3. Moreover, it is extended into 3D environments for micro flying robots.

In Chapter 2, we present a novel sensor network based global path planning algorithm for non-holonomic ground mobile robots. In this method, the motion of obstacles is considered as uniform linear motion. A sensor network is used to detect the static and dynamic obstacles in the 2D environments and generate the

globally shortest candidate path for each mobile robot. Comparing with other works, the main feature of this method is that the generated path is the globally optimal path in dynamic environments. It is mathematically proved that when sampling time approaches zero, the generated path converges to the globally shortest path in the environment, whereas many other algorithms do not consider the optimal path planning (e.g. [74, 117, 114, 81, 23]) or do not consider dynamic environments (e.g. [14]). Moreover, a non-holonomic constraint on the motion of the mobile robot model is considered in the proposed method unlike the most of other works. Moreover, the performance of our method is proved mathematically and other heuristic algorithms cannot be proved. (e.g. [54, 113, 26, 3, 202]). Additionally, our method only requires a low-level path tracking controller on the mobile robot. It saves more robot power in computation and sensors than other local navigation algorithms (e.g. [54, 117, 74]). Furthermore, the proposed algorithm can be easily implemented in multi-robot systems. The sensor network plans the path for each mobile robot at any time.

In Chapter 3, we present a sensor network based ground mobile robot real-time navigation algorithm. In this method, we take the advantage of the range finder sensor network to navigate all the ground mobile robots centrally in the workspace. The main feature of the proposed method is that the navigation tasks for all the mobile robots are completely transferred and integrated into the sensor network. Different types of robots can be navigated simultaneously in the workspace by the sensor network. Each robot is only required to have a low-level path tracking controller and some basic navigation sensors, like inertial navigation sensors or odometry sensors. It does not require any robot sensor for obstacle detection and any other extra navigation algorithm. Moreover, the sensor network based navigation is more flexible in configuration than local navigation in an industrial environment with different types of robots working cooperatively. New robots can be added into the workspace directly without any specialization in navigation. Additionally, a sensor network navigates robots according to the extensive measurements of the environ-

ment and performs a shorter and more efficient trajectory than local navigation algorithm. Therefore, this is an efficient, safe and economic navigation system for multiple robots in a dynamic industrial workspace. Furthermore, a practical non-holonomic industrial mobile robot model is considered in our method and dynamic environments with moving obstacles are supposed in our method, unlike other path planning algorithms e.g. [206, 210].

In Chapter 4, we present a sensor network based flying mobile robot navigation algorithm. In this method, we take the advantages of the 3D range finder sensor network to navigate the flying robots in 3D dynamic environment. The main feature of the presented method is that the navigation of all the flying robots is completely transferred and integrated into the sensor network, which is different from the local navigation algorithms; e.g. [90, 76]. Only a low-level path tracking controller is required for each robot. The robot is not equipped with any obstacle detection sensors and does not execute any complex algorithm. Different type of micro flying robot can be navigated directly by the sensor network without any specialization. In the previous works, the navigation approaches for flying robot with sensor network are proposed in [12, 27]. In the work [12], RSSI-based sensor network is used to achieve the flying robot navigation. In another work [27], binary sensors are used to indicate the surrounding obstacles. Both of these two methods simplified the flying robot navigation as 2D planar navigation, unlike our work. The main difference between our work and these two works is that, in our method, the computation load of the navigation is completely transferred to the sensor network and in the other two works, the robot obtain the environment information from the sensor network and perform the navigation on the robot's computer. Moreover, in our work, the 3D range finder sensors are used to obtain sophisticate maps of the dynamic environment and the motion planning is considered, which achieves an accurate and efficient navigation for flying robots. Because of the large detection rage of the sensor network, the trajectories of the flying robots in our navigation algorithm are shorter than local navigation algorithms. Therefore, this is an efficient, safe and

economical navigation system for multiple micro unmanned aerial vehicles (UAVs) in a dynamic environment. The proposed sensor network can be implemented as one of the fundamental units in smart factory with multiple micro flying robots working cooperatively. It also provides a centralized framework to manage and supervise the working of each micro UAV for high-level management system.

In Chapter 5 and 6, we study the map building and area exploration problem with single robot. We consider the robot only has limited capability of computation for simple algorithm. Then we presented a simple area exploration and map building algorithm for 2D environment in Chapter 5 and extended it into 3D environment for 3D map building in Chapter 6.

In Chapter 5, we present a safe area search and 2D map building algorithm for ground mobile robot in 2D environment. The main feature of our method is that it combines complete map building with obstacle avoidance for a unicycle model with constant speed, whereas some other papers do not solve the safe navigation problem for this model. For example, in the works [42, 190], different methods to select the candidate points and build a search tree are proposed. However, like the most of other frontier-based exploration methods, these methods do not consider the motion control from a candidate to next candidate for a special model. Therefore, safe navigation should be considered additionally for these methods with special robot models. Secondly, the proposed algorithm is relatively simple and computationally efficient. Therefore, the computational load of the proposed randomized algorithm is smaller than the most of frontier-based algorithms such as e.g. [124, 4, 10]. Moreover, the proposed algorithm is based on a 2D range finder sensor. Comparing with other types of sensors (see e.g. [6, 133]), the range finder measurements are simpler to process and the map is built more accurately with a range finder sensor; see e.g. [191, 190]. Furthermore, a more realistic non-holonomic model of the mobile robot's motion is considered whereas many other papers [190, 4, 124, 195, 119] in the area do not take into account non-holonomic motion constraints. Additionally, unlike many other publications in this topic, a mathematically rigorous theoretical

analysis of the developed algorithm is given.

In Chapter 6, we present an safe area search and 3D map building algorithm for ground mobile robot in 3D environment. Comparing with the previous works, the main feature of our method is that it combines the complete 3D map building and the collision-free area search navigation together in one method, unlike other works [166, 16, 78, 33]. Furthermore, we consider a 2D range finder sensor to scan the 3D structure of the environment. It performs an accurate 3D map than a monocular camera or other vision sensors; see [33]. Although there are a number of researchers using time-of-flight camera to capture depth image of the environment for map building like [89, 166, 78], a 2D range finder sensor we used is much more economical than a ToF camera and has a larger scanning angle. It can be noticed that in the work [16], the authors used the same type of sensor, 2D range finder sensor, to scan the environment. However, in the work [16], the 2D range finder sensor is connected with a motor, which can be controlled to adjust the pitch of the sensor. It makes the map building control more complicate than our method. Moreover, our proposed method is relatively simple and, therefore, it can be used in small-sized robot with limited power supply and poor computer performance, like the robot used in [170], to build an accurate 3D map, unlike the rough 2D map in [170]. Another feature of our method is that a non-holonomic model is considered, which is more practical than other works. According to the mathematical analysis, it is proved that with the probability 1 the robot can complete the 3D map building in a finite time.

1.5 Thesis outline

This thesis is organised as follows: In Chapter 2, we present a global path planning algorithm for ground mobile robots by a 2D range finder sensor network. Then based on the presented path planning algorithm, a real-time sensor network based navigation algorithm is developed in Chapter 3. After that, the algorithm is extended into 3D environments in Chapter 4.

We also present a 2D map building and area search algorithm for a ground mobile robot in Chapter 5. It is extended to solve the 3D map building problem in Chapter 6.

Chapter 2

Global Path Planning for Ground Mobile Robots

This chapter is based on the the publication [82]. It is the first stage of designing a sensor network based navigation algorithm. The presented algorithm in this chapter is an artificial potential field based path planning algorithm, which is improved in Chapter 3 and 4 and combined with other methods to perform a real-time navigation. In this chapter, we present an artificial potential field based global path planning algorithm for a non-holonomic mobile robot. In our method, the motion of obstacles is considered as uniform linear motion. A sensor network is used to detect the obstacles. The velocity of each obstacle is estimated by the sensor network. In the proposed algorithm, robot path is approximately represented as a series of equally spaced points tracked by the non-holonomic mobile robot. Some candidate paths are generated and optimized by the presented method to search the shortest candidate path. The presented navigation framework is a networked control system that the environment measurements, control input and robot's states are exchanged through the sensor network; see examples [103, 104, 150, 105].

2.1   Problem description

A planar mobile robot is modelled as an unicycle with a non-holonomic constraint in a planar environment. It is widely used to describe many ground robots, unmanned aerial vehicles and missile etc. [186, 95, 108, 44, 154, 38]. The robot travels with a constant speed v_r and is controlled by angular velocity u. The model of the vehicle is described as follows (see Fig. 2.1):
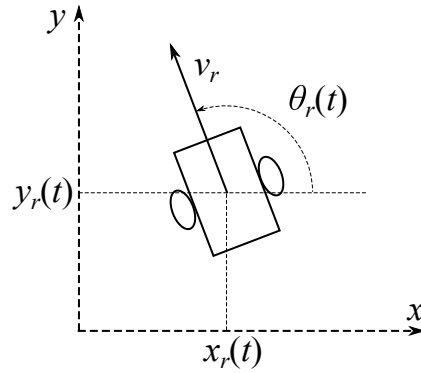
Figure 2.1: Unicycle model of the robot.

$$
\begin{cases}
\dot{x}_r(t) = v_r \cos\theta_r(t) & x_r(0) = x_0 \\
\dot{y}_r(t) = v_r \sin\theta_r(t) & , \ y_r(0) = y_0 \\
\dot{\theta}_r(t) = u(t) \in [-u_M, u_M] & \theta_r(0) = \theta_0
\end{cases}
\tag{2.1}
$$

In robot model (2.1), (x_r, y_r) is the Cartesian coordinate of the vehicle and θ_r is the robot's heading at time t. The robot's minimum turning radius is

$$
R_{\min} = \frac{v_r}{u_M}.
\tag{2.2}
$$

The robot is equipped with an odometry sensor to help to obtain the position and heading relative to its starting location and heading.

Assumption 2.1.1 *The robot's initial location (x_0, y_0) and heading θ_0 are known.*

In the planar environment, there are some disjoint static and moving obstacles $D_1, D_2, \ldots$. Notice that the shape of obstacles can be irregular and non-convex.

Definition 2.1.1 *For any i, D_i is a closed, bounded, and connected point set.*

To detect the obstacles, a sensor network is deployed on the ground. The sensor network consists of several 2D range finder sensors that measure the distances to the nearest obstacles in different directions in the scanning ranges (see Fig. 2.2). A global map consisting of the obstacles' boundaries and unoccupied areas can be built by the sensor network.
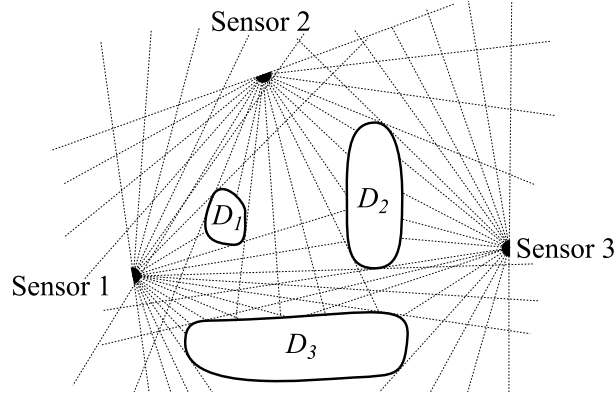
Figure 2.2: Sensor network in the planar environment.

Assumption 2.1.2 *Let p be an arbitrary point on the boundaries of obstacles. At the initial time, there exists at least one sensor that the segment between the sensor and point p does not intersect any obstacle. It means the point p can be detected by at least one sensor.*

Notation 2.1.1 *Let p be an arbitrary point. For any closed set D, the minimum distance $\rho(D, p)$ between p and D is*

$$\rho(D, p) := \min_{q \in D} \|p - q\|. \tag{2.3}$$

Definition 2.1.2 *Let $d_s > 0$ be a given safety margin. The robot should keep this safety margin to any obstacle while travelling. The d_s-enlarged region $\mathcal{E}[D, d_s]$ of a closed set D is defined as follows (see Fig. 2.3):*

$$\mathcal{E}[D, d_s] := \{p : \rho(D, p) \leq d_s\}. \tag{2.4}$$
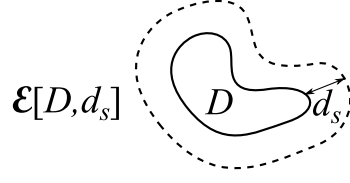
Figure 2.3: Enlarged region $\mathcal{E}[D, d_s]$.

Assumption 2.1.3 *The planar sets $\mathcal{E}[D_i, d_s]$, $i = 1, 2, \ldots$ are closed, bounded, connected and linearly connected sets.*

Assumption 2.1.4 *For any $i \neq j$, $\mathcal{E}[D_i, d_s]$ does not overlap $\mathcal{E}[D_j, d_s]$ at any time.*

Assumption 2.1.5 *The motion of any obstacle D_i, $i = 1, 2, \ldots$ is uniform linear motion with the speed $v_{D_i} \leq v_r$.*

Assumption 2.1.6 *Let $\beta = \frac{v_{D_i}}{v_r}$. For any i, the boundary of $\mathcal{E}[D_i, d_s]$ is smooth and the curvature of the boundary is smaller than $\frac{1}{(1+\beta)R_{\min}}$ at any point.*

Remark 2.1.1 *If Assumption 2.1.5 and 2.1.6 do not hold, the non-holonomic mobile robot may fail to patrol the d_s-enlarged boundaries of moving obstacles. In the proposed algorithm, patrolling the d_s-enlarged boundaries of obstacles is required.*

With the above definitions and assumptions, the sensor network can detect the complete boundary of any obstacle D_i at the initial time. Moreover, with a given sampling interval δ, the velocity v_{D_i} of each obstacle D_i can be estimated by calculating the translation during the sampling interval δ. For any obstacle D_i, let $D_i(k)$ denote the position of the obstacle at time $k\delta$. At the beginning, $D_i(k)$ can be predicted according to the detected initial position $D_i(0)$ and the estimated velocity v_{D_i} (see Fig. 2.4):

$$D_i(k) = D_i(0) + v_{D_i}k\delta t. \tag{2.5}$$

Assumption 2.1.7 *Give a target point $\mathcal{T}$. For any i and k, $\rho(D_i(k), \mathcal{T}) > d_s$.*

Assumption 2.1.8 *Let $p = (x_0, y_0)$ be the initial position of mobile robot. For any i, $\rho(D_i(0), p) > d_s$.*
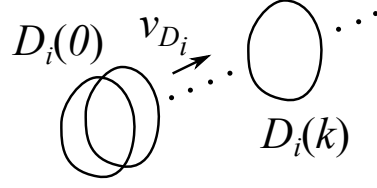
Figure 2.4: Position prediction of obstacles.

Definition 2.1.3 *Let $t_e > 0$ be the travel time of the robot. For a robot path $P = \{(x_r(t), y_r(t)) : t \in [0, t_e]\}$, if the inequality*

$$\rho(D_i(\frac{t}{\delta}), (x_r(t), y_r(t))) \geq d_s \tag{2.6}$$

holds for any $t \in [0, t_e]$ and

$$(x_r(t_e), y_r(t_e)) = \mathcal{T}, \tag{2.7}$$

then the path P is called a collision-free path.

Consider an X-Y-T spacetime for the path planning problem. For any collision-free path with a travel time $t_e > 0$, the obstacles $D_1(t), D_2(t), \ldots$ are closed and bounded set in the X-Y-T spacetime during the period $[0, t_e]$ (see Fig. 2.5); and the collision-free path is a curver from the initial position $(x_0, y_0, 0)$ to the target point $(x_r(t_e), y_r(t_e), t_e)$ (see Fig. 2.6). Then, we present the following definitions (see [145]).

Definition 2.1.4 *Let P_1 and P_2 be two collision-free paths in the X-Y-T spacetime. P_1 and P_2 are homotopic if and only if one can be continuously deformed into the other without intersecting the obstacles $D_1(t), D_2(t), \ldots$. The set of all paths that are homotopic to each other is denoted as homotopy class.*

Definition 2.1.5 *Let P be a collision-free path in the X-Y-T spacetime. If for any other paths which are homotopic to P, the travel time is larger than the travel time of P, then P is called the locally shortest path in the corresponding homotopy class.*

Definition 2.1.6 *Let P be a locally shortest path, if for any homotopy class which P does not belong to, the travel time of the locally shortest path is larger than the travel time of P, then P is called the globally shortest path.*
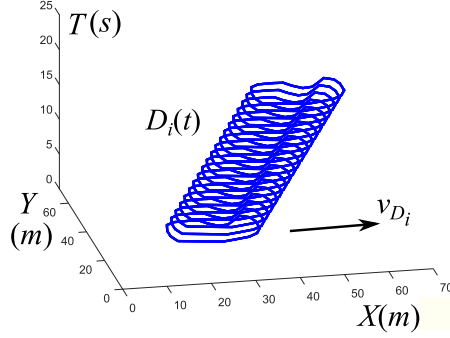
Figure 2.5: Set $D_i(t)$ in the X-Y-T spacetime.
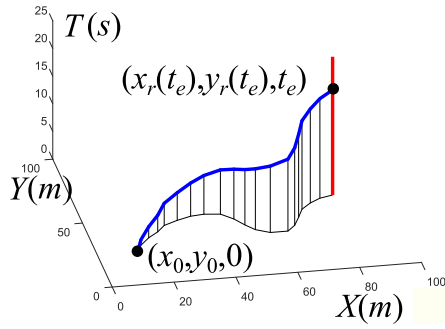
Figure 2.6: A robot path in the X-Y-T spacetime.

The objective of the proposed algorithm is to generate a path which converges towards the globally shortest collision-free path with the minimum travel time t_e. The environment measurements, robot's position and heading, and robot's angular velocity are exchanged through the sensor network to construct a networked control system; see examples [103, 104, 150, 105].

2.2   Path planning algorithm

Here, the shortest path planning algorithm is proposed. In the proposed method, a robot path P is approximately represented by finite equally spaced points $p_0, \ldots, p_n$ (see Fig. 2.7). The interval between any two successive points is

$$L = v_r \delta, \tag{2.8}$$

where δ is a the sampling interval of the computer control system.

Figure 2.7: Robot path approximately represented by a series of equally spaced points.

To search a path (denoted by P^*) which converges towards globally shortest path, several candidate paths $P_1, \ldots, P_m$ which converge towards different locally shortest paths are generated by an iterative search algorithm. Each candidate path is obtained by a proposed optimization algorithm with artificial potential fields. The path P^* is proved to be the shortest path among the obtained candidate paths $P_1, \ldots, P_m$. The path P^* can be tracked by a path tracking controller e.g. [22].

In the following parts, the path optimization algorithm and the iterative search algorithm are presented respectively.

2.2.1 Path optimization algorithm

In the path optimization algorithm, a candidate path P is prolonged gradually from the initial position $p_0 = (x_0, y_0)$ to the target $\mathcal{T}$ by generating the path points p_i, $i = 1, 2, \ldots$, one by one in the artificial potential fields with the proposed rules.

Firstly, we define three vector fields $\vec{F}_I$, $\vec{F}_R$ and $\vec{F}_P$ in the plane. For any point $p_i \in P$, $i \neq 0$, the resultant vector in the three vector fields is

$$\vec{F}(p_i) = \vec{F}_I(p_i) + \vec{F}_R(p_i) + \vec{F}_P(p_i). \tag{2.9}$$

The path P is optimized by moving any point p_i, $i \neq 0$ to an equilibrium point where $\vec{F}(p_i) = \vec{0}$. When P is optimized and the last path point of P does not reach the target $\mathcal{T}$, a new path point is added into P next to the last path point to prolong P. With the new path point, P needs to be optimized again since the equilibrium

point might changes. The optimization can be presented as follows.

A1: For each point p_i, $i \neq 0$, initialize a velocity vector $\vec{v}_i = \vec{0}$ for p_i.

A2: Start the following loop:

A2.1: Calculate $\vec{F}(p_i)$ for each point p_i, $i \neq 0$.

A2.2: Change the position and velocity of each p_i, $i \neq 0$, according to the following equations:

$$\begin{cases} p_i \leftarrow p_i + \vec{v}_i \\ \vec{v}_i \leftarrow G_N \vec{v}_i + \vec{F}(p_i) \end{cases}, \tag{2.10}$$

where $0 < G_N < 1$ is a tunable attenuation.

A3: Exit loop **A2** if $\|\vec{F}(p_i)\| < F_{th}$ for any $i \neq 0$, where $F_{th} > 0$ is a given threshold.

Now, we are here to give the definitions of the three vector fields $\vec{F}_I$, $\vec{F}_R$ and $\vec{F}_P$.

Definition of $\tilde{\mathbf{F}}_{\mathbf{I}}$: $\vec{F}_I$ is a vector field that guarantees the interval between any two successive points of P is approximately equal to L. For any $i \neq 0$, let distance $l_i = \|p_i - p_{i-1}\|$ and $\vec{c}_i$ be the unit vector pointing towards p_{i-1} from p_i (see Fig. 2.8). Then $\vec{F}_I(p_i)$ is defined as follows:

$$\vec{F}_I(p_i) = \begin{cases} G_I((l_i - L)\vec{c}_i - (l_{i+1} - L)\vec{c}_{i+1}) & , i \neq n \\ G_I((l_i - L)\vec{c}_i) & , i = n \end{cases}, \tag{2.11}$$

where $G_I > 0$ is a tunable gain.
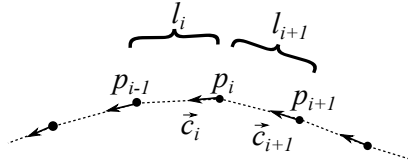
Figure 2.8: Vector $\vec{c}_i$ and distance l_i .

Definition of $\tilde{\mathbf{F}}_{\mathbf{R}}$: $\vec{F}_R$ is the repulsion around obstacles. It guarantees that, for any i, the minimum distance d_i from point p_i to all obstacles $D_j(i)$, $j = 1, 2, \ldots$ at

time $i\delta$ is $\geq d_s$. Let $\vec{r}_i$ be the unit vector from p_i to the closest obstacle at time $i\delta$ (see Fig. 2.9), then $\vec{F}_R(p_i)$ is defined as follows:

$$\vec{F}_R(p_i) = \begin{cases} G_R(d_i - d_s)\vec{r}_i & , d_i \leq d_s \\ \vec{0} & , d_i > d_s \end{cases}, \qquad (2.12)$$

where $G_R > 0$ is a tunable gain.
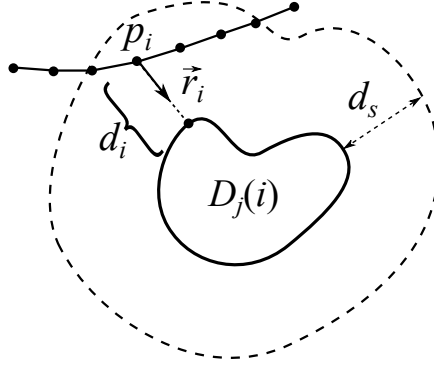
Figure 2.9: Vector $\vec{r}_i$ and distance d_i.

Definition of $\tilde{\mathbf{F}}_{\mathbf{P}}$: $\vec{F}_P$ is a vector field that only acts on the last point p_n. For other points p_i, $i \neq 0$ and n, $\vec{F}_P(p_i) = \vec{0}$. The vector $\vec{F}_P(p_n)$ guides the prolonging of the path P with two modes $R1$ and $R2$. The two modes and the mode transition rules are defined as follows (see Fig. 2.10).

R1: When mode $R1$ is active, $\vec{F}_P(p_n)$ points towards the target $\mathcal{T}$ with a constant magnitude. Thus, the path P is prolonged towards the target. Mode $R1$ is the initial mode at the beginning when P only contains the first path point p_0.

R1 $\rightarrow$ R2: If there exists an obstacle D_φ that makes $\rho(D_\varphi(n), p_n) \leq d_s$ with mode $R1$, the mode will transition to $R2$ from $R1$.

R2: When mode $R2$ is active, $\vec{F}_P(p_n)$ is a vector that can be decomposed into two mutually orthogonal components. The first component points towards the d_s-enlarged boundary radially with a linearly proportional magnitude to the difference of $\rho(D_\varphi(n), p_n)$ and d_s. Another component only has a constant magnitude. Thus, the path P is optimized and extended to bypass the obstacle D_φ with the guidance of $\vec{F}_P(p_n)$.

R2 → R1: If the line segment between p_n and $\mathcal{T}$ does not intersect the d_s-enlargement of $D_\varphi(n)$ with mode $R2$, the mode will transition to $R1$ from $R2$.
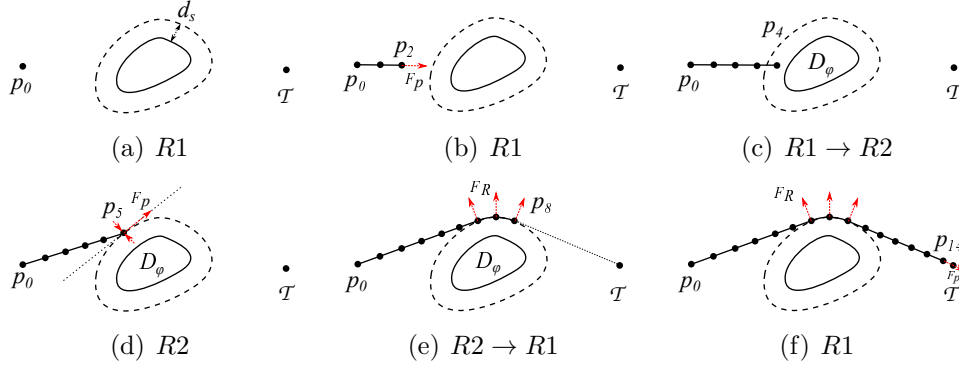
Figure 2.10: Mode transitions with a static obstacle.

Let $\vec{a}$ be a unit vector pointing towards $\mathcal{T}$ from p_n. Let h equals to $\rho(D_\varphi(n), p_n)$ and $\vec{e}$ be the unit vector pointing towards $D_\varphi(n)$ radially. Let $\vec{b}$ be a unit vector perpendicular to $\vec{e}$ (see Fig. 2.11). For the last point p_n, $\vec{F}_P(p_n)$ is defined as follows:

$$\vec{F}_P(p_n) = \begin{cases} G_P\vec{a} & , R1 \\ G_R(h - d_s)\vec{e} + \gamma G_P\vec{b} & , R2 \end{cases}, \tag{2.13}$$

where $G_P > 0$ is a tunable gain. Notice that $\gamma \in \{1, -1\}$ is a parameter that determines the direction (clockwise or anti-clockwise direction) in which the path P bypasses obstacle D_φ with the guidance of $\vec{F}_P$.
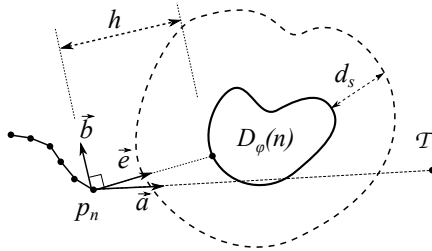
Figure 2.11: Vectors $\vec{a}$, $\vec{r}_n$, $\vec{c}$ and distance h.

Now, with the proposed path optimization algorithm **A1**-**A3**, the path prolonging algorithm is presented as follows:

B1: Initialize a candidate path $P = \{p_0(x_0, y_0)\}$. Mode $R1$ is active.

B2: Start the following loop:

B2.1: Let p_n denote the last point of P. Generate a new point $p_{n+1} \in P$ next to p_n towards the target $\mathcal{T}$ with the distance L. $n \leftarrow n + 1$.

B2.2: Optimize P by the path optimization algorithm **A1**-**A3**.

B2.3: If P meets the mode transition conditions, the active mode transitions to another mode. Additionally, if $R1 \rightarrow R2$ occurs, the parameter γ will be determined.

B3: Exit loop **B2** if $\|p_n - \mathcal{T}\| < L$.

With the proposed algorithm **A1**-**A3** and **B1**-**B3**, a candidate path P can be generated. P is proved to converge towards the locally shortest path in the corresponding homotopy class. The increase of G_I and G_R or decrease of G_P can decrease the error involved by discretization but may increase the time consumption of the algorithm or make the algorithm unstable. Therefore, the gains need to be selected according to different implementations.

2.2.2 Iterative search algorithm

In the algorithm **B1**-**B3**, a candidate path is generated by simultaneous prolonging and optimization. During this procedure, the parameter γ is determined each time when $R1 \rightarrow R2$ occurs. It is easy to know that with different combinations of the value choices of γ, the different paths belong to the different homotopy classes. Let $\mathcal{P}$ be a set of candidate paths Here we present the iterative algorithm as follows:

C1: Initialize the first candidate path $P_1 = \{p_0(x_0, y_0)\}$ and $\mathcal{P} = \{P_1\}$.

C2: Start the following loop:

C2.1: For any candidate path $P_i \in \mathcal{P}$ which does not reach the target, prolong P_i by the algorithm **B2**-**B3**. During the prolonging, at each time when $R1 \rightarrow R2$ occurs, a new candidate path with duplicate path points of P_i

is generated into $\mathcal{P}$. The path P_i and the new candidate path are with different value choice of the parameter γ.

C3: Exit loop **C2** if any candidate path $P_i \in \mathcal{P}$ reaches the target. Then the shortest candidate path is the path P^*.

The presented iterative binary search algorithm can be used to search all the homotopic classes of paths which are possible to be the shortest path in time-state space. Then the presented artificial potential field based algorithm can be used to find out the shortest path in each homotopic class with the smallest energy in time-state space with the designed potential fields. Finally, the shortest path can be obtained.

2.3   Computer simulations

In this section, computer simulations are carried out to confirm the performance of the proposed method in both static and dynamic environments. Moreover, the proposed method can be implemented with multi-robot systems. A computer simulation is carried out to show the performance of the proposed method with multiple robots.

2.3.1   Simulations in static environments

The first simulation (see Fig. 2.12) is carried out in a static environment. There are some static non-convex obstacles on the plane. With the proposed path planning algorithm, several candidate paths are generated and the shortest path P^* is chosen. Table. 2.1 shows the parameters used in this simulation. According to the simulation result, it can be seen that the mobile robot reaches the target successfully and keeps a given safety margin d_s to the obstacles (see Fig. 2.13). There are 152 samples in this simulation. The control input satisfies the bound constraint.

Table 2.1: Parameters of The First Simulation.

Name	Symbol	Value
Robot's speed	v_r	0.5m/s
Robot's maximum angular velocity	u_M	2rad/s
Sampling interval	δ	0.3s
Safe margin	d_s	0.6m
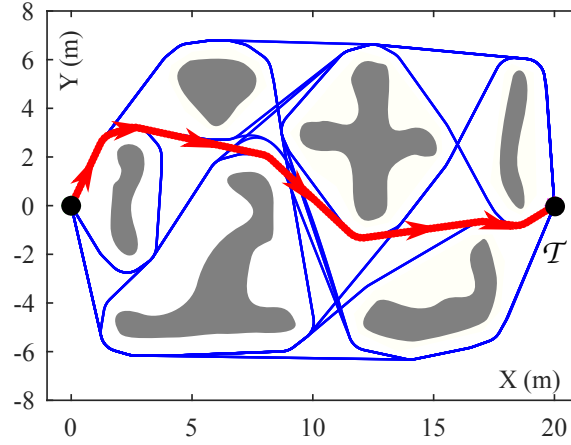
Figure 2.12: Result of the first simulation. The blue lines are all the candidate paths generates by the proposed algorithm. The red line is the trajectory of the robot.
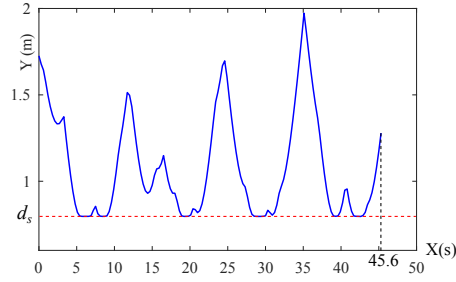
Figure 2.13: Minimum distance to obstacles in the first simulation.

2.3.2 Simulations in dynamic environments

The second simulation (see Fig. 2.14) is carried out in a dynamic environment with some moving obstacles. The motion of each obstacle is uniform linear motion. With the proposed algorithm, several robot candidate paths are generated like the first simulation. Each candidate path is a target-reaching path. According to the simulation result, the robot chooses the shortest candidate path to track and avoids all the moving obstacles with the safety margin. Table. 2.2 shows the parameters

used in this simulation. Notice that the safety margin d_s is different from the first simulation. It is changed to make the simulation result clear to be identified. Fig. 2.15 indicates that the mobile robot keeps the safety margin d_s to obstacles while travelling. The proposed algorithm is a discrete-time control system. The discretization involves the errors that cannot be eliminated in computer control system. It makes the distance lower than safety margin slightly at 14s. The control input satisfies the bound constraint.

Table 2.2: Parameters of The Second Simulation.

Name	Symbol	Value
Robot's speed	v_r	0.5m/s
Robot's maximum angular velocity	u_M	2rad/s
Sampling interval	δ	0.3s
Safe margin	d_s	0.8m

In the third simulation, we consider the collision-free path planning for multiple robots by the proposed algorithm. The paths of the robots are planned successively. When plan the i-th path, the previous $i-1$ robots are considered as obstacles with known planned path. In this simulation, there are ten mobile robots in the planar environment with different initial positions and targets. The result of the simulation (see Fig. 2.16) shows that each mobile robot tracks the planned collision-free path to the target. The Fig. 2.17 confirms that all the robots successfully avoid the collision with any other robot and keep the safety margin $d_s = 0.8$m to each other.

2.4   Summary

In this chapter, a shortest path planning algorithm is proposed here for non-holonomic mobile robots in dynamic environments. The motion of the obstacles is considered as uniform linear motion. The sensor network measures obstacles and estimates the velocity of each obstacle. With the complete information of the obstacles, the proposed method searches a path converging towards the globally shortest path with obstacle avoidance.
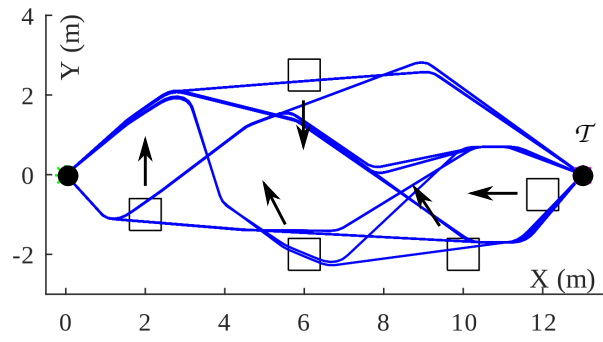
(a) $t = 0$s
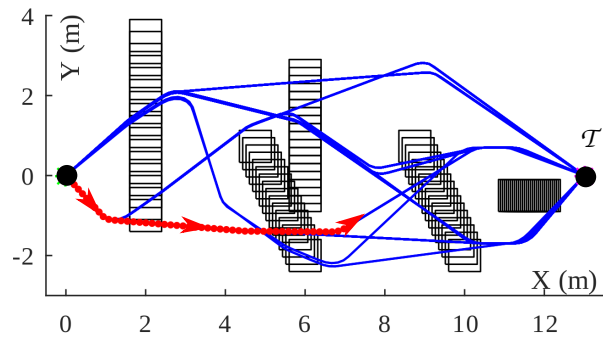
(b) $t = 15$s
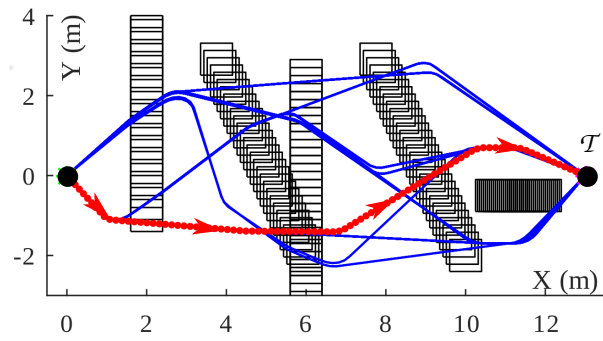
(c) $t = 27.3$s

Figure 2.14: Result of the second simulation. The blue lines is all the possible paths generates by the proposed algorithm. The red line is the trajectory of the robot. The black objects are the moving obstacles.
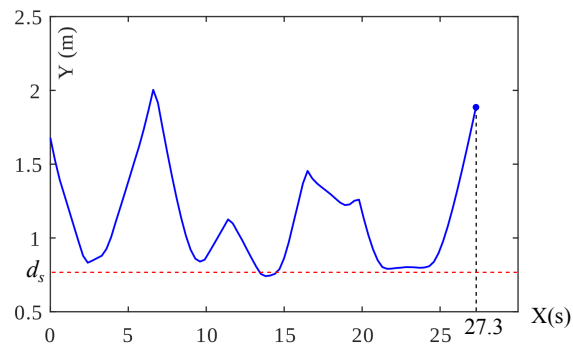
Figure 2.15: Minimum distance to obstacles in second simulation.

(a) $t = 0$s   (b) $t = 36.9$s
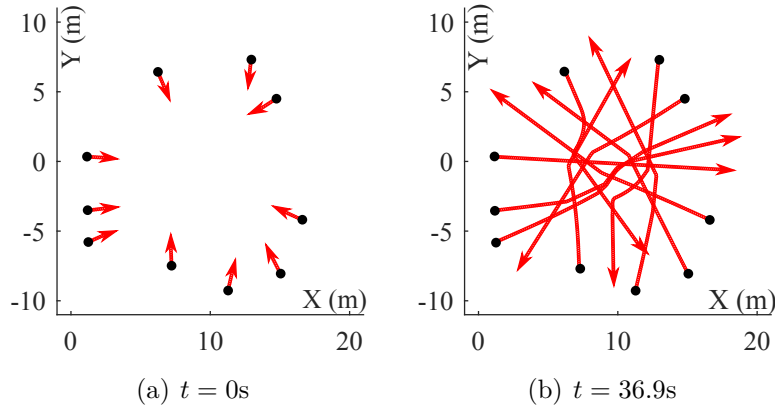
Figure 2.16: Result of the third simulation.
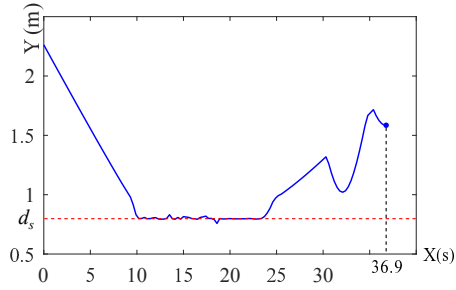
Figure 2.17: Minimum distance between any robots in the third simulation.

The proposed method has some advantages. Firstly, the proposed path planning algorithm guides the robot with a globally shortest collision-free path in a dynamic environment. Secondly, a bounded control is considered. Furthermore, the proposed global path planning algorithm can be implemented in multi-robot systems.

Chapter 3

Safe Navigation of Ground Mobile Robots in Dynamic Environments by a Sensor Network

This chapter is based on the the publications [83] and [86]. It is the second stage of designing a sensor network based navigation algorithm. The path planning algorithm presented in Chapter 2 is improved in this chapter. Moreover, a graph search algorithm is presented and combined with the path planning algorithm to perform a real-time navigation for ground mobile robot. In this chapter, we take the advantage of the range finder sensor network to navigate all the mobile robots centrally in the workspace. Each range finder sensor node is deployed in dynamic environments, such as floor, to detect walls, equipments, moving robots and walking people. Simultaneously, each robot measures its own real-time location and direction by localization, like odometry, and sends the measurements to the sensor network by the wireless communication. With the measurements of environment and robots' position, temporarily safe paths can be dynamically generated and the robots are navigated according to the generated path by the sensor network. The presented ground mobile robots navigation system is a networked control system; see examples [103, 104, 150, 105]. In the presented system, the environment measurements,

control input and robots' states are exchanged through the sensor network.

3.1   Problem description

The Robot model we considered is the unicycle model presented in Chapter 2; see Equation (2.1). In this chapter, $(x(t), y(t))$ is the Cartesian coordinates of the vehicle and $\theta(t)$ is the robot's heading at time t. The robot's minimum turning radius is

$$R_{\min} = \frac{v_r}{u_M}. \tag{3.1}$$

Assumption 3.1.1 *The robot measures its location and heading by some localization technologies, like odometry, etc.*

In planar environments, there are some static and moving obstacles, like walls, other robots and moving people. The obstacles can be non-convex and their velocity can be dynamic and unknown. To detect the obstacles, a sensor network is deployed in the workspace. The sensor network consists of some range finder sensor nodes. Each node measures the distances to the nearest obstacles in different directions within a measurement range denoted by R_s (see Fig. 3.1). Each range finder is deployed higher than any mobile robots in the workspace. It means any mobile robots on the floor cannot be detected by the sensor network. Furthermore, a central computer node connects to the sensor network to obtain the real-time measurements from each sensor node. The central computer also can obtain the location and direction of any robots in the workspace by wireless communication. After obtaining the environment information and the location and direction of each robot, the sensor network dynamically generates a safe path for each robot according to the proposed navigation algorithm and send the paths to the robots for tracking.

For each sensor node, obstacles are detected partly in the measurement range and a detected area is obtained in the local coordinate system of the sensor node (see Fig. 3.2).
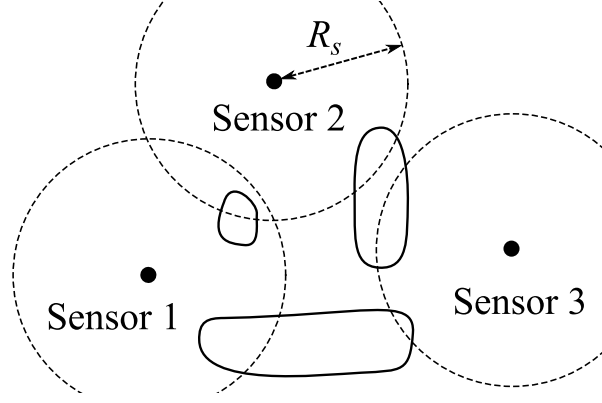
Figure 3.1: Sensor network in the planar environment.
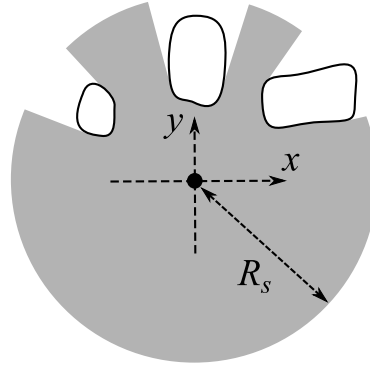
Figure 3.2: Detected area of a sensor node.

Definition 3.1.1 *For each sensor node, the detected area is a closed, bounded, and connected point set.*

Assumption 3.1.2 *For each sensor node, the location and direction where it is deployed are known.*

Assumption 3.1.3 *The physical size of any robot in the workspace can be covered by a disk with a given radius R_r on the center of the robot (see Fig. 3.3). The disk called robot disk.*

With the help of the location and direction of each sensor node, the local detected areas of all the sensor nodes at any time t can be converted to the global coordinate system, then a total detected area, which is the union of the local detected areas, is obtained. Moreover, the robot should avoid other mobile robots in the workspace. According to Assumption 3.1.3 and other robots' locations obtained by the central
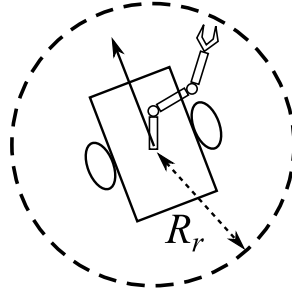
Figure 3.3: Physical size of a mobile robot.

computer, the unoccupied area denoted by $\mathcal{A}(t)$ can be calculated, which equals to the relative complement of all the robot disks in the total detected area (see Fig. 3.4).
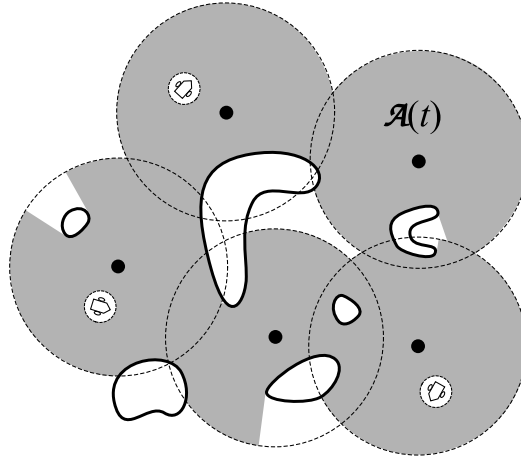
Figure 3.4: Unoccupied area $\mathcal{A}(t)$.

Notation 3.1.1 *Let p be an arbitrary point. For any closed set D, the minimum distance $\rho(D, p)$ between p and D is*

$$\rho(D, p) := \min_{q \in D} \|p - q\|. \tag{3.2}$$

Assumption 3.1.4 *Let p be an arbitrary point in the set $\mathcal{A}(t)$ and $\partial\mathcal{A}(t)$ denote the boundary of $\mathcal{A}(t)$. The derivative of the minimum distance $\rho(\partial\mathcal{A}(t), p)$ with respect to time t is smaller than or equal to $V_{\max}$, where $V_{\max}$ is a given constant which is smaller than v_r.*

Remark 3.1.1 *If Assumption 3.1.4 does not hold, the robot may fail to avoid the dynamic undetected areas, which may contain obstacles.*

Definition 3.1.2 *Let $d > 0$ be a constant. Let ∂D denote the boundary of a closed set D. The d-reduction of the closed set D is a set $\mathcal{R}[D, d]$ defined as follows (see Fig. 3.5):*

$$\mathcal{R}[D, d] := \{p \in D : \rho(\partial D, p) \geq d\}. \tag{3.3}$$
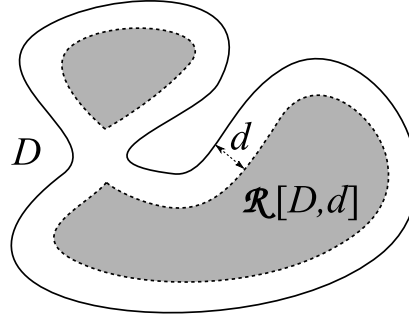
Figure 3.5: *d*-reduced region $\mathcal{R}[D, d]$ of a set D.

Definition 3.1.3 *The safety margin $d_s > 0$ is a given constant that the robot should keep from the boundary $\partial \mathcal{A}(t)$ at any time t.*

Assumption 3.1.5 *Let p be an arbitrary point on the boundary of $\mathcal{R}[\mathcal{A}(t), d_s]$. At any time t, if p is not a singularity and the osculating circle at the point p is to the different side of the tangent with $\mathcal{R}[\mathcal{A}(t), d_s]$, then the radius of the osculating circle is greater than or equal to $R_{\min}$ (see Fig. 3.6).*

Assumption 3.1.6 *The robot's initial position (x_0, y_0) belongs to the initial unoccupied area $\mathcal{R}[\mathcal{A}(0), d_s]$.*

The objective of the proposed algorithm is to drive the mobile robot to travel in the dynamic and deformable d_s-reduced area $\mathcal{R}[\mathcal{A}(t), d_s]$ and finally reach a target point denoted by $\mathcal{T}$ with a relatively short trajectory. In the presented method, the environment measurements, robot's position and heading, and robots' angular velocity are exchanged through the sensor network to construct a networked control system; see examples [103, 104, 150, 105].
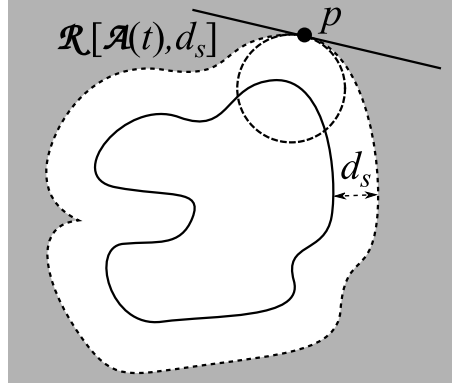
Figure 3.6: Osculating circle of the boundary of $\mathcal{R}[\mathcal{A}(t), d_s]$.

Assumption 3.1.7 *The target $\mathcal{T}$ belongs to the set $\mathcal{R}[\mathcal{A}(t), d_s]$ at any time t.*

Assumption 3.1.8 *The set $\mathcal{R}[\mathcal{A}(t), d_s]$ is a connected set at any time t.*

Assumption 3.1.9 *The distance between the target and the initial position is $>$ $2R_{\min}$.*

3.2   Safe navigation algorithm

In this section, we propose our sensor network based navigation algorithm. Firstly, we give the brief introduction of the algorithm as follows. In an environment, each node of a range finder sensor network is deployed to detect the surrounding obstacles. There is a central computer node connecting to the sensor network to collect the data from each node. Any robots in the workspace can upload the real-time locations and directions to the central computer node via the sensor network by wireless communication. Let δ be the given sampling interval. At any discrete time step $t = 0, \delta, 2\delta, \ldots$, the central computer node calculates the unoccupied area $\mathcal{A}(t)$ according to the obtained real-time information. Then it generates a relatively short safe path, denoted by P^*, from the robot's current position $(x(t), y(t))$ to the target $\mathcal{T}$. The path P^* satisfies the non-holonomic constraint of the robot's motion. Let T be a given time window, which is a positive integer. The path P^* is proved to be safe over the time period $[t, t + T\delta]$. However, the robot only tracks P^* over the next sampling interval $[t, t + \delta]$, then the algorithm repeats and updates the path

P^* at the next time step $t + \delta$. The algorithm repeats periodically at each time step and navigates the robot to the target without any collision.

Let $k = 0, 1, 2, \ldots$. To guarantee the safety of the robot in the partly detected dynamic environment during the given time window $[t, t + T\delta]$, we consider any potentially unsafe areas in the environment and give the following definition.

Definition 3.2.1 *Let t be the current time. According to Assumption 3.1.4, the area which is obsoletely safe at time $t + k\delta$ is $\mathcal{R}[\mathcal{A}(t), k\delta V_{\max}]$. Therefore we define an area $\hat{\mathcal{A}}(t, k)$ as follows to help to generate a temporarily safe path:*

$$\hat{\mathcal{A}}(t, k) := \begin{cases} \mathcal{R}[\mathcal{A}(t), k\delta V_{\max}] & \text{if } k \leq T \\ \mathcal{R}[\mathcal{A}(t), T\delta V_{\max}] & \text{if } k > T \end{cases}. \tag{3.4}$$

Assumption 3.2.1 *The target point $\mathcal{T}$ belongs to the set $\mathcal{R}[\mathcal{A}(t), d_s + T\delta V_{\max}]$ at any time t.*

Assumption 3.2.2 *The set $\mathcal{R}[\mathcal{A}(t), d_s + T\delta V_{\max}]$ is a connected set at any time t.*

Let p_0 denote the robot's current position $(x(t), y(t))$ at the current time t. A robot path can be approximately represented as some finite equally spaced points $p_0, p_1, \ldots, p_n$ (see Fig. 3.7). Each point p_k represents the position of the robot at the future time $t + k\delta$. The interval between any two successive points is a constant L which equals to $v_r \delta$. According to the non-holonomic constraint, the radius of the circumscribed circle of any three successive points p_{k-1}, p_k and p_{k+1} should be greater than or equal to $R_{\min}$ and the angle between the robot's current heading $\theta(t)$ and the vector from p_0 to p_1 should be smaller than or equal to $\arcsin \frac{L}{2R_{\min}}$.

Definition 3.2.2 *Let $P = \{p_0, \ldots, p_n\}$ be a robot path at time t. For any k, if the point p_k belongs to the set $\mathcal{R}[\hat{\mathcal{A}}(t, k), d_s]$, P is called a temporarily safe path. It is guaranteed that the path P is absolutely safe over the time period $[t, t + T\delta]$.*

Definition 3.2.3 *Let $P = \{p_0, \ldots, p_n\}$ be a robot path. If $\|\mathcal{T} - p_n\| \leq L$, the path P is called a target-reaching path. It guides the robot to the target $\mathcal{T}$.*
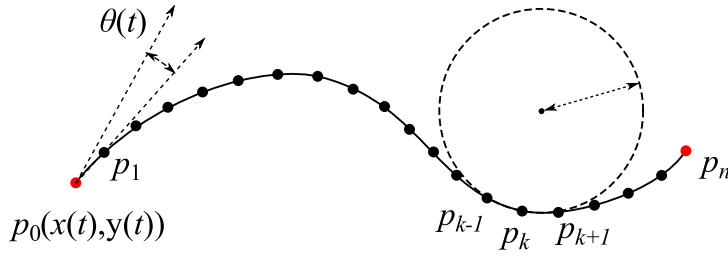
Figure 3.7: Robot path represented as some equally spaced points.

Considering the minimum turning radius of the robot, we define two circles called initial circles [157] as follows.

Definition 3.2.4 *The two initial circles are tangent to the robot's current heading* $\theta(t)$ *and cross the robot's current position* $p_0(x(t), y(t))$. *The radius of each initial circle is equal to* $R_{\min}$ *(see Fig. 3.8).*
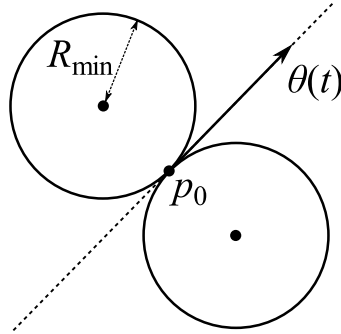
Figure 3.8: Two initial circles.

Moreover, according to [145], we give the following definition of homotopic paths.

Definition 3.2.5 *Let* P_1 *and* P_2 *be two paths with same initial point and target.* P_1 *and* P_2 *are homotopic if and only if one can be continuously deformed into the other without intersecting the boundaries of the area* $\mathcal{A}(t)$ *and the initial circles. The set of all paths that are homotopic to each other is denoted as homotopy class.*

To generate the relatively short target-reaching path P^*, firstly, a path planning algorithm is proposed to adjust a candidate path to an approximate shortest temporarily safe path among the homotopy class of the given path. It is followed by a

graph search algorithm that generates several appropriate candidate paths, which belong to different homotopy classes. Combining the graph search algorithm with the path planning algorithm, different candidate paths, which belong to different homotopy classes, are adjusted. Then, the path P^* is selected as the shortest adjusted candidate paths.

3.2.1 Path planning algorithm

Let $P = \{p_0, p_1, \ldots, p_n\}$ be a given candidate path that needs to be adjusted. A path planning algorithm is proposed here to adjust the path P to an approximate shortest temporarily safe path among the homotopy class of the path P. Any point of P should be in the unoccupied area $\mathcal{A}(t)$.

We define four vector fields $\vec{F}_I$, $\vec{F}_R$, $\vec{F}_P$ and $\vec{F}_C$ in the plane. For any $k \neq 0$, the resultant vector at point p_k is

$$\vec{F}(p_k) = \vec{F}_I(p_k) + \vec{F}_R(p_k) + \vec{F}_P(p_k) + \vec{F}_C(p_k). \tag{3.5}$$

The path P is adjusted by moving any p_k, $k \neq 0$ towards the direction of $\vec{F}(p_k)$ to an equilibrium point where $\vec{F}(p_k) = \vec{0}$. Moreover, while adjusting, some path points may be added or removed to prolong or shorten the length of P until $\vec{F}(p_k) = \vec{0}$ for any k. Now, we are here to give the definitions of four vector fields.

Definition of $\tilde{\mathbf{F}}_\mathbf{I}$: $\vec{F}_I$ is a vector field which guarantees that the interval between any two successive points of P is approximately equal to L. For any $k \neq 0$, let $\vec{l}_k$ be the vector from p_k to p_{k-1} (see Fig. 3.9), then $\vec{F}_I(p_k)$ is defined as follows:

$$\vec{F}_I(p_k) := \begin{cases} G_I(b(k)\vec{l}_k - b(k+1)\vec{l}_{k+1}) & \text{if } k \neq n \\ G_I(b(k)\vec{l}_k) & \text{if } k = n \end{cases}, \tag{3.6}$$

where $G_I > 0$ is a tunable gain and $b(k)$ is

$$b(k) := 1 - \frac{L}{\|\vec{l}_k\|}. \tag{3.7}$$
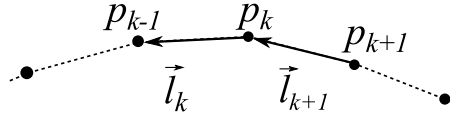
Figure 3.9: Vector $\vec{l}_k$.

Definition of $\tilde{\mathbf{F}}_{\mathbf{R}}$: $\vec{F}_R$ is a vector field which guarantees that the minimum distance from the point p_k to the boundary of $\hat{\mathcal{A}}(t,k)$ is greater than or equal to d_s for any $k \neq 0$. Let $\vec{r}_k$ be the shortest vector from p_k to the boundary of $\mathcal{A}(t)$, then $\vec{F}_R(p_k)$ is defined as follows (see Fig. 3.10):

$$\vec{F}_R(p_k) := \begin{cases} G_R(1 - \frac{\hat{d}_s(k)}{\|\vec{r}_k\|})\vec{r}_k & \text{if } \|\vec{r}_k\| \leq \hat{d}_s(k) \\ \vec{0} & \text{if } \|\vec{r}_k\| > \hat{d}_s(k) \end{cases}, \tag{3.8}$$

where $G_R > 0$ is a tunable gain and $\hat{d}_s(k)$ is

$$\hat{d}_s(k) := \begin{cases} d_s + k\delta V_{\max} & \text{if } k \leq T \\ d_s + T\delta V_{\max} & \text{if } k > T \end{cases}. \tag{3.9}$$
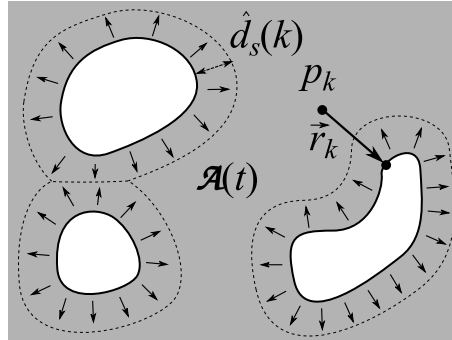
Figure 3.10: Vector field $\vec{F}_R$ and vector $\vec{r}_k$.

Definition of $\tilde{\mathbf{F}}_{\mathbf{P}}$: $\vec{F}_P$ is a vector field that only acts on the last point p_n. For any other $k \neq n$, $\vec{F}_P(p_k) = \vec{0}$. For the point p_n, let $\vec{a}$ be a unit vector pointing towards $\mathcal{T}$ from p_n (see Fig. 3.11), then $\vec{F}_P(p_n)$ is defined as follows:

$$\vec{F}_P(p_n) := G_P\vec{a}, \tag{3.10}$$

where $G_P > 0$ is a tunable gain.
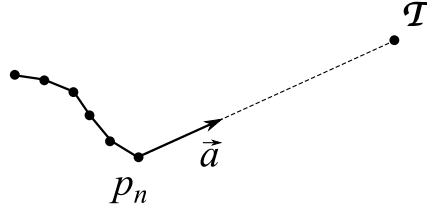
Figure 3.11: Vectors $\vec{a}$.

Definition of $\tilde{\mathbf{F}}_{\mathbf{C}}$: $\vec{F}_C$ is a vector field which guarantees that the path satisfies the non-holonomic constraint at the beginning. For any k, let $\vec{h}_k$ be the vector from the point p_k to the centre of the initial circle which is to the same side of the robot's heading direction with the point p_1 (see Fig. 3.12), then $\vec{F}_C(p_k)$ is defined as follows:

$$\vec{F}_C(p_k) := \begin{cases} G_C(1 - \frac{R_{\min}}{\|\vec{h}_k\|})\vec{h}_k & \text{if } \|\vec{h}_k\| \leq R_{\min} \\ \vec{0} & \text{if } \|\vec{h}_k\| > R_{\min} \end{cases}, \qquad (3.11)$$

where $G_C > 0$ is a tunable gain.
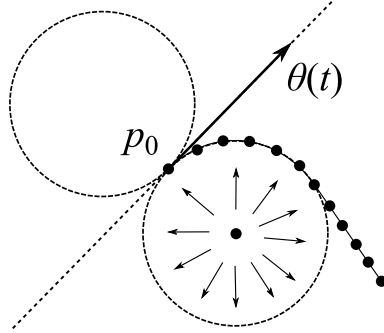
Figure 3.12: Vector field $\vec{F}_C$.

Now, we are here to propose the path planning algorithm as follows to adjust the given candidate path P:

A1: For each point $p_k \in P$, $k \neq 0$, initialize a velocity vector $\vec{v}_k = \vec{0}$ for p_k.

A2: Start the following loop:

A2.1: Calculate the resultant vector $\vec{F}(p_k)$ for any point p_k.

A2.2: Change the position vector and velocity vector of each p_k as follows:

$$\begin{cases} p_k \leftarrow p_k + \vec{v}_k \\ \vec{v}_k \leftarrow G_N \vec{v}_k + \vec{F}(p_k) \end{cases}, \tag{3.12}$$

where $0 < G_N < 1$ is a tunable attenuation.

A2.3: Let p_{last} denote the last point of P. If $\|p_{\text{last}} - \mathcal{T}\| < \underline{L}$, remove p_{last} from P. If $\|p_{\text{last}} - \mathcal{T}\| > \overline{L}$, add a new path point next to p_{last} in P. Here $\underline{L}$ and $\overline{L}$ are two given constants, which satisfy the inequalities $0 < \underline{L} < L$ and $L < \overline{L} < 2L$.

A3: Exit loop **A2** if the inequality $\|\vec{F}(p_k)\| < F_{th}$ holds for any $k \neq 0$, where $F_{th} > 0$ is a given threshold.

According to the proposed path planning algorithm, an approximate shortest temporarily safe path among the homotopy class of the path P can be obtained, which satisfies the criteria defined in Definition 3.2.2 and the non-holonomic constraint; e.g. see Fig. 3.13. The increase of G_I, G_R and G_C or decrease of G_P can decrease the error involved by discretization but may increase the time consumption of the algorithm or make the algorithm unstable. Therefore, the gains need to be selected according to different implementations.

(a) The path before adjustment   (b) The path after adjustment
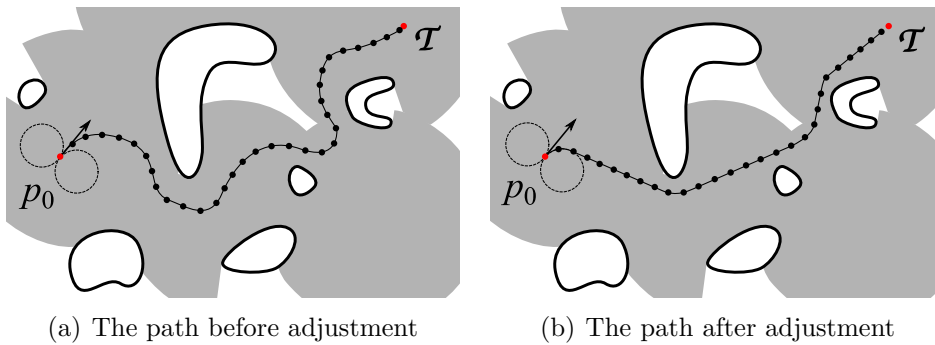
Figure 3.13: Adjustment of a given target-reaching path.

3.2.2   Candidate paths generation

A graph search algorithm is proposed here to generate some target-reaching candidate paths belonging to the different homotopy classes. Then, these candidate paths

are adjusted according to the algorithm **A1**-**A3** to search the path P^*. Firstly we introduce a graph as follows.

The boundary of the d_s-reduced area $\mathcal{R}[\mathcal{A}(t), d_s]$ can be represented as the union of some simple closed curves denoted by $C_0, C_1, \ldots$. Let C_0 denote the largest curve which encircles other curves $C_1, C_2, \ldots$.

Definition 3.2.6 *For any i, if a line passing through the target $\mathcal{T}$ is tangent to C_i and the segment between the tangent point and the target $\mathcal{T}$ does not intersect with any curve, the tangent point is called an (A)-point and the segment is called an (AT)-segment (see Fig. 3.14(a)); if the segment between the tangent point and the target $\mathcal{T}$ intersects with any C_j, $i \neq j$, the tangent point is also called an (A)-point and the closest point of intersection to the (A)-point is called an (A')-point; the segment between a pair of (A)-point and (A')-point is called an (AA')-segment (see Fig. 3.14(b)). Furthermore, a ray that its endpoint is the robot's position $p_0(x(t), y(t))$ and its direction is opposite to the target $\mathcal{T}$ intersects with the curve C_0 at some points which are called (S)-points (see Fig. 3.14(c)).*

According to Definition 3.2.4, there exist two initial circles. For each initial circle, there exists one tangent line which passes through the target $\mathcal{T}$ and is able to be tracked by the robot to exit the initial circle from the tangent point. If the tangent point is not encircled by any curve C_i, $i \neq 0$, it is called a (B)-point. If the segment between the (B)-point and the target $\mathcal{T}$ does not intersect with any curve, the segment is called a (BT)-segment (see Fig. 3.14(d)); otherwise the closest point of intersection to the (B)-point is called a (B')-point and the segment between a pair of (B)-point and (B')-point is called a (BB')-segment (see Fig. 3.14(e)). Moreover, if the initial circles intersect with any curve, the points of intersection are called (V)-points (see Fig. 3.14(f)).

Definition 3.2.7 *A graph denoted by $\mathcal{G}$ is introduced that its vertices are the target $\mathcal{T}$, robot's position $p_0(x(t), y(t))$, the points of (A), (A'), (B), (B'), (S) and (V) types. Its edges are the segments of the curves $C_0, C_1, \ldots$, the arc of the initial circles and the segments of (AT), (AA'), (BT) and (BB') types (see Fig. 3.15).*

(a) (AT)-segment   (b) (AA')-segment   (c) (S)-point

(d) (BT)-segment   (e) (BB')-segment   (f) (V)-point
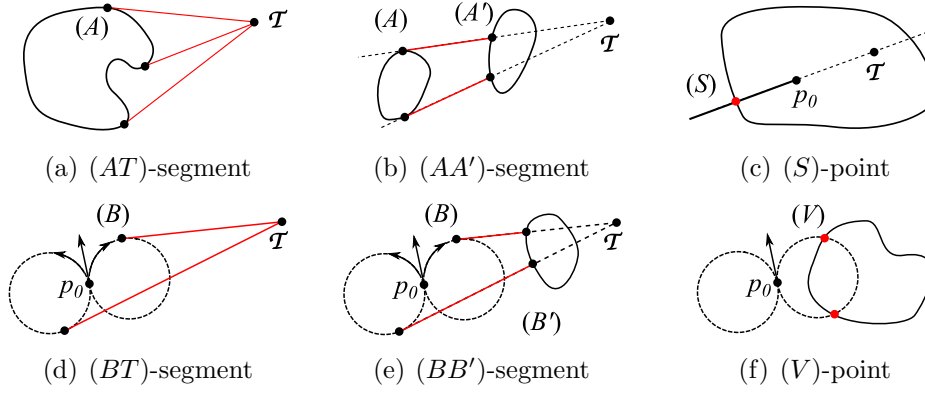
Figure 3.14: Different types of segments and points.
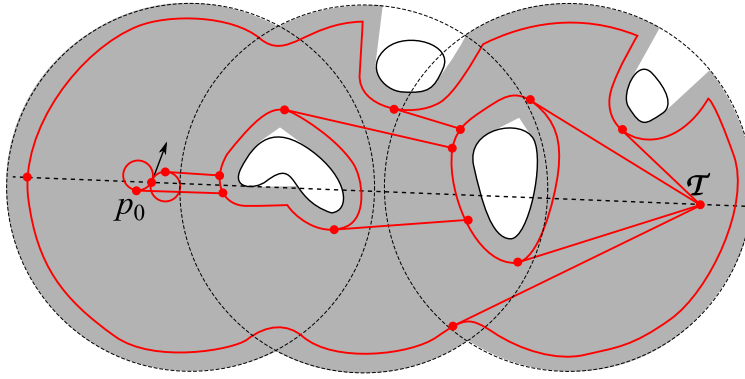
Figure 3.15: Graph $\mathcal{G}$. The red lines are the edges of the graph including the boundaries of the enlarged obstacles and the two initial circles. The red points are the vertices of the graph including the robot's position and the target.

Now, we propose the candidate paths generation rules as follows to generate some candidate paths iteratively according to the graph $\mathcal{G}$:

1. At the beginning, initialize two candidate paths, each of which only includes the point p_0. Generate new points along the different initial circle to prolong each candidate path respectively until the candidate path meets a (B)-point or a (V)-point.

2. If a candidate path meets an (A)-point or a (B)-point, continue generating new points along the corresponding segment of (AA'), (AT), (BB') or (BT) type until the candidate path meets an (A')-point, a (B')-point or the target $\mathcal{T}$.

3. If a candidate path meets an (A')-point or a (B')-point, generate a new can-

didate path with the duplicate points of this candidate path, then continue generating new points of these two candidate paths along the curve in different directions until each candidate path meets an (A)-point or an (S)-point.

4. If a candidate path meets a (V)-point, continue generating new points along the curve in the same direction with the initial circle (clockwise or anti-clockwise direction) until the candidate path meets an (A)-point or an (S)-point.

5. If a candidate path meets an (S)-point, this candidate path is abandoned.

6. If a candidate path meets the target $\mathcal{T}$, the generation of this candidate path is completed.

By following the paths generation rules, some target-reaching candidate paths can be generated iteratively according to the graph $\mathcal{G}$ (e.g. Fig. 3.16). Then, all the candidate paths are adjusted according to the proposed path planning algorithm **A1**-**A3**. Finally, the shortest adjusted path is selected as the path P^*, which should be tracked by the robot.

(a)

(b)

(c)

(d)
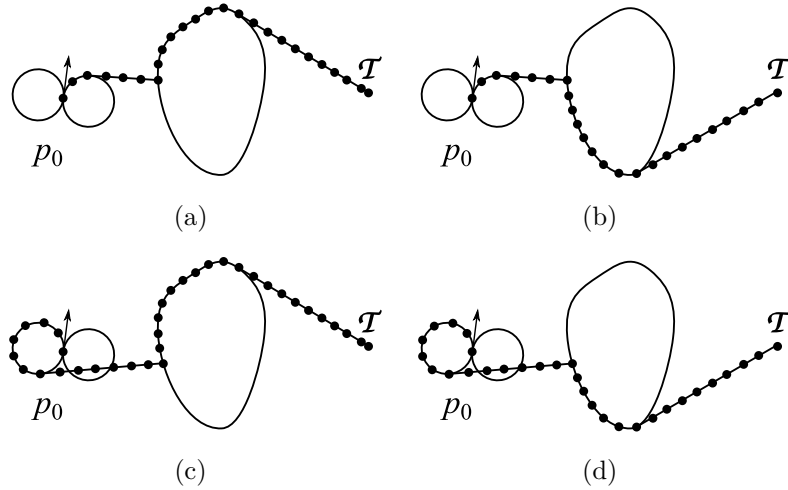
Figure 3.16: Target-reaching candidate paths generated by the graph $\mathcal{G}$.

Remark 3.2.1 *Instead of adjusting all the candidate paths, we can only select and adjust the shortest candidate path. It reduces the computation of the navigation algorithm in practical implementations.*

Notice that there exist two cases, in which the adjustment of a candidate path cannot be successful and the corresponding candidate path should be abandoned. The first case is that the vector field $\vec{F}_C$ clashes with the field $\vec{F}_R$. The second case is that the field $\vec{F}_R$ clashes with itself.

3.3   Computer simulations

In this section, computer simulations are carried out to confirm the performance of the proposed navigation algorithm in static and dynamic environments. In the presented computer simulations, the practical scenes are simulated with static and moving obstacles. In the scenes, the static obstacles can be walls, machines and equipments and the moving obstacles can be walking people and other moving robots. The presented simulations focus on the navigation of a single robot to assess the performance of the proposed navigation algorithm. In general cases with multiple robots, any robots can be navigated simultaneously to the destination by the sensor network.

Firstly, to drive the ground mobile robot to track the generated path P^* during each sampling interval, we are here to give a simple control strategy, which is a modification of the control law proposed in [107]. Let $e(t)$ be the minimum distance from the robot's position to the path P^*, which is a polygonal chain with vertices $p_0, p_1, \ldots, p_n$ belonging to P^* (see Fig. 3.17), and make the minimum distance positive if the closest point on the polygonal chain to the robot's position is in the upper half-plane of the robot's local coordinate system and negative if in the lower half-plane. Let $\lambda > 0$ and $\sigma > 0$ be tunable constants. The angular velocity of the robot is controlled as follows:

$$u(t) = u_M \text{sgn}[\dot{e}(t) + \mathcal{X}(e(t))]. \tag{3.13}$$

The saturation function $\mathcal{X}(z)$ is

$$\mathcal{X}(z) := \begin{cases} \lambda z & \text{if } |z| \leq \sigma \\ \lambda\sigma\text{sgn}(z) & \text{if } |z| > \sigma \end{cases} \tag{3.14}$$

and the sign function $\text{sgn}(x)$ is

$$\text{sgn}(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x < 0 \end{cases} \cdot \tag{3.15}$$
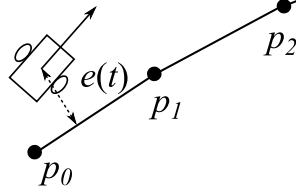
Figure 3.17: Minimum distance $e(t)$.

Now, we are here to carry out four simulations in different scenes including static and dynamic environments. To reduce the computational load, only the shortest candidate path generated according to the graph $\mathcal{G}$ is selected and adjusted, instead of all the candidate paths.

3.3.1 Simulations in static environments

To confirm the performance of the proposed navigation algorithm in static environments, we build two static scenes with same static obstacles but different deployment of the sensor nodes.

In the first scene, there are some static, non-convex and irregular-shaped obstacles. To detect these obstacles, there are some range finder sensors deployed in the scene (see Fig. 3.18). The main parameters in this simulation are indicated in Table 3.1. The parameters V_{max} and T are 0.5m/s and 4. In this scene, a robot moves from the initial position to a target (see Fig. 3.19). It can be seen that the robot's trajectory is relatively short. During the travelling, the robot avoids all the obsta-

cles and undetected areas successfully and keeps the given safety margin (see Fig. 3.20). The proposed algorithm is a discrete-time control system. The discretization involves the errors that cannot be eliminated in computer control system. It makes the distance lower than safety margin slightly at 47s. The control input satisfies the bound constraint.

Table 3.1: Main Parameters for the First Scene

Measurement range	R_s	15m
Speed of robot	v_r	1m/s
Maximum angular velocity	u_M	0.5rad/s
Safety margin	d_s	1m
Sampling interval	δ	1s
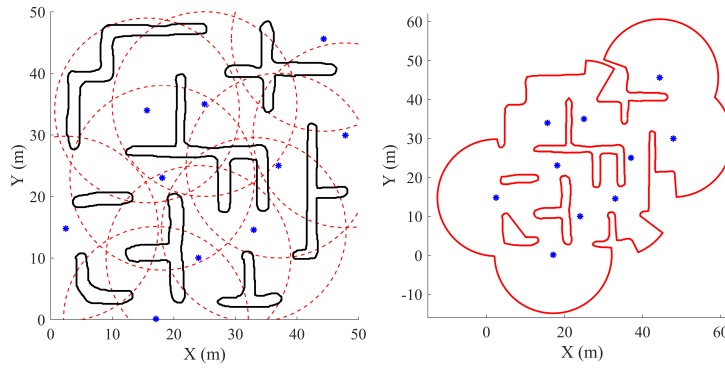
Figure 3.18: First scene and the boundaries of the unoccupied area. The asterisks are sensor nodes. The dashed lines are the measurement range of each sensor node.

Figure 3.19: Simulation in the first scene. The green circle is the target. The blue line is the trajectory of the robot.
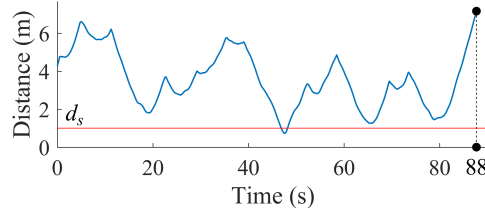
Figure 3.20: Minimum distance to the undetected areas in the first scene.

In the second scene, there are the same obstacles with the first scene. Comparing with the first scene, less range finder sensors are deployed (see Fig. 3.21). All the parameters, robot's initial position and target are same as the first simulation. In this scene, a robot successfully reaches the target with a safe and relatively short trajectory in the unoccupied area (see Fig. 3.22). During the travelling, the robot keeps the given safety margin to the undetected areas (see Fig. 3.23). The control input satisfies the bound constraint.
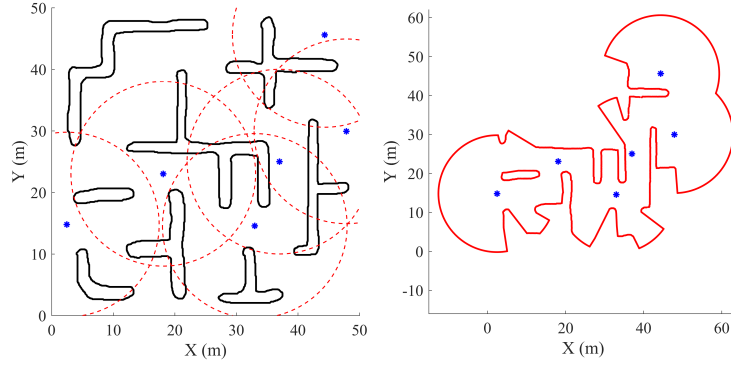
Figure 3.21: Second scene and the boundaries of the unoccupied area. The asterisks are sensor nodes. The dashed lines are the measurement range of sensor nodes.

3.3.2 Simulations in dynamic environments

To confirm the performance of the proposed navigation algorithm in dynamic environments, we build another two scenes with moving obstacles. The moving obstacles can be walking people and other robots.

In the third scene, there are four obstacles moving in the plane. To detect these obstacles, there are four range finder sensors deployed in the scene (see Fig. 3.24). The safety margin d_s, the measurement range R_s of the sensor nodes, the parameter
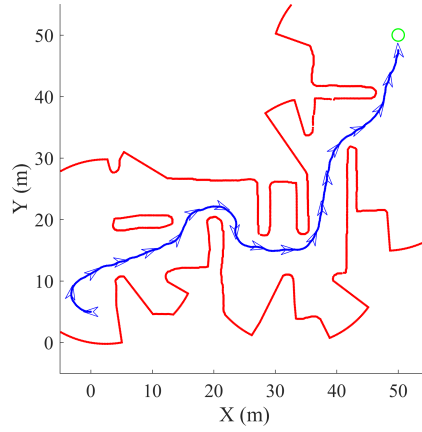
Figure 3.22: Simulation in the second scene. The green circle is the target. The blue line is the trajectory of the robot.
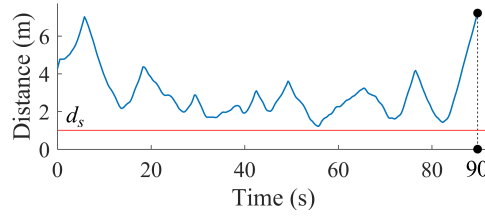
Figure 3.23: Minimum distance to the undetected areas in the second scene.

T and the maximum speed of obstacle $V_{\max}$ are 3m, 50m, 7 and 0.5m/s in this simulation. The other parameters are the same as Table. 3.1. In this scene, these obstacles can be detected completely by the sensor network. A robot moves towards a given target in this scene and avoids the moving obstacles (see Fig. 3.25). During the travelling, the robot keeps the given safety margin to the obstacles as we expect (see Fig. 3.26). The control input satisfies the bound constraint.

In the fourth scene, there are three moving obstacles arranged in the environment. The measurement range R_s of the sensor nodes is 50m. Therefore, the obstacles in this scene cannot be detected completely (see Fig. 3.27). Other parameters in this simulation are the same as the third simulation. In this scene, a robot moves with a safe and relatively short trajectory and avoids the obstacles and the dynamic deformed undetected areas (see Fig. 3.28). It can be seen that the robot keeps the given safety margin to the dynamic undetected areas (see Fig. 3.29). This simulation indicates that the mobile robots can be navigated to avoid any possible obstacles which are not detected by the sensor network. The control input satisfies
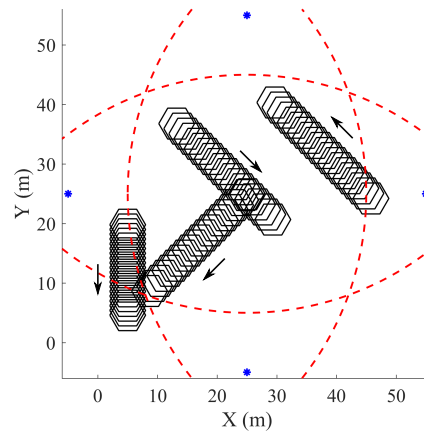
Figure 3.24: Third scene. The asterisks are sensor nodes. The dashed lines are the measurement range of sensor nodes.

(a) $t = 20$s

(b) $t = 40$s

(c) $t = 50$s

(d) $t = 78$s
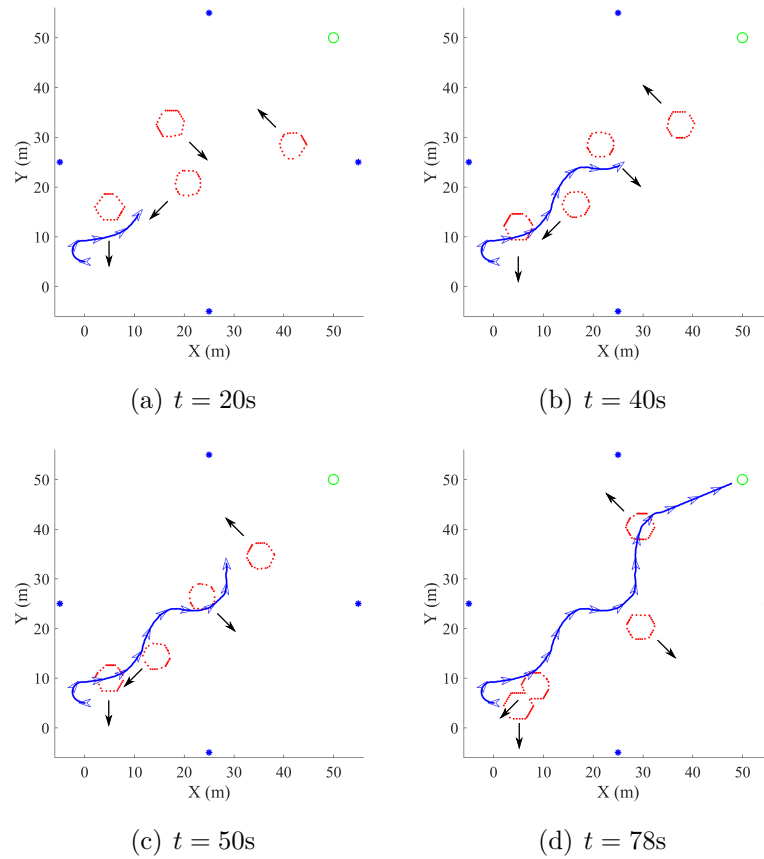
Figure 3.25: Simulation in the third scene. The green circle is the target. The blue line is the trajectory of the robot.

the bound constraint.
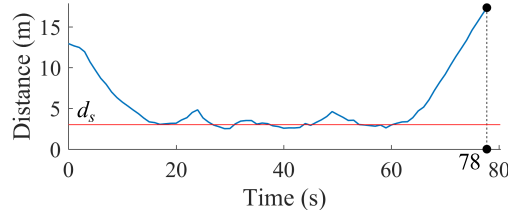
Figure 3.26: Minimum distance to the undetected areas in the third scene.
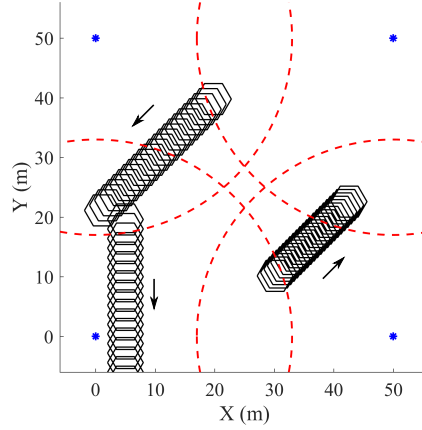
Figure 3.27: Fourth scene. The asterisks are sensor nodes. The dashed lines are the measurement range of sensor nodes.

3.4   Experiments with real mobile robot

To evaluate and confirm the performance of the proposed navigation system in practical environments, three experiments with a range finder sensor network and a real mobile robot are carried out in this section. In the presented experiments, the similar scenes to practical environments are arranged with some static obstacles and moving people. The presented experiments focus on the navigation of a single robot. In general cases with multiple robots, any robots can be navigated respectively and simultaneously to the targets by the sensor network.

In the experiments, a real sensor network is deployed on the floor of an indoor environment. The sensor network consists of three SICK LMS-200 laser range finders (see Fig. 3.30(a)), which are connected to a central computer node. Each laser range finder's scanning range is 180°. The location and direction of each laser range finder are known previously. The central computer is programmed with the proposed navigation algorithm. It collects measurements of the environment and

(a) $t = 5$s

(b) $t = 20$s

(c) $t = 30$s

(d) $t = 40$s

(e) $t = 60$s

(f) $t = 89$s
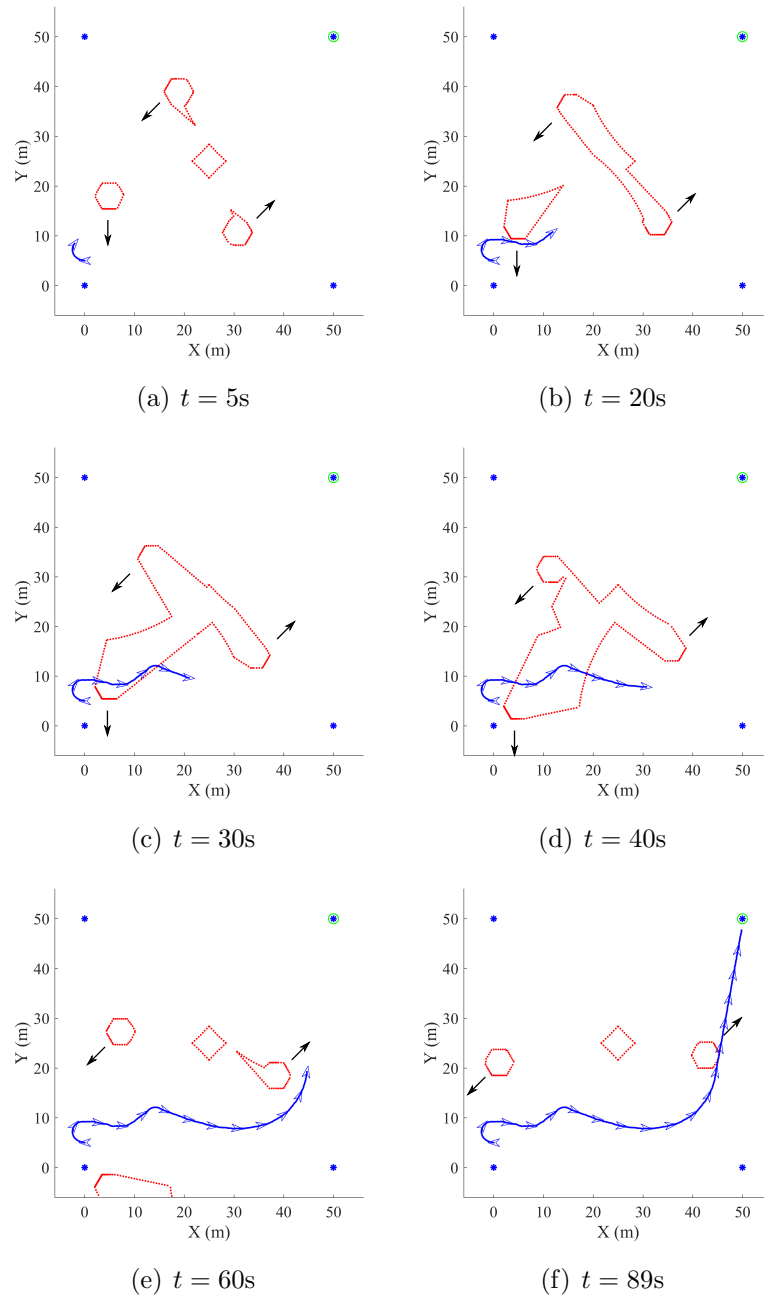
Figure 3.28: Simulation in the fourth scene. The green circle is the target. The blue line is the trajectory of the robot.
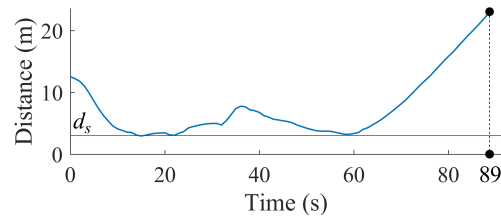
Figure 3.29: Minimum distance to the undetected areas in the fourth scene.

obstacles from each sensor node, obtains the location and heading of the mobile robot from the robot. Then, the computer calculates the safe path P^* according to the obtained information and sends the path P^* to the mobile robot by wireless communication. The target coordinates is predetermined. The real mobile robot used in the experiments is a Pioneer3-DX mobile robot (see Fig. 3.30(b)). It has an on-board computer and a wireless network device. It can upload the location and heading direction to the central computer and receive the path P^* from the central computer. The mobile robot is programmed with the proposed controller (3.13) to track the received path P^*. The mobile robot measures its location and heading direction by odometry. The robot's initial location is $(0, 0)$ and the initial heading is $0°$. The odometry error can be ignored in the experiments because the scenes in the experiments are not large. In larger and practical factories, other approaches can be used to reduce the odometry error e.g. [34, 35, 148]. The inertial navigation sensors also can be used to increase the accuracy of the localization. The presented algorithm in this chapter is designed based on the algorithm in Chapter 2. The suboptimal path is calculated at each time step with a time consumption of about $1s$. Therefore, it can be implemented with a real sensor network and ground mobile robot.

(a) SICK LMS-200 laser range finder   (b) Pioneer3-DX robot

Figure 3.30: Laser range finder and mobile robot used in the experiments.

3.4.1 Experiments in static environments

There were two experiments made in static indoor environments. Some folding cartons were arranged as static obstacles. The sensor network was deployed in the environment to detect the obstacles. The main parameters in the experiments are indicated in Table 3.2. Notice that the safety margin d_s in the second experiment was changed to 0.6m.

In the first experiment (see Fig. 3.31), it can be seen that three laser range finders detected different parts of the environment and the central computer combine the measurements from each sensor node. Then, it navigated the mobile robot from the initial position to the predetermined target without collision. The experimental result (see Fig. 3.32) shows that the sensor network mapped the environment correctly and navigated the mobile robot accurately with a relatively short trajectory. According to Fig. 3.33, it can be seen that the distance from the obstacles and undetected areas to the mobile robot were larger than the safety margin d_s while the robot was travelling. There are 61 samples in this simulation. The control input satisfies the bound constraint.

Table 3.2: Main Parameters for the First Experiment

Measurement range	R_s	8m
Speed of robot	v_r	0.3m/s
Maximum angular velocity	u_M	1.2rad/s
Safety margin	d_s	0.5m
Sampling interval	δ	1s

Similarly, in the second experiment (see Fig. 3.34), the sensor network navigated the mobile robot to avoid any obstacle on the floor and reach the target. The experimental result (see Fig. 3.35) shows the trajectory of the robot and the map built by the sensor network. According to Fig. 3.36, it can be seen that the mobile robot was keeping the safety margin d_s while travelling. The control input satisfies the bound constraint.

(a) $t = 0$s

(b) $t = 34$s

(c) $t = 49$s

(d) $t = 66$s
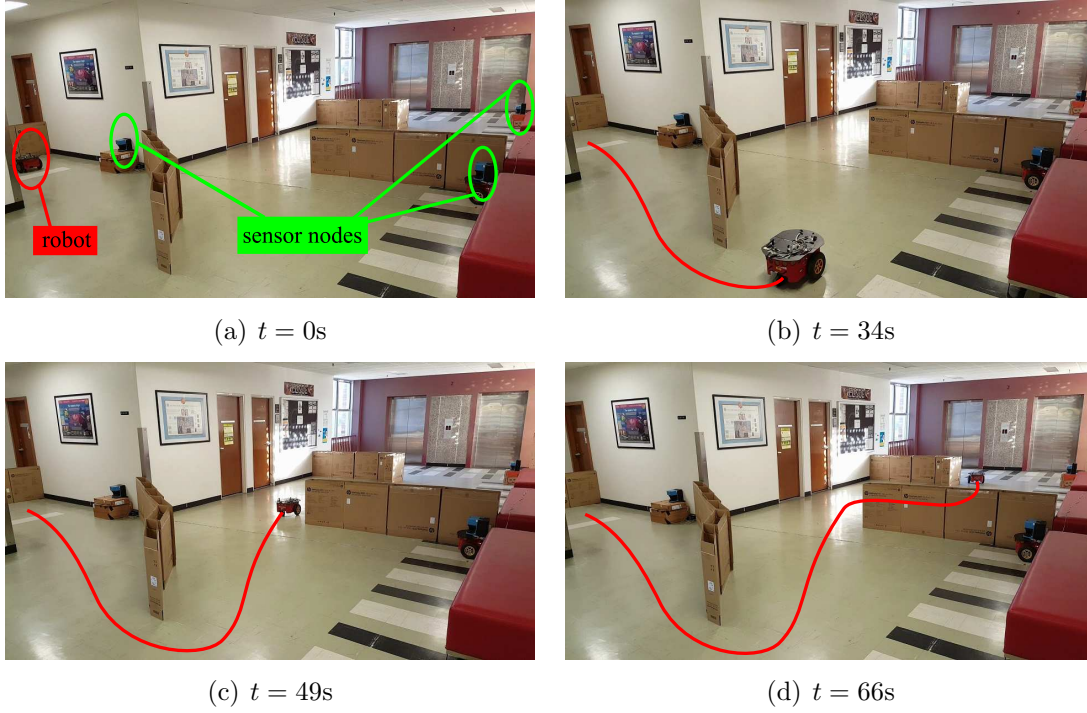
Figure 3.31: Pictures of the first experiment.
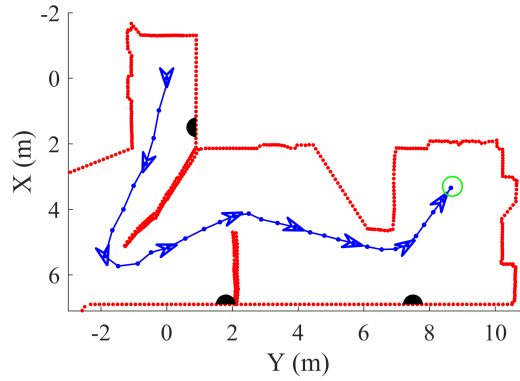
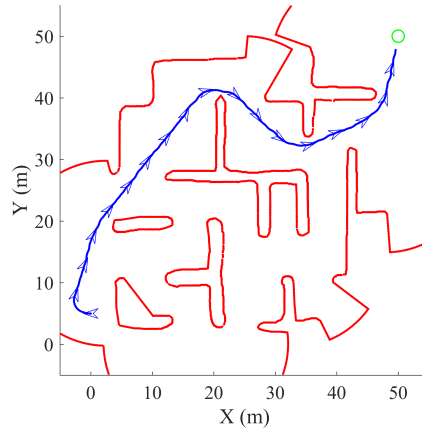Figure 3.32: Experimental result of the first experiment. The green circle is the target. The blue line is the trajectory of the robot.
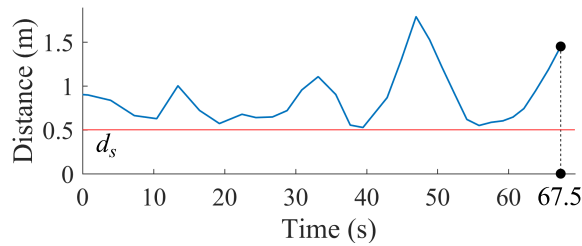
Figure 3.33: Minimum distance to the undetected areas in the first experiment.

3.4.2   Experiments in dynamic environments

In the third experiment (see Fig. 3.37), we tested the proposed navigation algorithm in a dynamic environment. In this scene, some folding cartons were arranged as two

(a) $t = 0$s

(b) $t = 14$s

(c) $t = 30$s

(d) $t = 42$s
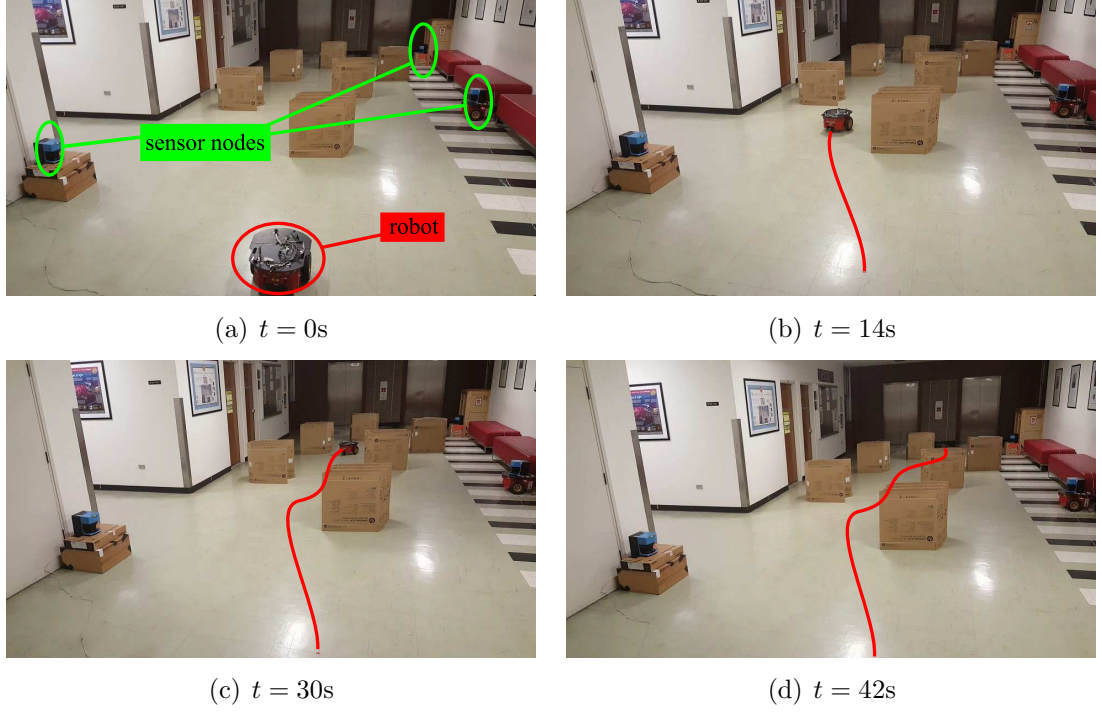
Figure 3.34: Pictures of the second experiment.
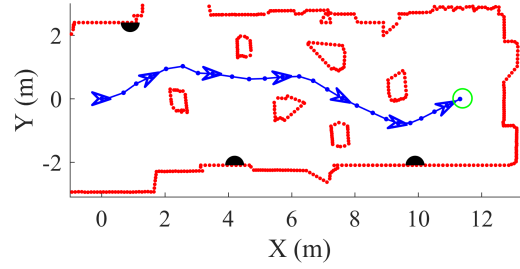
Figure 3.35: Experimental result of the second experiment. The green circle is the target. The blue line is the trajectory of the robot.
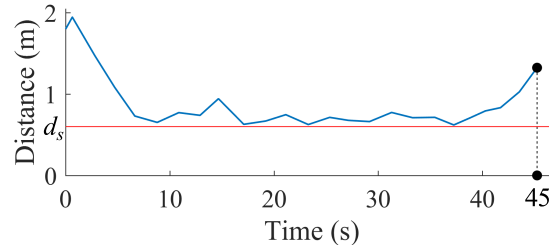
Figure 3.36: Minimum distance to the undetected areas in the second experiment.

static obstacles and two volunteers were walking in this scene. The volunteers' speed were smaller than the given maximum speed $V_{\max}$, which is 0.4m/s. The parameter T is determined as 2. Other parameters for this experiment are indicated in Table 3.3. According to the experimental result in Fig. 3.38, it can be seen that the

sensor network built the real-time map of the dynamic environment. The mobile robot successfully avoided both the static obstacles and the moving obstacles. Fig. 3.39 shows that the mobile robot was keeping the safety margin d_s while travelling. The control input satisfies the bound constraint.

Table 3.3: Main Parameters for the Third Experiment

Measurement range	R_s	8m
Speed of robot	v_r	0.5m/s
Maximum angular velocity	u_M	2rad/s
Safety margin	d_s	0.5m
Sampling interval	δ	1s
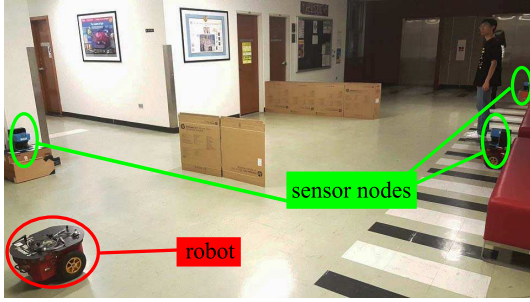
(a) $t = 0$s
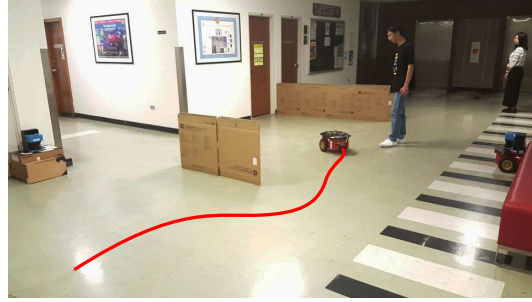
(b) $t = 23$s
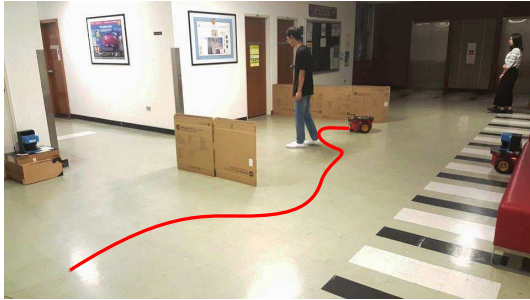
(c) $t = 30$s
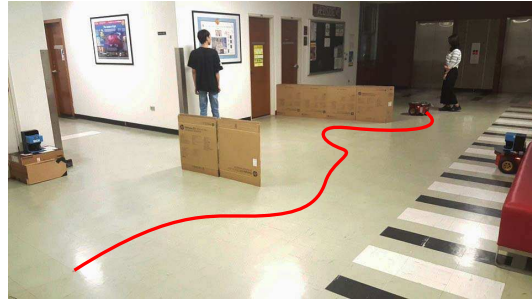
(d) $t = 45$s
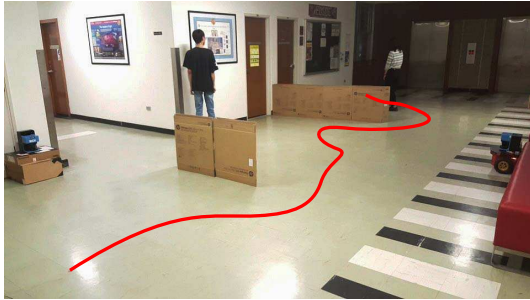
(e) $t = 49$s
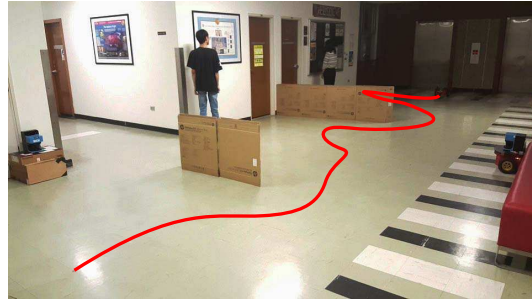
(f) $t = 54$s

Figure 3.37: Pictures of the third experiment.

(a) $t = 11$s

(b) $t = 23$s

(c) $t = 30$s

(d) $t = 43$s

(e) $t = 48$s

(f) $t = 54$s
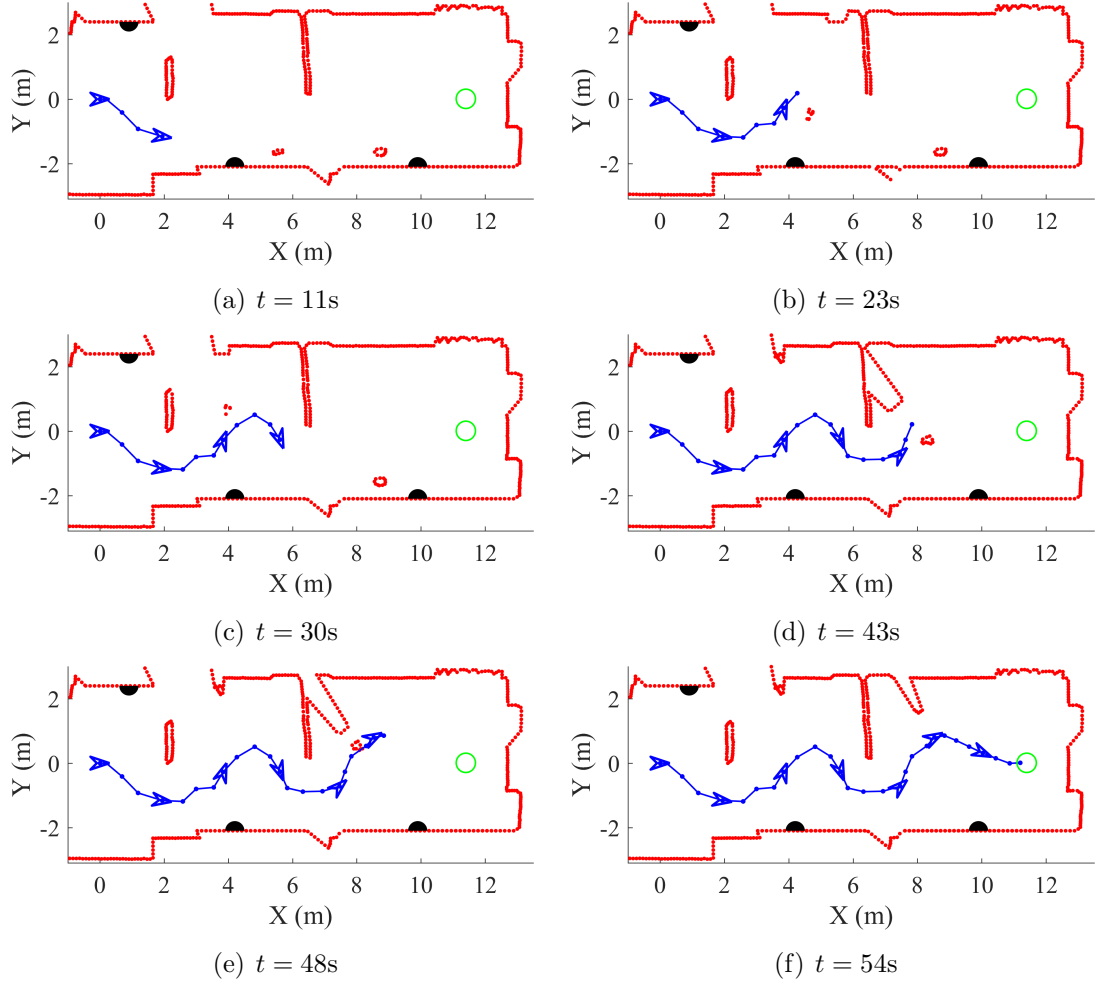
Figure 3.38: Experimental result of the third experiment. The green circle is the target. The blue line is the trajectory of the robot.
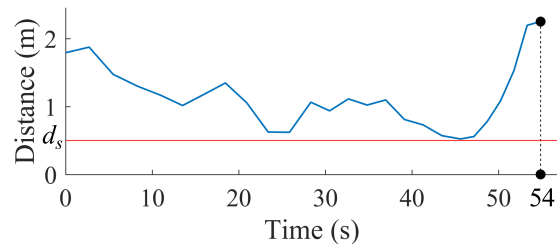
Figure 3.39: Minimum distance to the undetected areas in the third experiment.

3.5 Summary

In this chapter, we take the advantage of the range finder sensor network and propose a sensor network based navigation algorithm for any types of mobile robots working in the environments. Comparing with other types of sensor network, the range finder sensor network can obtain accuracy environment maps to help perform a faster and

more reliable navigation than RSSI-based sensor networks. Moreover, It involves simpler data processing than visual sensor networks.

The proposed sensor network is deployed in dynamic environments to detect the static and dynamic obstacles. Simultaneously, each robot measures and uploads its own location and direction by the wireless communication. Then, the sensor network navigate each mobile robot respectively according to the environment and robots information.

The main advantage of the proposed method is that the navigation tasks for any mobile robots are integrated into the sensor network. The sensor network can be used to navigate different types of robots simultaneously. The proposed navigation method does not require any robot sensor for obstacle detection and any other extra navigation algorithm. Moreover, the proposed method is flexible in configuration for multiple robots. Each robot is only required for a low-level path tracking controller. Therefore, this is an efficient, integrated and economic navigation system for multiple robots in practical dynamic environments.

The proposed navigation framework can be implemented as one of the most fundamental units with various robots working cooperatively. With the arranged sensor network, manufacturers can purchase any types of robots without considering the navigation and obstacle avoidance. This system also provides a centralized framework to manage and supervise the working of all the robots. The sensor network collects each robot's status and sensor measurements, such as battery, system errors and radiant intensity alarm, and sends the information to the manager or high-level management system. On the other hand, the manufacturers of mobile robots can remove the obstacle detection sensors and navigation algorithms from the robots to decrease the robots' cost and price.

Chapter 4

Safe Navigation of Flying Robots in Dynamic Environments by a Sensor Network

This chapter is based on the the publications [84] and [85]. In this chapter, we extend the path planning algorithm presented in Chapter 2 and 3 into 3D environment and combine it with a probabilistic roadmap to perform a real-time navigation for micro flying robot. In this chapter, we take the advantages of the wireless sensor network to navigate the flying robots. The sensor network consists of some 3D range finder sensors, such as time-of-flight cameras. Each sensor node is deployed in the dynamic environments, such as floors and walls, to detect the obstacles like walls, equipments and walking people. Simultaneously, each flying robot measures the real-time location and direction by localization technology, like odometry, and sends the measurements to the central computer via the wireless sensor network. With the measurements of the environment and each robot's position, instant safe paths are dynamically generated from each robot's current position to the destinations by the central computer. The instant safe paths are updated at each time step and the flying robots keep tracking the generated path to the targets. The presented flying robot navigation framework is a networked control system that the sensor

data, control input and flying robots' states are exchanged via the sensor network; see examples [103, 104, 150, 105].

4.1   Problem description

In this chapter, we consider any micro flying robot in the IIoT as a three-dimensional under-actuated non-holonomic autonomous vehicle, which is widely used to describe 3D robots and missile; e.g. [188, 101, 100, 126, 61, 127]. For different types of real flying robots (e.g. quadcopter and fixed-wing aircraft), a suitable controller can be designed such that the real flying robot can move like this model. In [92], such a controller is designed and implemented with a real robot ARDrone for 3D Dubins curve tracking. Moreover, a navigation algorithm designed for this kinematic model also has been successfully implemented with real flying robots in [188, 189]. Furthermore, the 3D kinematic model for flying robots studied in this chapter is a natural extension of 2D kinematic model for planar robots. Controllers designed based on the 2D kinematic model were successfully implemented with Pioneer type ground robots, wheelchairs and autonomous hospital beds, see e.g. [106, 187, 163]. The flying robot is equipped with the inertial measurement unit (IMU) and a wireless communication device. The mathematical model of the flying robot is as follows. Let

$$\boldsymbol{s}(t) := [x(t), y(t), z(t)] \tag{4.1}$$

be the Cartesian coordinates of the robot in the 3D space. For multiple flying robots, let $\boldsymbol{s}_i$ denote the i-th robot's position. Then, the motion of the robot can be described by the following equations (see Fig. 4.1):

$$\begin{cases} \dot{\boldsymbol{s}} = v_r \vec{\boldsymbol{i}} \\ \dot{\vec{\boldsymbol{i}}} = \vec{\boldsymbol{u}} \end{cases}, \tag{4.2}$$

where v_r is a constant speed, $\vec{\boldsymbol{i}} \in \mathbb{R}^3$ is the unit vector indicating the direction of the robot's velocity. $\vec{\boldsymbol{u}} \in \mathbb{R}^3$ is a two degree of freedom control input, which is subject to

$$
\begin{cases}
\|\vec{\boldsymbol{u}}\| \leq U_M \\
\vec{\boldsymbol{i}} \perp \vec{\boldsymbol{u}}
\end{cases}.
\tag{4.3}
$$

Here $\perp$ denotes that two vectors are orthogonal. The robot's location $\boldsymbol{s}(t)$ and the direction of velocity $\vec{\boldsymbol{i}}(t)$ can be obtained by the robot's IMU and odometry. The constraint of the control input $\vec{\boldsymbol{u}}$ implies that the minimum turning radius of the flying robot is

$$
R_{\min} = \frac{v_r}{U_M}.
\tag{4.4}
$$
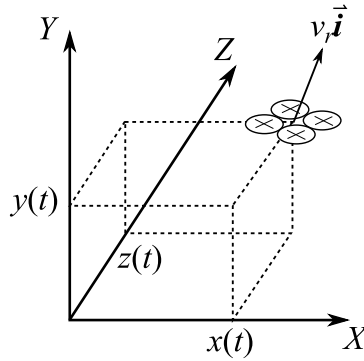
Figure 4.1: Flying robot model.

Notation 4.1.1 *Let p be an arbitrary point. For any closed set D, the minimum distance $\rho(D, p)$ between p and D is*

$$
\rho(D, p) := \min_{q \in D} \|p - q\|.
\tag{4.5}
$$

Considering the minimum turning radius, we define a torus called initial torus, which is a extension of the initial circle proposed in [157]; also see [96].

Definition 4.1.1 *Let B be a circle described as follows:*

$$B = \{p \in \mathbb{R}^3 : (p - \boldsymbol{s}) \perp \vec{\boldsymbol{i}}, \|p - \boldsymbol{s}\| = R_{min}\}. \tag{4.6}$$

Then, an initial torus $\mathcal{Q}$ can be defined as follows (see Fig. 4.2):

$$\mathcal{Q} := \{p \in \mathbb{R}^3 : \rho(B, p) = R_{min}\}. \tag{4.7}$$
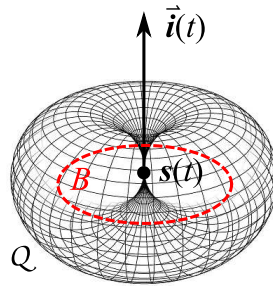
Figure 4.2: Initial torus.

In an practical environment, there are static and moving obstacles (e.g. walls, walking people, etc.) with irregular shape and dynamic velocity. To detect the obstacles extensively, we consider a WSN deployed in the workspace (see Fig. 4.3). Each sensor node is a 3D range finder, which measures the distances to the nearest obstacles in different directions within the field of view (see Fig. 4.4). There are two types of 3D range finder that are commonly used in many applications. The first type is spinning 2D range finder, see [123], which provides omnidirectional measurements of distance. The second type is ToF camera that only measures the distance in a limited field of view like a normal camera; such as [5]. However, a ToF camera has a better performance in dynamic environments than a spinning 2D range finder. Both of these two types of 3D range finder can be used in the proposed method.

Assumption 4.1.1 *For any sensor node, the location and attitude, in which it is deployed, are static and known.*

Generally, in an environment, the sensor nodes are fixed on the walls, ceilings
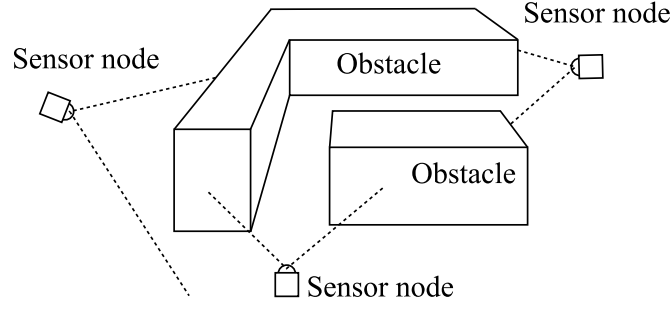
Figure 4.3: A 3D range finder WSN in the environment.
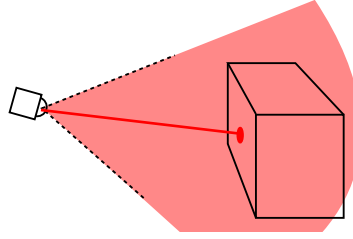
Figure 4.4: A 3D range finder.

or ground. The RSSI is commonly available for current WSN and flying robots. Therefore, the locations of both the sensor nodes and the flying robots can be estimated based on RSSI; see examples [171, 144]. Moreover, magnetic sensor and gravity sensor can be simply used on sensor nodes to obtain the attitude of each sensor node. In the presented method, a central computer is involved to execute the proposed navigate algorithm for the micro flying robots. It connects to one of the sensor nodes and accesses all the sensor nodes via the WSN to collect the real-time measurements of the obstacles from each 3D range finder. Then, the central computer combine the measurements to build a 3D map of the unoccupied area. In addition, the flying robots in the workspace can connect to the WSN. Thus, the central computer can communicate with any flying robot via the WSN as well. In practical implementations in environments, the micro flying robots can be equipped with some light sensors for some tasks and upload the sensor data to the central computer. For instance, radiation detectors can be mounted on the flying robots to monitor the radiant intensity in nuclear power stations. The framework of this navigation system is shown in Fig. 4.5.

In the environment, the obstacles and flying robots can be detected by the sensor
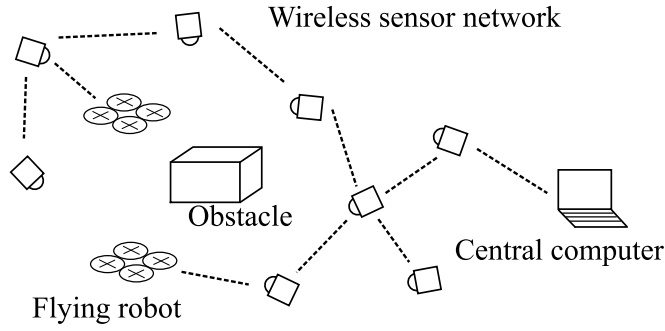
Figure 4.5: Framework of the proposed method.

nodes. For any i and time t, the i-th sensor node detects objects partly in its field of view and obtains a 3D detected region, denoted by $\mathcal{N}_i(t)$. The region $\mathcal{N}_i(t)$ can be obtained by converting the sensor's measurements to the global coordinate system from its local coordinate system with the help of the its location and attitude (see Fig. 4.6).
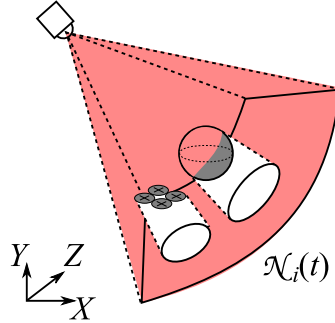
Figure 4.6: Detected region $\mathcal{N}_i(t)$ of the i-th sensor node.

Definition 4.1.2 *For any i and t, the detected region $\mathcal{N}_i(t)$ is a closed, bounded, and connected point set in $\mathbb{R}^3$.*

Definition 4.1.3 *Let $d > 0$ be a constant. Let ∂D denote the boundary of a closed set D. The d-reduction of the closed set D is a set $\mathcal{R}[D, d]$ defined as follows (see Fig. 4.7):*

$$\mathcal{R}[D, d] := \{p \in D : \rho(\partial D, p) \geq d\}. \tag{4.8}$$

Definition 4.1.4 *Let $d > 0$ be a constant. The d-enlargement of the closed set D*
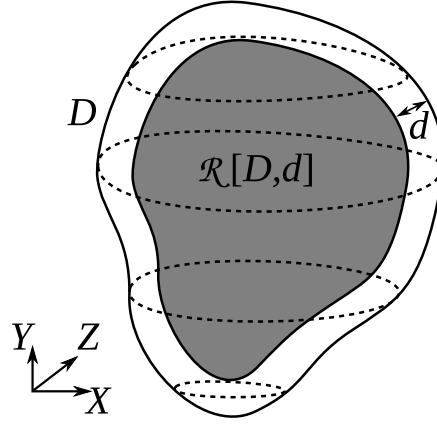
Figure 4.7: d-reduced region $\mathcal{R}[D, d]$ of a set D.

is a set $\mathcal{E}[D, d]$ defined as follows (see Fig. 4.8):

$$\mathcal{E}[D, d] := \{p \in \mathbb{R}^3 : \rho(D, p) \leq d\}. \tag{4.9}$$
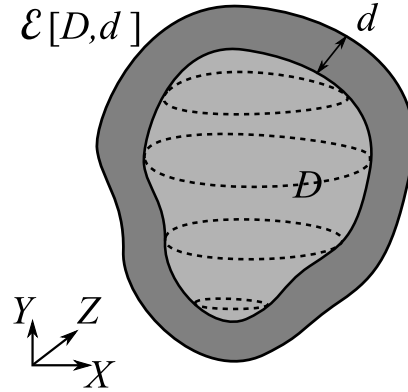
Figure 4.8: d-enlargement region $\mathcal{E}[D, d]$ of a set D.

Assumption 4.1.2 *The physical size of any flying robot in the workspace can be covered by a sphere with a given radius R_r on the center of the robot (see Fig. 4.9). The sphere is called robot sphere; i.e. the i-th robot sphere is the set $\mathcal{E}[\boldsymbol{s}_i(t), R_r]$.*

It is noticed that each micro flying robot's location is uploaded to the central computer. At the same time, it is also detected by the 3D range finder sensors. Therefore, we suppose that error correction for the robot's odometry is applied. There are a lot of methods in the error correction of odometry by 3D range finder, such as [50].
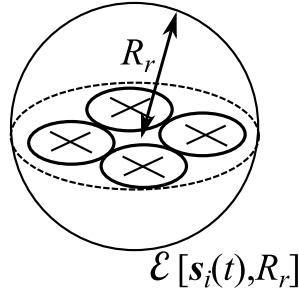
Figure 4.9: Physical size of a flying robot.

Moreover, because the location of each robot is known and the size of each robot is micro, a filter should be involved to remove the effect of any robots on each 3D region $\mathcal{N}_i(t)$, $i = 1, 2, \ldots$; i.e. for the measurements of the i-th sensor node, the occupied positions belonging to any robots can be recognized by using the robots' positions and radius R_r, then a low-pass filter is applied on the depth measurements corresponding to the robots' occupied positions. Here, let $\hat{\mathcal{N}}_i(t)$ denote the estimated region of $\mathcal{N}_i(t)$ by the filter (see Fig. 4.10).
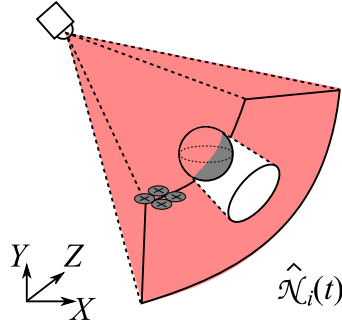
Figure 4.10: Estimated region $\hat{\mathcal{N}}_i(t)$ without the effect of micro flying robots for the i-th sensor node.

Now, an unoccupied area, denoted by $\mathcal{A}(t)$, can be calculated as follows (see Fig. 4.11):

$$\mathcal{A}(t) = \bigcap_{i=1}^{m} \hat{\mathcal{N}}_i(t), \tag{4.10}$$

where m is the number of the sensor nodes.

Assumption 4.1.3 *Let p be an arbitrary point in the set $\mathcal{A}(t)$ and $\partial \mathcal{A}(t)$ denote the boundary of $\mathcal{A}(t)$. The derivative of the minimum distance $\rho(\partial \mathcal{A}(t), p)$ with respect*
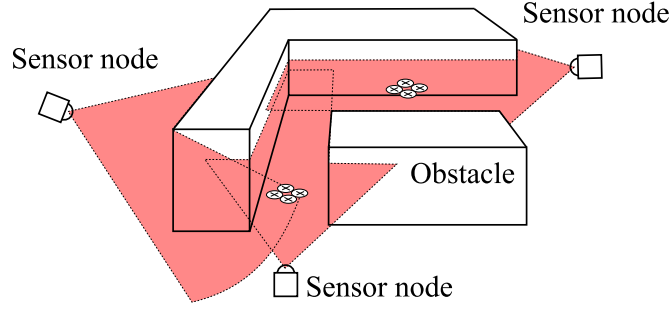
Figure 4.11: Unoccupied area $\mathcal{A}(t)$.

to time t is smaller than or equal to $V_{\max}$, where $V_{\max}$ is a given constant, which is smaller than v_r.

Remark 4.1.1 *If Assumption 4.1.3 does not hold, the robot may need to avoid the dynamic obstacles and undetected regions which are faster than the robot. It is obvious that in this case, it is impossible to obtain mathematically rigorous conditions for the robot to surely avoid collisions for any navigation strategy. However, there are still some algorithms that can be modified and combined to help the robot keep safe in some conditions of this case; see the algorithm and mathematical analysis in [163].*

Definition 4.1.5 *The safety margin $d_s > 0$ is a given constant that the flying robots should keep from the boundary $\partial \mathcal{A}(t)$ at any time t. It satisfies the following inequality:*

$$d_s \geq R_{min}. \tag{4.11}$$

Assumption 4.1.4 *The robot's initial position $\boldsymbol{s}(0)$ belongs to the set $\mathcal{R}[\mathcal{A}(0), d_s]$ and is far away from the boundary of $\mathcal{A}(0)$.*

Assumption 4.1.5 *The set $\mathcal{R}[\mathcal{A}(t), d_s]$ is a connected set at any time t.*

The objective of the proposed algorithm is to drive a micro flying robots to travel in the dynamic and deformable d_s-reduced region $\mathcal{R}[\mathcal{A}(t), d_s]$, avoid any other flying robot and reach a target, which is denoted by $\mathcal{T}$, with a relatively short trajectory. In the presented method, the environment measurements, robot's position and heading,

and robots' control signals velocity are exchanged through the sensor network to construct a networked control system; see examples [103, 104, 150, 105].

Assumption 4.1.6 *The target $\mathcal{T}$ belongs to the set $\mathcal{R}[\mathcal{A}(t), d_s]$ at any time t.*

Assumption 4.1.7 *The target $\mathcal{T}$ is far away from the robot's initial position $\boldsymbol{s}(0)$.*

4.2   Safe navigation algorithm

In this section, we propose a discrete-time navigation algorithm for the micro flying robots in the workspace based on the presented WSN. Each flying robot tracks a relatively short and temporarily safe path denoted by P^*, which is generated by the central computer according to the proposed navigation algorithm.

The path P^* is represented as some successive and finite equally spaced points denoted by $P^*(0), P^*(1), \ldots, P^*(n)$ (see Fig. 4.12), where n is the length of the path. $P^*(0)$ is the flying robot's current position $\boldsymbol{s}(t)$. For any k, $P^*(k)$ represents the possible position of the robot at the future time $t+k\delta$. The distance between any two successive points is a constant L, which is equal to $v_r\delta$. P^* should approximately satisfy the non-holonomic constraint of the flying robots; i.e. the radius of the circumscribed circle of any three successive points should be greater than or equal to $R_{\min}$ and the point $P^*(1)$ should not be within the initial torus $\mathcal{Q}$.

In the proposed method, we consider two conditions of a micro flying robot. In the first condition, a flying robot is steady at its current position before a target is arranged or after it reach a target. In this case, P^* has only one point, i.e. $n = 0$. In the second condition, the flying robot is travelling to an arranged target with a generated path P^*, i.e. $n > 0$.

Let δ be the sampling interval. At each time step $t = 0, \delta, 2\delta, \ldots$, the central computer calculates the unoccupied area $\mathcal{A}(t)$ and obtains the flying robots' real-time positions $\boldsymbol{s}_1(t), \boldsymbol{s}_2(t), \ldots$ via the WSN. Then, at each time step, the paths $P_1^*, P_2^*, \ldots$ for all the robots are updated successively according to the real-time unoccupied area and the robots' positions.
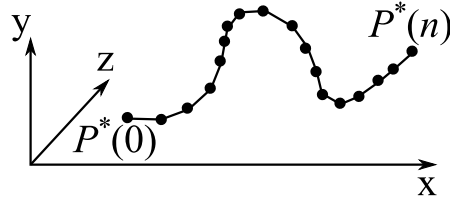
Figure 4.12: A 3D robot path represented as some equally spaced points.

Let T be a given time window ,which is a positive integer. The generated path P^* has the following properties. Firstly, P^* is a path from the robot's current position $\boldsymbol{s}(t)$ to the target $\mathcal{T}$. Secondly, P^* satisfies the non-holonomic constraint of the robot's motion. Thirdly, P^* is guaranteed to be collision-free to the obstacles and other robots over the time period $[t, t + T\delta]$. Because P^* is updated at each time step with interval δ, the robot finally can reach the target successful without any collisions. If P^* is updated with a time delay which is smaller than $T\delta$, the robot is still safe in tracking P^*.

To guarantee the safety of each robot in the partly detected dynamic environment, we consider any potentially unsafe regions excluding robot spheres in the environment and give the following definitions and assumptions.

Definition 4.2.1 *Let $k = 0, 1, 2, \ldots$ and t be the current time. Over the time window T, the region which is obsoletely safe at time $t + k\delta$, $k \leq T$ is*

$$\bar{\mathcal{A}}(t, k) := \mathcal{R}[\mathcal{A}(t), k\delta V_{\max}]. \tag{4.12}$$

Then we extend it as follows to help to generate a temporarily safe path in the time window T:

$$\bar{\mathcal{A}}(t, k) := \mathcal{R}[\mathcal{A}(t), T\delta V_{\max}] \quad , k > T. \tag{4.13}$$

Definition 4.2.2 *Let $k = 0, 1, 2, \ldots$ and t be the current time. For the i-th micro flying robot, considering the collisions with other robots, a valid area $\hat{\mathcal{A}}(t, k)$ at any*

future time $t + k\delta$ is defined as follows:

$$\hat{\mathcal{A}}(t,k) := \bar{\mathcal{A}}(t,k) \setminus \bigcup_{j \neq i} \mathcal{E}[P_j^*(k), R_r]. \tag{4.14}$$

Here we extend any robot's path P_j^* as follows

$$P_j^*(k) := P_j^*(n) \quad , k > n \tag{4.15}$$

and $\setminus$ denotes the set difference.

Assumption 4.2.1 *For any t and time step k, the target point $\mathcal{T}$ belongs to the set $\mathcal{R}[\hat{\mathcal{A}}(t,k), d_s]$.*

Assumption 4.2.2 *For any t and k, the set $\mathcal{R}[\hat{\mathcal{A}}(t,k), d_s]$ is a connected set.*

Definition 4.2.3 *The path P^* is called a temporarily safe path, if for any k and t, the point $P^*(k)$ belongs to the set $\mathcal{R}[\hat{\mathcal{A}}(t,k), d_s]$.*

Definition 4.2.4 *The path P^* is called a target-reaching path if $\|\mathcal{T} - P^*(n)\| \leq L$.*

Now, we are here to briefly introduce the two parts of the proposed navigation algorithm. The first part of the algorithm is rough path generation. It is followed by the second part, which is path planning. At each time step, for any flying robot, if a target is arranged and the robot needs to be navigated to the target, an initial rough path $\bar{P}$ need to be generated firstly by the probabilistic roadmap algorithm. Then the generated rough path $\bar{P}$ is adjusted to be the path P^* by the proposed path planning algorithm. With the presented algorithm, any micro flying robot navigated by the WSN can avoid any obstacles and other flying robots in the workspace and reach the target successfully.

4.2.1   Rough path generation

To initialize a temporarily safe rough path for a micro flying robot with an arranged target $\mathcal{T}$, the probabilistic roadmap method is used to generate a relatively short

rough path $\bar{P}$ in the area $\mathcal{R}[\mathcal{A}(t), d_s + T\delta V_{\max}]$. The probabilistic roadmap method proposed in [69] is widely used in robotics to solve the motion planning problem; e.g. [145].

In the probabilistic roadmap method, a graph $\mathcal{G}$ is construct to search the shortest path from a starting point to a target point in the graph. The vertices of the graph $\mathcal{G}$ are the random samples of the configuration space of the robot, the starting point and the target point. Each edge of the graph $\mathcal{G}$ represents a valid path connecting the corresponding pair of vertices. Then, a shortest path search algorithm is used to find the shortest path from the starting point to the target point in the graph $\mathcal{G}$. In our method, the vertices of $\mathcal{G}$, excluding the starting point and the target point, are randomly taken from the area $\mathcal{R}[\mathcal{A}(t), d_s + T\delta V_{\max}]$ with uniform distribution. Then, we give a definition and the algorithm description as follows.

Definition 4.2.5 *Let D be a closed point set. A line segment path between any two points a, b ∈ D is called a valid path if the line segment between a and b does not intersect with the boundary of D (see Fig. 4.13).*
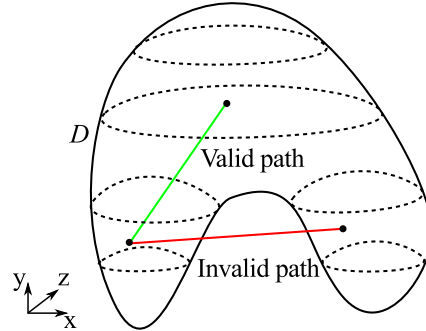
Figure 4.13: A valid path between two points.

Now, we are here to give the rough path generation algorithm as follows (see Fig. 4.14):

A1: Initialize a graph $\mathcal{G} = (V, E)$, where the set of vertices V and the set of edges E are empty sets.

A2: Let $N_s > 0$ be a given integer. Then, select N_s points $\{\xi_1, \xi_2, \ldots, \xi_{N_s}\}$ randomly from the point set $\mathcal{R}[\mathcal{A}(t), d_s + T\delta V_{\max}]$ with uniform distribution. Then,

add these selected points into the set V as follows:

$$V \leftarrow V \cup \{\xi_1, \xi_2, \ldots, \xi_{N_s}\}. \tag{4.16}$$

A3: Let $N_c > 0$ be a given integer that is smaller than N_s. For any point $\xi_i \in V$, $i = 1, 2, \ldots, N_s$, choose the N_c closest vertices, denoted by $\{\xi'_{i,1}, \xi'_{i,2}, \ldots, \xi'_{i,N_c}\}$, in V according to the distance. For each chosen vertex $\xi'_{i,j} \in V$, $j = 1, 2, \ldots, N_c$, if the line segment between ξ_i and $\xi'_{i,j}$ is a valid path in $\mathcal{R}[\mathcal{A}(t), d_s + T\delta V_{\max}]$, add the corresponding edge into the set E as follows:

$$E \leftarrow E \cup \{(\xi_i, \xi'_{i,j})\}. \tag{4.17}$$

A4: Let ξ_{init} be the flying robot's current position $\boldsymbol{s}(t)$ and add it into the set V as follows:

$$V \leftarrow V \cup \{\xi_{\text{init}}\}. \tag{4.18}$$

For the starting vertex $\xi_{\text{init}} \in V$, choose the N_c closest vertices, denoted by $\{\xi'_{\text{init},1}, \xi'_{\text{init},2}, \ldots, \xi'_{\text{init},N_c}\}$ in V according to the distance. Considering the constraint of the control input (4.3), for each chosen vertex $\xi'_{\text{init},j} \in V$, $j = 1, 2, \ldots, N_c$, if the line segment between ξ_{init} and $\xi'_{\text{init},j}$ is a valid path in $\mathcal{R}[\mathcal{A}(t), d_s + T\delta V_{\max}]$ and the following inequality holds:

$$\arccos\left(\frac{a(t)\zeta_j}{\|a(t)\|\|\zeta_j\|}\right) \leq \arcsin\left(\frac{LU_M}{2v_r}\right), \tag{4.19}$$

where $\zeta_j = \xi'_{\text{init},j} - \xi_{\text{init}}$, then add the corresponding edge into the set E as follows:

$$E \leftarrow E \cup \{(\xi_{\text{init}}, \xi'_{\text{init},j})\}. \tag{4.20}$$

A5: Let ξ_{goal} be the target point $\mathcal{T}$ and add it into the set V as follows:

$$V \leftarrow V \cup \{\xi_{\text{goal}}\}. \tag{4.21}$$

For the goal vertex $\xi_{\text{goal}} \in V$, choose the N_c closest vertices, denoted by $\{\xi'_{\text{goal},1}, \xi'_{\text{goal},2}, \ldots, \xi'_{\text{goal},N_c}\}$ in V according to the distance. For each chosen vertex $\xi'_{\text{goal},j} \in V$, $j = 1, 2, \ldots, N_c$, if the line segment between ξ_{goal} and $\xi'_{\text{goal},j}$ is a valid path in $\mathcal{R}[\mathcal{A}(t), d_s + T\delta V_{\max}]$, then add the corresponding edge into the set E as follows:

$$E \leftarrow E \cup \{(\xi_{\text{goal}}, \xi'_{\text{goal},j})\}. \tag{4.22}$$

A6: Search the shortest path from the starting vertex ξ_{init} to the goal vertex ξ_{goal} in the graph $\mathcal{G}$. Generally, Dijkstra's algorithm can be used to search the shortest path in a graph.

A7: Convert the selected shortest path to the rough path $\bar{P}$, which consists of equally spaced points with interval L.
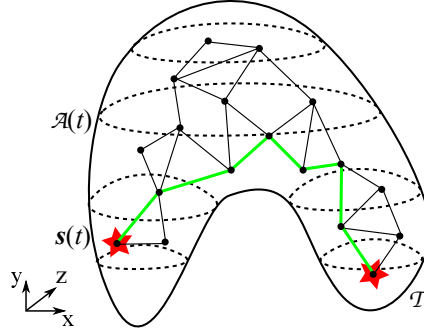
Figure 4.14: Rough path generation by probabilistic roadmap method. The black lines are the graph and the green line is the shortest path in the graph.

4.2.2 Path planing

Now, we are here to propose the path planning algorithm that adjusts a given flying robot path to be an temporarily safe path P^*, which is relatively short in the valid area $\mathcal{R}[\hat{\mathcal{A}}(t, k), d_s]$. Let P denote the given path that needs to be adjusted. It can be the initialized rough path $\bar{P}$ or the last temporarily safe P^* at the last time step.

Here, let $p_0, p_1, \ldots, p_n$ denote the path points of the given path P. Then, we define four vector fields $\vec{\boldsymbol{F}}_I$, $\vec{\boldsymbol{F}}_R$, $\vec{\boldsymbol{F}}_P$ and $\vec{\boldsymbol{F}}_C$ in $\mathbb{R}^3$ to help adjust the path points

of P. For any $k \neq 0$, the resultant vector of these four fields at point p_k is

$$\vec{\boldsymbol{F}}(p_k) = \vec{\boldsymbol{F}}_I(p_k) + \vec{\boldsymbol{F}}_R(p_k) + \vec{\boldsymbol{F}}_P(p_k) + \vec{\boldsymbol{F}}_C(p_k). \tag{4.23}$$

The path P is adjusted by moving any p_k, $k \neq 0$ towards the direction of $\vec{\boldsymbol{F}}(p_k)$ to an equilibrium point where $\vec{\boldsymbol{F}}(p_k) = \vec{\boldsymbol{0}}$. Moreover, while adjusting, some points would be added or removed to prolong or shorten the length of P until $\vec{\boldsymbol{F}}(p_k) = \vec{\boldsymbol{0}}$ for any $k \neq 0$. Now, we are here to give the definitions of four vector fields.

Definition of $\tilde{\boldsymbol{F}}_{\mathbf{I}}$: $\vec{\boldsymbol{F}}_I$ is a vector field which guarantees that the interval between any two successive points of P is approximately equal to L. For any $k \neq 0$, let $\vec{l}_k$ be the vector from p_k to p_{k-1} (see Fig. 4.15), then $\vec{\boldsymbol{F}}_I(p_k)$ is defined as follows:

$$\vec{\boldsymbol{F}}_I(p_k) := \begin{cases} G_I(b(k)\vec{l}_k - b(k+1)\vec{l}_{k+1}) & \text{if } k \neq n \\ G_I(b(k)\vec{l}_k) & \text{if } k = n \end{cases}, \tag{4.24}$$

where $G_I > 0$ is a tunable gain and $b(k)$ is

$$b(k) := 1 - \frac{L}{\|\vec{l}_k\|}. \tag{4.25}$$
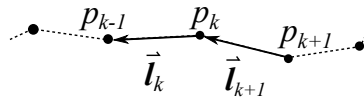
Figure 4.15: Vector $\vec{l}_k$.

Definition of $\tilde{\boldsymbol{F}}_{\mathbf{R}}$: $\vec{\boldsymbol{F}}_R$ is a vector field which guarantees that the minimum distance from the point p_k to the boundary of the valid area $\hat{\mathcal{A}}(t, k)$ is greater than or equal to d_s for any $k \neq 0$. Let $\vec{r}_k$ be the shortest vector from p_k to the boundary of $\hat{\mathcal{A}}(t, k)$, then $\vec{\boldsymbol{F}}_R(p_k)$ is defined as follows (see Fig. 4.16):

$$\vec{\boldsymbol{F}}_R(p_k) := \begin{cases} G_R(1 - \frac{d_s}{\|\vec{r}_k\|})\vec{r}_k & \text{if } \|\vec{r}_k\| \leq d_s \\ \vec{\boldsymbol{0}} & \text{if } \|\vec{r}_k\| > d_s \end{cases}, \tag{4.26}$$

where $G_R > 0$ is a tunable gain.
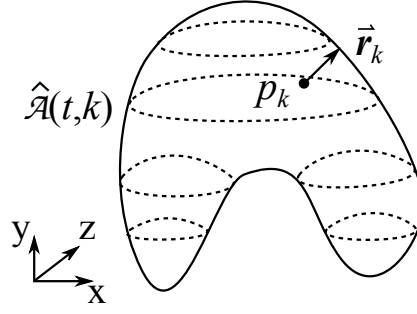
Figure 4.16: Vector $\vec{r}_k$.

Definition of $\tilde{F}_{\mathbf{P}}$: $\vec{F}_P$ is a vector field pointing towards the target $\mathcal{T}$, which only attracts the last point p_n of the path P. For any other $k \neq n$, $\vec{F}_P(p_k) = \vec{0}$. For the point p_n, let $\vec{g}$ be a unit vector pointing towards $\mathcal{T}$ from p_n (see Fig. 4.17), then $\vec{F}_P(p_n)$ is defined as follows:

$$\vec{F}_P(p_n) := G_P\vec{g}, \tag{4.27}$$

where $G_P > 0$ is a tunable gain.
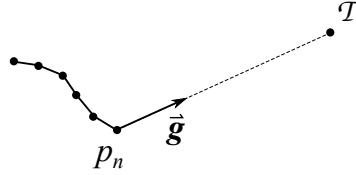
Figure 4.17: Direction vector $\vec{g}$.

Definition of $\tilde{F}_{\mathbf{C}}$: $\vec{F}_C$ is a vector field which guarantees that the path satisfies the non-holonomic constraint and does not intersect with the initial torus $\mathcal{Q}$ at the beginning. For any $k \neq 0$, let $\vec{h}_k$ be the vector from the point p_k to the closest point at the circle B, then $\vec{F}_C(p_k)$ is defined as follows:

$$\vec{F}_C(p_k) := \begin{cases} G_C(1 - \frac{R_{\min}}{\|\vec{h}_k\|})\vec{h}_k & \text{if } \|\vec{h}_k\| \leq R_{\min} \\ \vec{0} & \text{if } \|\vec{h}_k\| > R_{\min} \end{cases}, \tag{4.28}$$

where $G_C > 0$ is a tunable gain.

Now, we are here to propose the path planing algorithm as follows to adjust the

given path P:

B1: For any p_k, $k \neq 0$, initialize a velocity vector $\vec{v}_k = \vec{0}$ for p_k.

B2: Start the following loop:

    B2.1: Change the position vector and velocity vector of each p_k, $k \neq 0$ as follows:

$$
\begin{cases}
p_k \leftarrow p_k + \vec{v}_k \\
\vec{v}_k \leftarrow G_N \vec{v}_k + \vec{F}(p_k)
\end{cases}, \qquad (4.29)
$$

    where $0 < G_N < 1$ is a tunable attenuation.

    B2.2: If $\|p_n - \mathcal{T}\| < \underline{L}$, remove p_n from P and $n \leftarrow n - 1$. If $\|p_n - \mathcal{T}\| > \overline{L}$, add a new point next to p_n into P and $n \leftarrow n + 1$. The parameters $\underline{L} < L$ and $\overline{L} > L$ are two given thresholds.

B3: Exit loop **B2** if the inequality $\|\vec{F}(p_k)\| < F_{th}$ holds for any $k \neq 0$, where $F_{th} > 0$ is a given threshold.

In the proposed path planning algorithm, the appropriate attenuation G_N should be chosen such that each p_k can converge fast to the position where $\|\vec{F}(p_k)\| < F_{th}$ with small error between p_k and equilibrium point. With the proposed algorithm, the instant safe path P^* can be obtained for current time step. It approaches to a local optimum which has the minimum length and energy in the potential fields within the neighbourhood of the state space satisfying the safety criteria defined in Definition 4.2.3, 4.2.4 and the non-holonomic constraint (4.3) (see Fig. 4.18). The increase of G_I, G_R and G_C or decrease of G_P can decrease the error involved by discretization but may increase the time consumption of the algorithm or make the algorithm unstable. Therefore, the gains need to be selected according to different implementations.

(a) The given path P (b) The temporarily safe path P^*
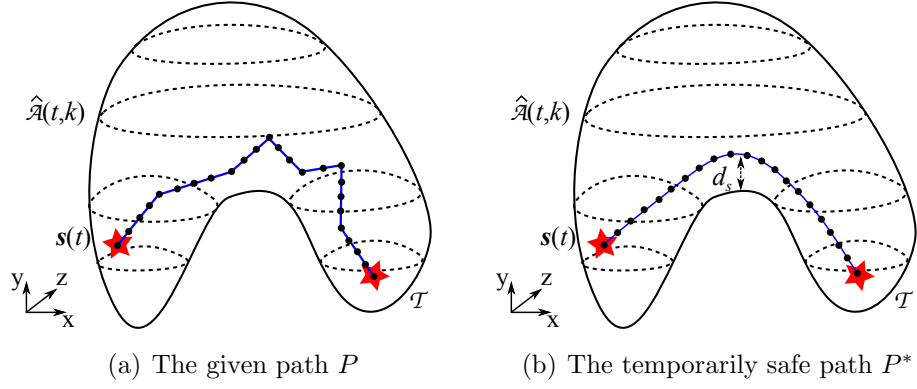
Figure 4.18: Adjustment of the given path P.

4.3 Computer simulations

In this section, computer simulations are carried out to confirm the performance of the proposed navigation algorithm in an practical environment. Moreover, a path tracking controller for the micro flying robot is proposed firstly.

4.3.1 Path tracking controller

Firstly, to drive the flying robot to track the generated path P^* during each sampling interval, we are here to propose a simple control strategy for flying robot path tracking. The proposed strategy is a modification of the control law which is proposed in [107].

In 3D flying robot path tracking, there are two errors which should be minimized. The first error denoted by $e_d(t)$ is the minimum distance from the robot's real-time position $s(t)$ to the generated path. The second error denoted by $e_a(t)$ is the angle between the robot's velocity direction vector $\vec{i}(t)$ and the plane $\Omega(t)$, which is determined by the robot's position $s(t)$ and the tangent line of the path at the closest point to $s(t)$ (see Fig. 4.19).

Notation 4.3.1 *The inner product of two vectors is denoted by* $\bullet$ *and the cross product of two vectors is denoted by* $\times$.

In the proposed control strategy, two sliding mode controllers are used to minimize the distance error $e_d(t)$ and the orientation error $e_a(t)$ respectively. Let $\vec{\boldsymbol{\nu}}_a(t)$
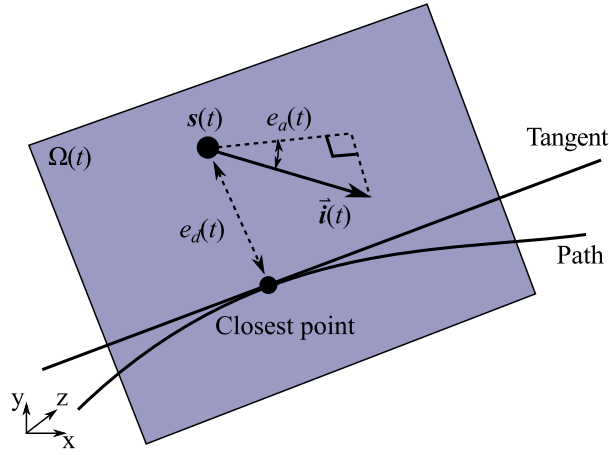
Figure 4.19: Minimum distance $e_d(t)$ and the angle $e_a(t)$.

be a normal vector of the plane $\Omega(t)$. The orientation error $e_a(t)$ can be obtained by the following equation:

$$e_a(t) = \arccos\left(\frac{\vec{i}(t) \bullet \vec{\nu}_a(t)}{\|s(t)\|\|\vec{\nu}_a(t)\|}\right) - \frac{\pi}{2}. \tag{4.30}$$

Let two tunable constants $\lambda_d > 0$ and $\sigma_d > 0$ be the parameters of the controller for the distance error $e_d(t)$; let another two tunable constants $\lambda_a > 0$ and $\sigma_a > 0$ be the parameters of the controller for the orientation error $e_a(t)$. Then, the two controllers for the two errors are designed as follows:

$$\begin{cases} u_d(t) = \operatorname{sgn}[\dot{e}_d(t) + \mathcal{X}(e_d(t), \lambda_d, \sigma_d)] \\ u_a(t) = \operatorname{sgn}[\dot{e}_a(t) + \mathcal{X}(e_a(t), \lambda_a, \sigma_a)] \end{cases}. \tag{4.31}$$

The saturation function $\mathcal{X}(z, \lambda, \sigma)$ is

$$\mathcal{X}(z, \lambda, \sigma) := \begin{cases} \lambda z & \text{if } |z| \le \sigma \\ \lambda\sigma\operatorname{sgn}(z) & \text{if } |z| > \sigma \end{cases} \tag{4.32}$$

and the sign function $\operatorname{sgn}(x)$ is

$$\text{sgn}(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x < 0 \end{cases} .$$ (4.33)

Let $\vec{\nu}_d$ be the unit vector from the robot's current position $\boldsymbol{s}(t)$ towards the closest point on the path. To minimize the two errors with one control input, we consider the sum of the two controller's outputs $u_d(t)$ and $u_a(t)$ in different directions $\vec{\nu}_d$ and $\vec{\nu}_a$ with the tunable weights $w_d > 0$ and $w_a > 0$ as follows:

$$\vec{\boldsymbol{u}}_s(t) = w_d u_d(t)\vec{\nu}_d + w_a u_a(t)\vec{\nu}_a.$$ (4.34)

Moreover, considering the constraint of the robot's control input (4.3), the direction vector of the orthogonal projection of the vector $\vec{\boldsymbol{u}}_s(t)$ onto the plane, of which the normal vector is $\vec{\boldsymbol{i}}(t)$, is calculated as follows (see Fig. 4.20):

$$\vec{\boldsymbol{u}}'(t) = \vec{\boldsymbol{i}}(t) \times \frac{\vec{\boldsymbol{u}}_s(t)}{\|\vec{\boldsymbol{u}}_s(t)\|} \times \vec{\boldsymbol{i}}(t).$$ (4.35)

Finally, the control input of the flying robot can is obtained as follows:

$$\vec{\boldsymbol{u}}(t) = \vec{\boldsymbol{u}}'(t)U_M.$$ (4.36)
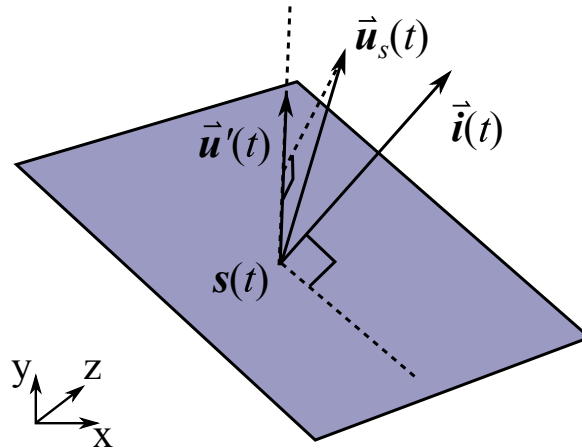
Figure 4.20: Control strategy of the flying robot path tracking.

To confirm the performance of the proposed path tracking controller, we carry out a computer simulation. In this simulation, a 3D robot path is given as a smooth curve from the robot's initial position to a target. The flying robot's positions $s(t)$ and velocity direction vector $\vec{i}$ are known to calculate the distance error $e_d(t)$ and the orientation error $e_a(t)$. The parameters used in this simulation are indicated in Table 4.1. The result of the simulation is shown in Fig. 4.21. It can be seen that the flying robot tracked the given path and reached the target. The distance error e_d and the orientation error e_a in this simulation are shown in Fig. 4.22, which indicates that the flying robot can be controlled accurately by the proposed path tracking controller. There are around 333 samples in this simulation.

Table 4.1: Parameters for the Path Tracking Simulation

Speed of robot	v_r	0.7m/s
Maximum angular velocity	u_M	2rad/s
Parameters of the controller	λ_d	2
	σ_d	1
	λ_a	3
	σ_a	1
	w_d	1
	w_a	1
Sampling interval	δ	0.2s

4.3.2 Time-of-flight camera

In our computer simulations, ToF camera are used to detect objects in the environments (see Fig. 4.23). It is a class of scannerless LIDAR by measuring the time-of-flight of a light signal between the camera and the objects for each pixel of the 2D image. It provides a depth image in the field of view with high frame rate, which perform a fast detection in dynamic environments. In the simulations, we use such type of 3D range finder sensors. In the simulation software, the provided ToF camera module is used. It provides the image with both colour and depth for each pixel. Therefore, it can be used as a normal optical camera for computer vision. In our simulation, only the distance information is considered. Thus, a grayscale image is enough to represent the measurements. Fig. 4.24 shows an example of
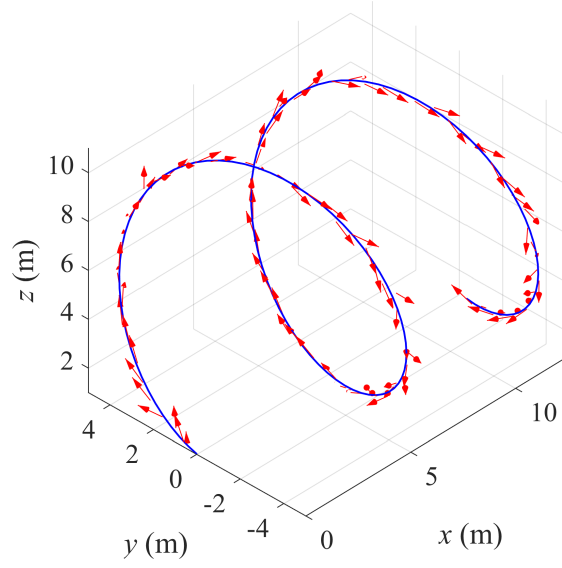
Figure 4.21: Result of the path tracking simulation. The blue solid line is the given 3D robot path and the red arrows are the trajectory and orientation of the flying robot.

(a)

(b)
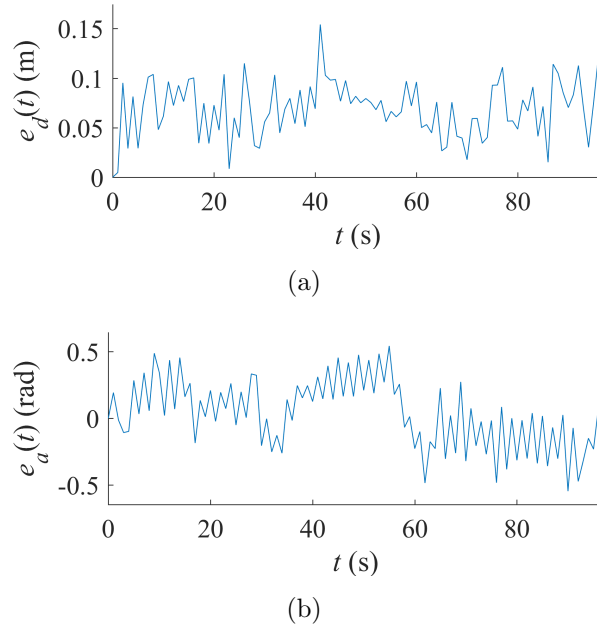
Figure 4.22: Distance error e_d and the orientation error e_a.

such a depth image, which is a grayscale image. According to the camera projection, the depth image can be converted to a point cloud to represent the objects in the environment (Fig. 4.25). The main parameters of the ToF camera used in the simulations are indicated in Table. 4.2.

Table 4.2: Parameters of the ToF Camera

Working range	0m to 9m
Lens	$135°h \times 135°v$
Resolution	128×128
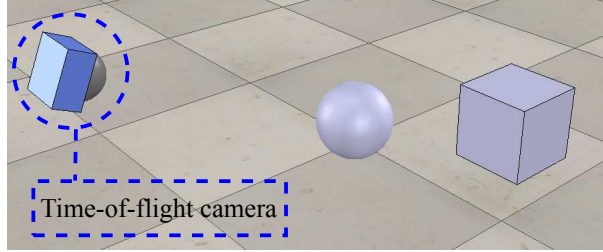
Figure 4.23: A ToF camera detects objects.
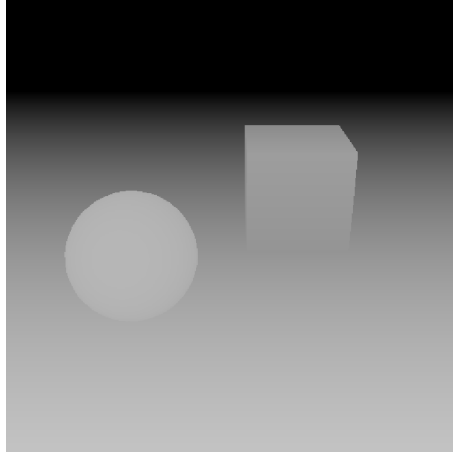
Figure 4.24: A depth image.
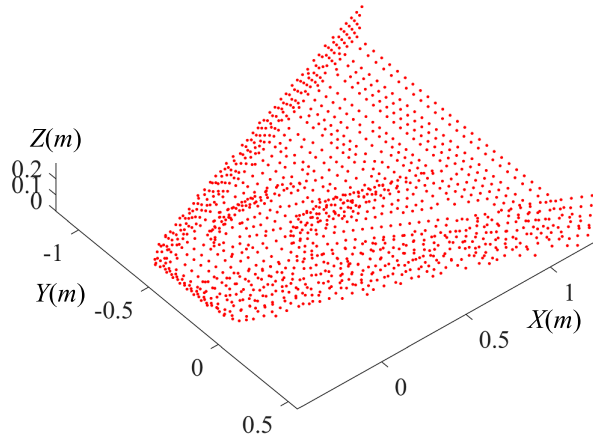
Figure 4.25: A point cloud of the ToF camera measurements.

4.3.3   Simulations in static environments

To confirm the performance of the proposed navigation algorithm **A1**-**A4**, **B1**-**B3**

in static environments, we built a static scene in a professional computer simulation

software called V-REP with a more realistic kinetic model than the model (4.2). The static scene is a closed indoor environment with some rooms. There are twelve ToF cameras deployed in this scene to detect the obstacles and construct a WSN. The size of the room is 9m × 9m × 6m with two floors. There are some holds on the walls to connect different rooms. The safety margin d_s is 0.5m in this simulation. The number of samples in PRM algorithm is 250. In real-time implementations, more samples involve a larger time consumption (longer time delay). Thus, to avoid the collision, the suitable time window T should be chosen such that $T\delta$ is larger than the time delay. Other parameters are the same as Table 4.1. A flying robot is navigated from an initial position to a given target by the proposed algorithm. The simulation results shown in Fig. 4.26 indicates that the flying robot flew though each rooms and reached the target successfully without any collision. The minimum distance from the robot to the obstacles is indicated in Fig. 4.27. It can be seen that the robot kept the safety margin d_s during the travelling. The control input satisfies the bound constraint.

4.3.4 Simulations in dynamic environments

To confirm the performance of the proposed navigation algorithm in dynamic environments, we present another computer simulation in a dynamic scene. In the dynamic scene, a closed indoor environment are constructed by some walls and ceiling. There are four people walking in the room with the maximum speed $V_{\max} = 0.26$m/s. The size of the room is 14m × 14m × 2m with one floor. There are six ToF cameras deployed on the walls to detect the obstacles and walking people. The safety margin d_s is 0.7m in this simulation. The time window T is 6. The number of samples in PRM algorithm is 250. Other parameters are the same as Table 4.1. According to the Fig. 4.28, it can be seen that the flying robot was moving under the sensor network's navigation. It started from the initial position and avoided the walls and any moving people. Finally, the flying robot was navigated to the target successfully by keeping the given safety margin d_s (see Fig. 4.29). It should be noticed that the

(a)   (b)

(c)   (d)

(e)   (f)
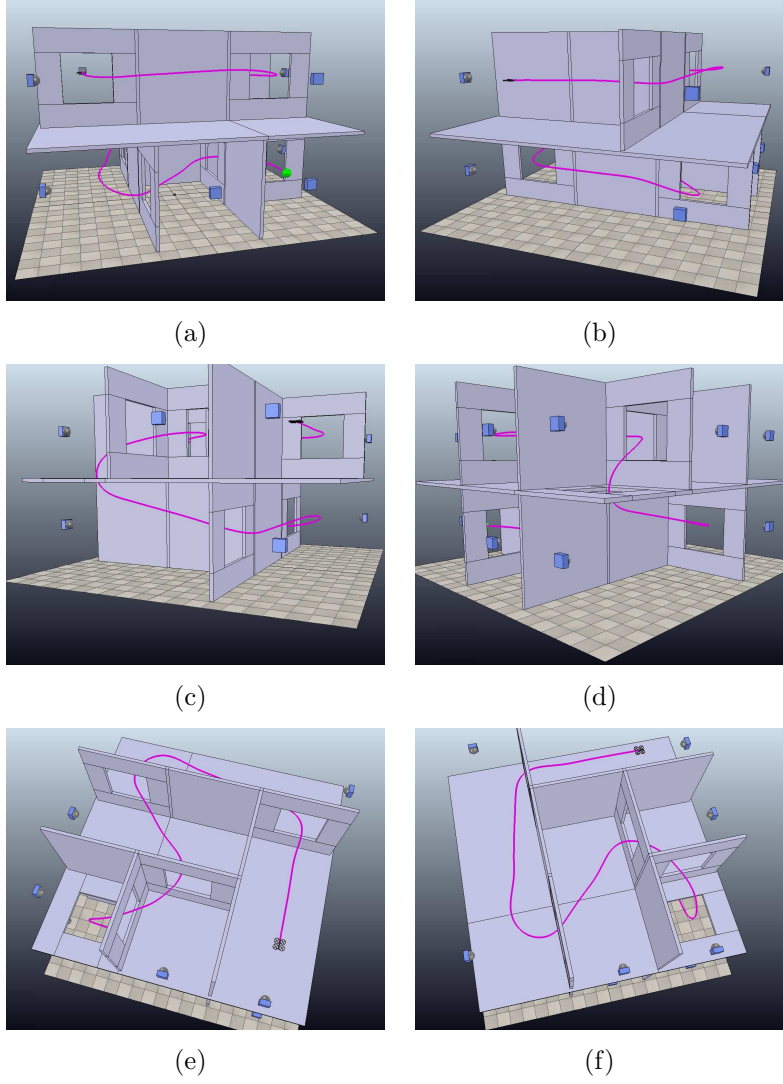
Figure 4.26: Results of the simulation in the static scene at different viewpoints. The top ceiling and exterior walls are not visible. The Green sphere indicates the robot's initial position. The magenta curve is the robot's trajectory.
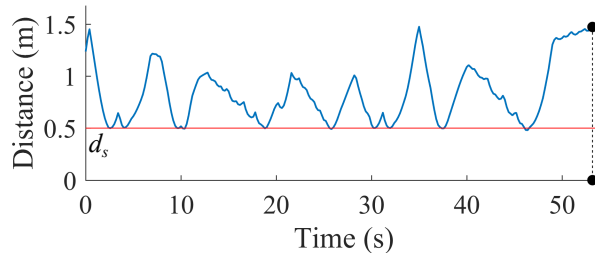
Figure 4.27: Minimum distance from the robot to the obstacles in the static scene.

safety constraint is slightly violated around $8s$ and $30s$ because in a computer control system, the discretization with non-zero sampling interval must involve errors between the ideal instant path and the generated instant path P^*. The error declines

with the decrease of the sampling interval. The sampling interval depends on the performance of the central computer and the WSN. Generally, the error is tolerable with a small sampling interval. The control input satisfies the bound constraint.

(a)   (b)

(c)   (d)
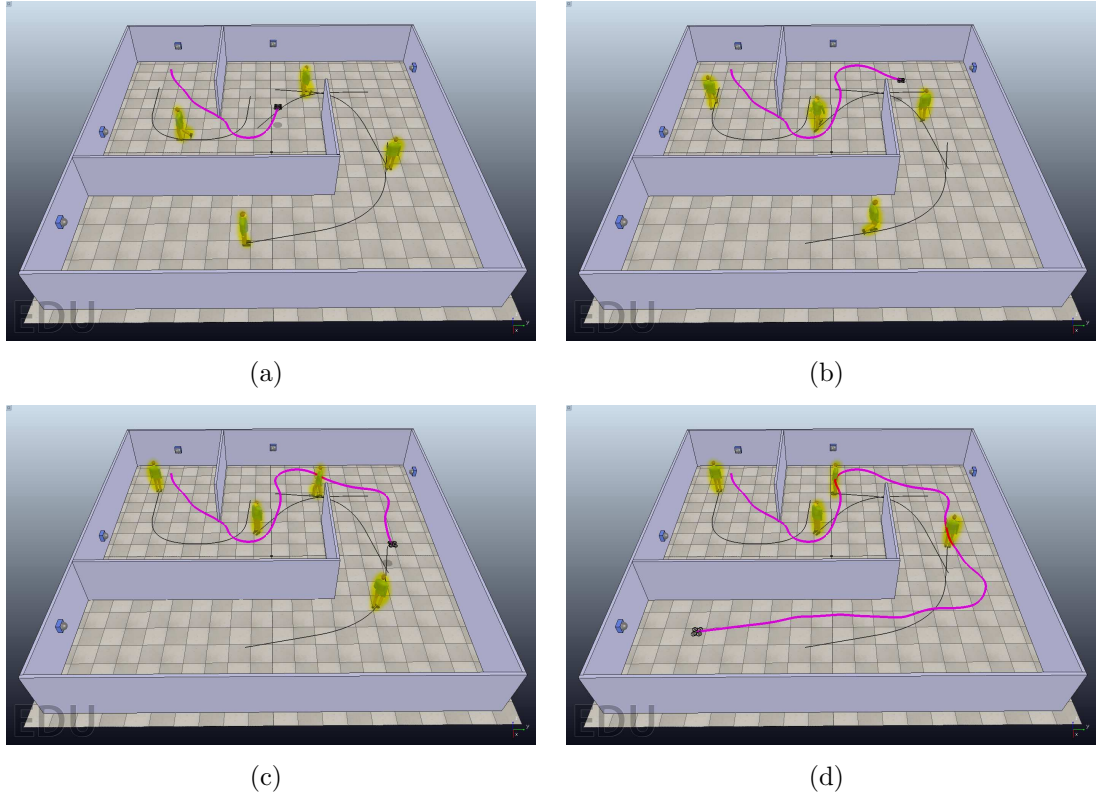
Figure 4.28: Results of the simulation in the dynamic scene at different time. The top ceiling is not visible. The magenta curve is the robot's trajectory. The yellow objects are walking people.
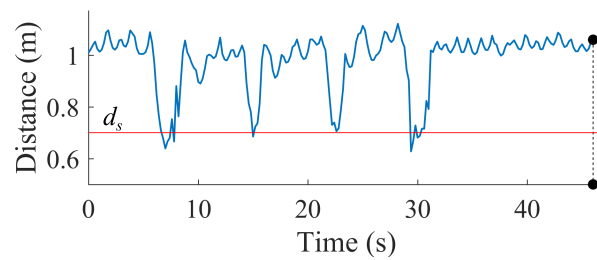
Figure 4.29: Minimum distance from the robot to the obstacles in the dynamic scene.

4.3.5   Simulations with multiple robots

In the third computer simulation, we confirm the performance of the proposed navigation algorithm in a multiple robot system. In the simulation scene, we construct

a closed indoor environment with static obstacles; e.g. walls. There are four micro flying robots working in the environment. The size of the room is 14m × 14m × 4m with one floor. To detect the environment, three time of flight cameras are deployed in the workspace. The safety margin d_s is 0.6m in this simulation. The number of samples in PRM algorithm is 350. Other parameters are the same as Table 4.1. In this simulation, four micro flying robots are navigated simultaneously by the sensor network. The robots moved from the initial positions to the targets and avoided each other with the safety margin d_s; see the simulation result in Fig. 4.30. It can be seen in Fig. 4.31 that the flying robots kept the given safety margin d_s to both the obstacles and other flying robots successfully. The control input satisfies the bound constraint.

(a)

(b)

(c)

(d)
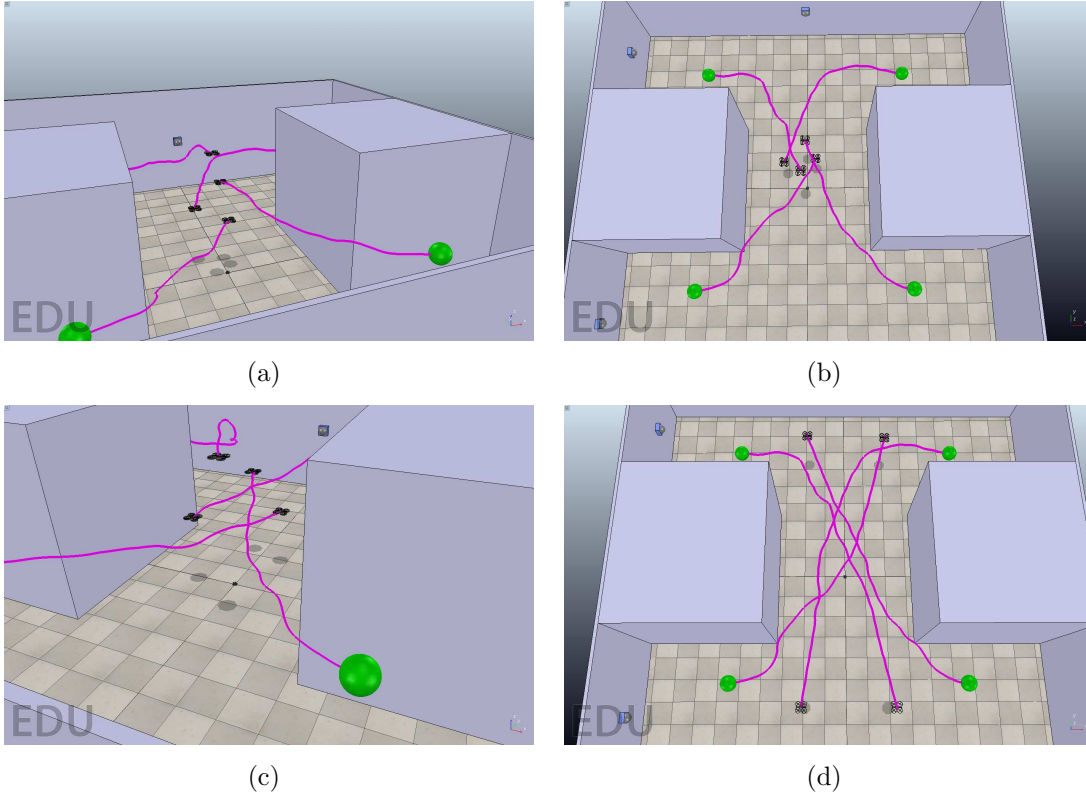
Figure 4.30: Results of the simulation with multiple micro flying robots. The top ceiling is not visible. The magenta curves are the robots' trajectories. The Green spheres indicate the robots' initial positions.
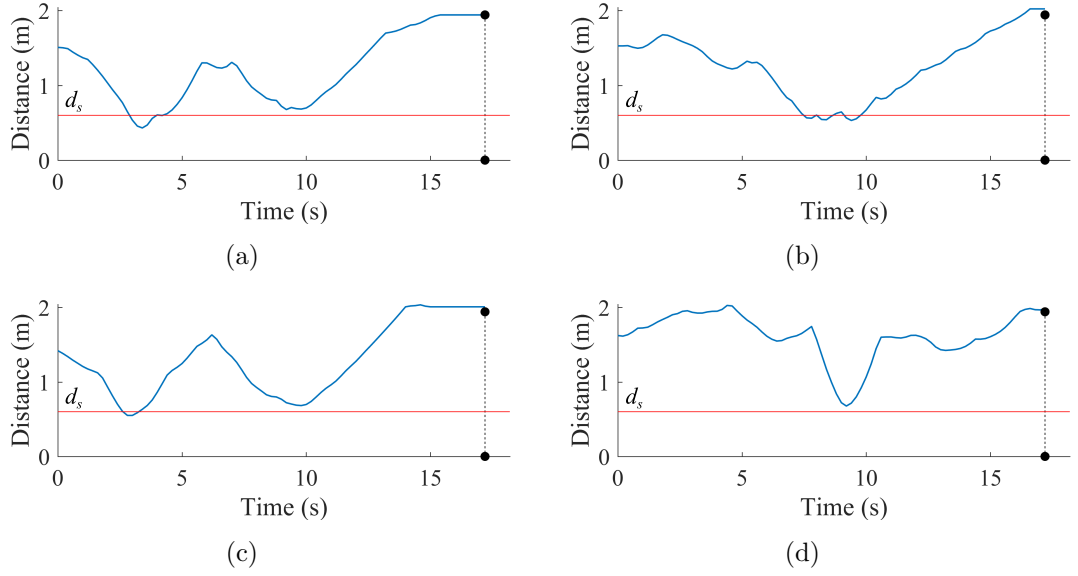
Figure 4.31: Minimum distances from each robot to the obstacles and other robots.

4.4 Summary

In this chapter, we proposed a collision-free navigation method for micro UAVs in dynamic environments by a wireless sensor network. A micro UAV cannot be equipped with heavy obstacle detection sensors. Therefore, to solve the navigation problem, a wireless sensor network consisting of 3D rang finders is involved to detect the static and dynamic obstacles in the workspace. With the navigation of the sensor network, only a path tracking controller is required for each micro flying robot. The robot can be navigated directly in the workspace without any specialization.

The proposed method can be implemented in many fields, such as the radiant intensity alarm in nuclear power stations. In the future work, a distributed navigation algorithm can be developed and other types of sensors in the sensor network can be used to make navigation system more economic.

Chapter 5

Safe Area Search and 2D Map Building of a Ground Mobile Robot in Bounded Area

This chapter is based on the the publications [158] and [157]. In this chapter, we study an area search and map building problem, which is different from Chapter 2, 3 and 4. In this chapter, we combine these two fundamental problems of modern robotics together and propose a robot navigation algorithm for exploring a complex unknown environment with obstacles and building a binary map while avoiding collisions with obstacles. The task is performed by a wheeled mobile robot equipped with a range finder sensor. We assume that the search area is cluttered with a number of possibly non-convex obstacles with unknown shapes and locations. We develop an algorithm for robot collision free navigation and map building that results in generating a complete map of the searched area based on the range finder sensor measurements. The proposed algorithm is randomized and we prove that with probability 1 the task of the area search and map building will be completed in a finite time. The control law we present in this method consists of switching between different control strategies with different conditions to construct a switched control system; see [102, 152, 159, 175].

5.1   Problem description

The Robot model we considered is the unicycle model presented in Chapter 2; see Equation (2.1). In this chapter, (x, y) is the vector of the vehicle's Cartesian coordinates, θ gives its orientation, v and u are the speed and angular velocity, respectively. The maximal angular velocity u_M is given. The minimum turning radius of the robot is

$$R_{min} = \frac{v}{u_M}. \tag{5.1}$$

The non-holonomic model (2.1) is commonly used to describe planar motion of ground robots, unmanned aerial vehicles and missiles; see e.g. [95, 161, 153, 106].

The wheeled robot is moving in a bounded planar unknown area $\mathcal{A}$ with several disjoint obstacles $D_1, \ldots, D_k$. Let the safety margin $d_0 > 0$ be given. The requirement is to drive the autonomous robot through the obstacle-free part of the area $\mathcal{A}$ while keeping the safety margin d_0 from both the obstacles and the boundary of area $\mathcal{A}$.

Available Measurements: The robot is equipped with a range finder sensor that at any time $t \geq 0$ measures the distance to the nearest object in all directions as follows: for any angle $0 \leq \omega < 2\pi$, the robot measures the distance $d(\omega, t)$ to the nearest boundary of an obstacle or the area $\mathcal{A}$ in the direction given by ω (see Fig. 5.1). Furthermore , the robot is equipped with an odometry type sensor which measures the robot's position and heading relative to its starting location and heading.
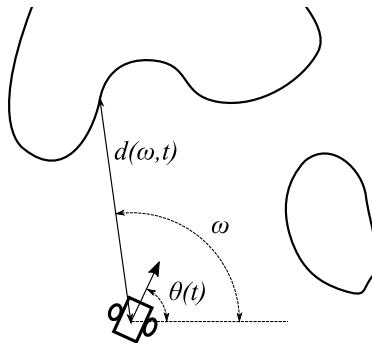
Figure 5.1: Measurements of a range finder sensor.

The goal of the robot is to build a complete map of the unknown cluttered area $\mathcal{A}$ while keeping a safe distance from the obstacles and the area boundary.

Definition 5.1.1 *Let $\mathcal{M}$ be a planar set and $\mathcal{F}$ be a binary map $\mathcal{M} \to \{0, 1\}$. The pair $(\mathcal{M}, \mathcal{F})$ is said to be the complete map of the area $\mathcal{A}$ if $\mathcal{M}$ and $\mathcal{A}$ are congruent, i.e. one can be transformed into the other by an isometry, and for any point $p \in \mathcal{M}$, $\mathcal{F}(p) = 1$ if and only if p under such isometry corresponds to a point of either the boundary of $\mathcal{A}$ or the boundary of one of the obstacles $D_1, \ldots, D_k$.*

In other words, a complete map means that the geometry of the area $\mathcal{A}$ is precisely known with all the obstacles $D_1, \ldots, D_k$ that belong to $\mathcal{A}$; see e.g. Fig. 5.2. A topological (binary) map is used to represent the environment. This map indicates the parts of the environment where the robot cannot go and the parts where it can go. The motivation to use such a map is to get a simplest representation of the environment to make our algorithm as computationally efficient as possible.
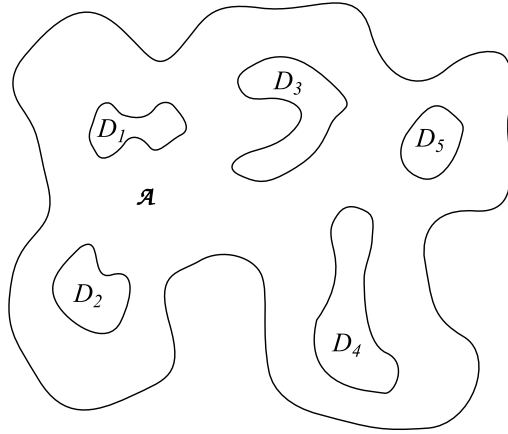
Figure 5.2: An example of a complete map.

For any robot's trajectory, the following map building algorithm is proposed.

M1: At any time $t \geq 0$, the robot builds a planar set $\mathcal{M}_0(t)$ and a binary map $\mathcal{F}_0(t)$ as follows: the set $\mathcal{M}_0(t)$ consists of all points p of the plane that are visible by the range finder sensor at time t and $\mathcal{F}_0(t)(p) = 1$ if p belongs to either obstacle boundary or area boundary, and $\mathcal{F}_0(t)(p) = 0$ otherwise; see Fig. 5.3. The pair $(\mathcal{M}_0(t), \mathcal{F}_0(t))$ is called the instant map at time t.

M2: At any time $t \geq 0$, the robot builds a planar set $\mathcal{M}(t)$ and a binary map $\mathcal{F}(t)$ by fusing with the help of odometry sensor all the instant maps $(\mathcal{M}_0(\tau), \mathcal{F}_0(\tau))$ for $\tau \in [0, t]$. The pair $(\mathcal{M}(t), \mathcal{F}(t))$ is called the total map at time t.
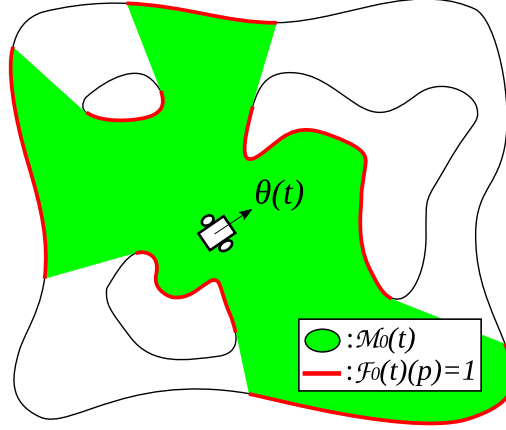
Figure 5.3: Planar set $\mathcal{M}_0(t)$ and binary map $\mathcal{F}_0(t)$.

Now some definitions are given as follows.

Definition 5.1.2 *Consider a robot trajectory and the map building algorithm* **M1**–**M2**. *A time $t_f \geq 0$ is said to be a map building completing time if the total map $(\mathcal{M}(t_f), \mathcal{F}(t_f))$ at time t_f satisfies the following property: the boundary of the set $\mathcal{M}(t_f)$ consists of points p such that $\mathcal{F}(t_f)(p) = 1$ and does not include any point p such that $\mathcal{F}(t_f)(p) = 0$; see Fig 5.4.*

Remark 5.1.1 *If t_f is a map building completing time, it can be seen that the total map $(\mathcal{M}(t_f), \mathcal{F}(t_f))$ at the time t_f is the complete map of the area $\mathcal{A}$. Moreover, if the robot has built by the time t_f the complete map of the area $\mathcal{A}$, then each point of the area outside of the obstacles was seen by the robot's sensor at some time.*

Definition 5.1.3 *A trajectory $p(t) = (x(t), y(t)), t \in [0, t_f]$ of the robot (2.1) is said to be* complete map building with obstacle avoidance *if t_f is a map completing time and the distance between the robot's position $p(t)$ and the boundary of the area $\mathcal{A}$ and the obstacles is no less than d_0 for all $t \in [0, t_f]$.*
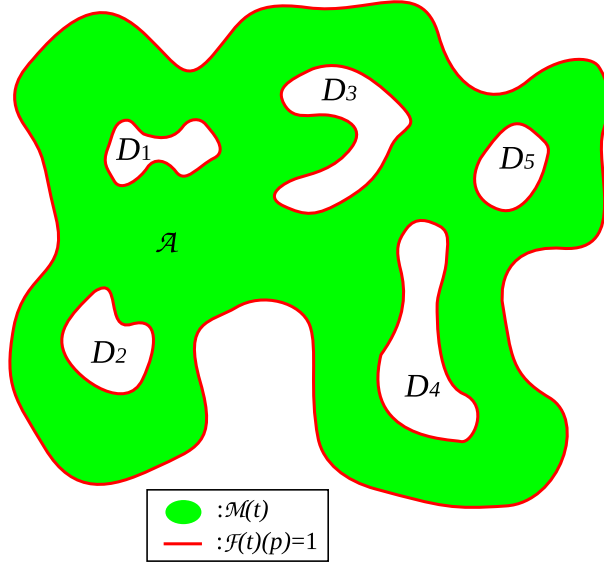
Figure 5.4: An example of $\mathcal{M}(t_f)$ and $\mathcal{F}(t_f)$ which satisfy the Definition 2.2.

Notation 5.1.1 *Let D be any closed set, p be a point in the plane. Introduce the distance $\rho(D, p)$ as*

$$\rho(D, p) := \min_{q \in D} \|p - q\|.$$

Notice that min *is achieved since D is closed. Also, $\rho(D, p) = 0$ if $p \in D$.*

Definition 5.1.4 *For $d_0 > 0$, the d_0-enlargement of the domain $D \subset \mathbf{R}^2$ is the set $\mathcal{E}[D, d_0]$ formed by all points at the distance smaller than or equal to d_0 from D, i.e., $\mathcal{E}[D, d_0] := \{p \in \mathbf{R}^2 : \rho(D, p) \leq d_0\}$. On the other hand, the d_0-reduction of the domain $D \subset \mathbf{R}^2$ is the set $\mathcal{R}[D, d_0]$ formed by all points of D at the distance greater than or equal to d_0 from the boundary ∂D of D, i.e., $\mathcal{R}[D, d_0] := \{p \in D : \rho(\partial D, p) \geq d_0\}$.*

Assumption 5.1.1 *The planar sets $\mathcal{R}[\mathcal{A}, d_0]$ and $\mathcal{E}[D_i, d_0]$, $i = 1, \ldots, k$ are closed, bounded, connected and linearly connected sets.*

Assumption 5.1.2 *The sets $\mathcal{E}[D_i, d_0]$ and $\mathcal{E}[D_j, d_0]$ do not overlap for any $i \neq j$. Any set $\mathcal{E}[D_i, d_0]$ is a subset of $\mathcal{R}[\mathcal{A}, d_0]$.*

Remark 5.1.2 *If Assumption 5.1.2 does not hold, it can be seen that map building with obstacle avoidance may be impossible (see e.g. Fig. 5.5).*

Assumption 5.1.3 *Any point of the boundaries of $\mathcal{A}$ or D_i can be connected by a straight line segment non-crossing any obstacle of the region's boundary with a point of the boundary of $\mathcal{R}[\mathcal{A}, d_0]$ or $\mathcal{E}[D_i, d_0]$, correspondingly.*

Remark 5.1.3 *Assumption 5.1.3 means that any point of the boundaries of $\mathcal{A}$ or D_i can be seen by the robot travelling along the boundaries of $\mathcal{R}[\mathcal{A}, d_0]$ or $\mathcal{E}[D_i, d_0]$, correspondingly, see e.g. Fig. 5.6. If Assumption 5.1.3 does not hold, map building with obstacle avoidance may be impossible since the boundaries of the obstacles or the region can contain some points which are impossible to see from a safe distance (see e.g. Fig. 5.7).*
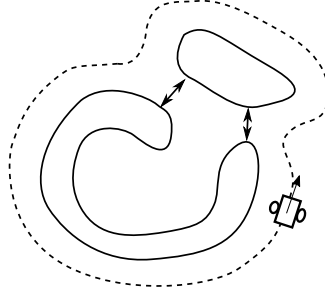
Figure 5.5: Robot cannot reach and map the area which is isolated by overlapped $\mathcal{E}[D_i, d_0]$.
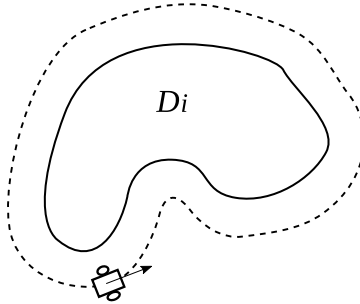
Figure 5.6: Robot can see any point of the boundary of $\mathcal{A}$ or D_i.

Notation 5.1.2 *Let $\partial D_i(d_0)$ denote the boundary of the set $\mathcal{E}[D_i, d_0]$, and $\partial \mathcal{A}(d_0)$ denote the boundary of the set $\mathcal{R}[\mathcal{A}, d_0]$.*
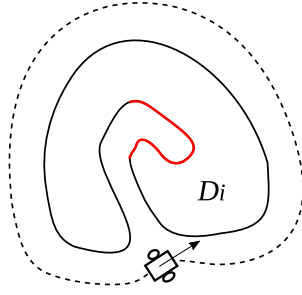
Figure 5.7: There is a segment on the boundary of D_i which cannot be seen by the robot.

Assumption 5.1.4 *For all i, the boundary $\partial D_i(d_0)$ is a non-self-intersecting, closed smooth curve with curvature $k_i(p)$ at any point p satisfying $|k_i(p)| \leq \frac{1}{R_{min}}$. Moreover, the boundary $\partial \mathcal{A}(d_0)$ is also a closed, non-self-intersecting smooth curve with curvature $k(p)$ at any point p satisfying $|k(p)| \leq \frac{1}{R_{min}}$.*

Remark 5.1.4 *Notice that the sets $\mathcal{R}[\mathcal{A}, d_0]$ and $\mathcal{E}[D_i, d_0]$ are not assumed to be convex. Therefore, their boundaries may have negative curvature at some points; see e.g. Fig. 5.8. The standard definition of curvature from differential geometry is used here (see e.g. [73]).*
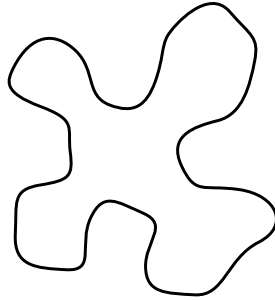
Figure 5.8: There may be negative curvature at some points on $\mathcal{R}[\mathcal{A}, d_0]$ and $\mathcal{E}[D_i, d_0]$.

Definition 5.1.5 *There are two circles with the radius R_{min} that cross the initial robot position $p(0)$ and tangent to the robot initial heading $\theta(0)$. The two circles are called initial circles.*

These initial circles are assumed to be far enough from the obstacles and the boundary of $\mathcal{A}$.

Assumption 5.1.5 *Both the initial circles lie in the set $\mathcal{R}[\mathcal{A}, d_0]$ and do not intersect the sets $\mathcal{E}[D_i, d_0]$.*

5.2 Safe navigation algorithm

In this section, the proposed algorithm of safe navigation with map building is described. First, a number of definitions are introduced.

Definition 5.2.1 *A straight line L is said to be a tangent line if one of the following conditions holds:*

1. *The line L is simultaneously tangent to two boundaries $\partial D_i(d_0)$ and $\partial D_j(d_0)$ where $i \neq j$.*

2. *The line L is simultaneously tangent to the boundary $\partial D_i(d_0)$ and an initial circle.*

3. *The line L is simultaneously tangent to the boundary $\partial D_i(d_0)$ and the boundary $\partial \mathcal{A}(d_0)$*

4. *The line L is simultaneously tangent to the boundary $\partial \mathcal{A}(d_0)$ and an initial circle.*

Points of the boundaries $\partial \mathcal{A}(d_0)$, $\partial D_i(d_0)$ and the initial circles belonging to the tangent lines are called tangent points.

For the case of simplicity, the following assumptions are introduced.

Assumption 5.2.1 *Any tangent point belongs to only one tangent line.*

Assumption 5.2.2 *If the set $\mathcal{A}(d_0)$ is non-convex, then there exists a tangent point belonging to $\partial \mathcal{A}(d_0)$.*

Remark 5.2.1 *If the set $\mathcal{A}(d_0)$ is convex, then it can be seen that the boundary $\partial \mathcal{A}(d_0)$ does not contain tangent points. If the set $\mathcal{A}(d_0)$ is non-convex, the boundary $\partial \mathcal{A}(d_0)$ may have tangent points, see Fig. 5.9.*
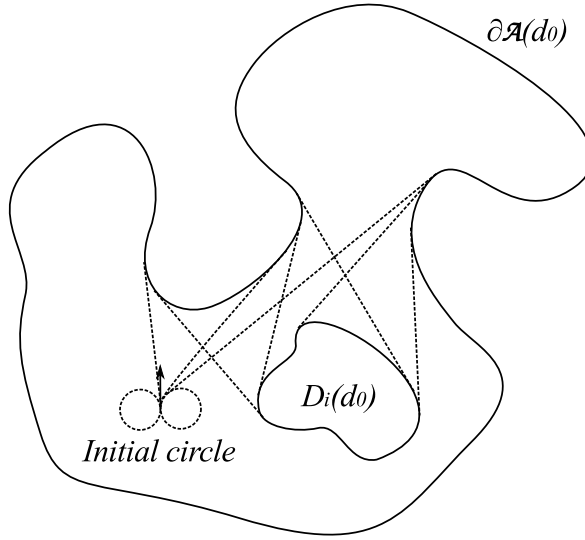
Figure 5.9: Tangent lines connecting the boundary $\partial\mathcal{A}(d_0)$ with either a boundary $\partial D_i(d_0)$ or an initial circle.

Definition 5.2.2 *Segments of tangent lines between two tangent points are called tangent segments. For any tangent point P, the direction of the corresponding tangent segment from this tangent point to the another tangent point of this segment is called the exit direction of the tangent point P; see Fig. 5.10.*
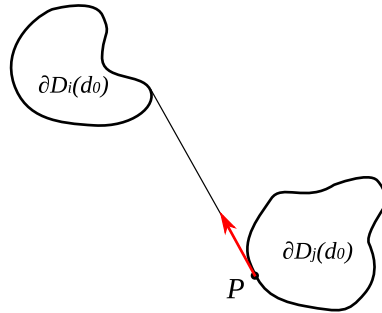
Figure 5.10: Exit direction of the tangent point P.

Definition 5.2.3 *The graph $\mathcal{G}$ is introduced that its vertices are the robot's initial position $p(0)$ and the tangent points, and its edges are the tangent segments, arcs of the initial circles and the segments of the boundaries of the extended obstacles and the reduced area that connect the vertices of the graph. The graph $\mathcal{G}$ is called the extreme graph (see e.g. Fig. 5.11). A path on this graph connecting two vertices is*

*said to be viable if the heading at the end of each edge of the path is equal to the
heading at the beginning of the next edge of the path (see Fig. 5.12)*
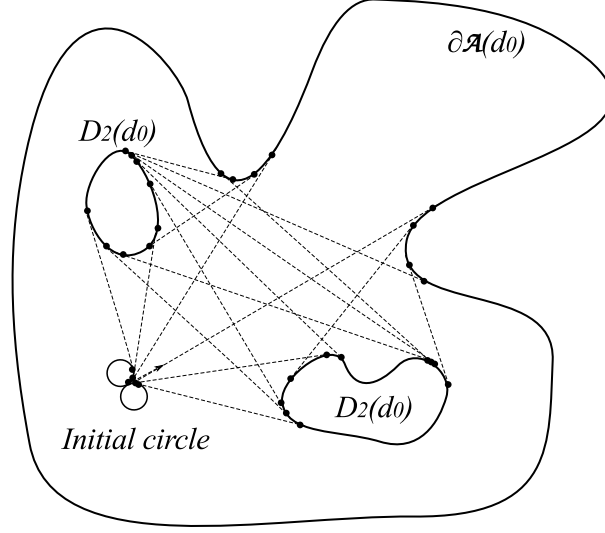
Figure 5.11: Graph $\mathcal{G}$.
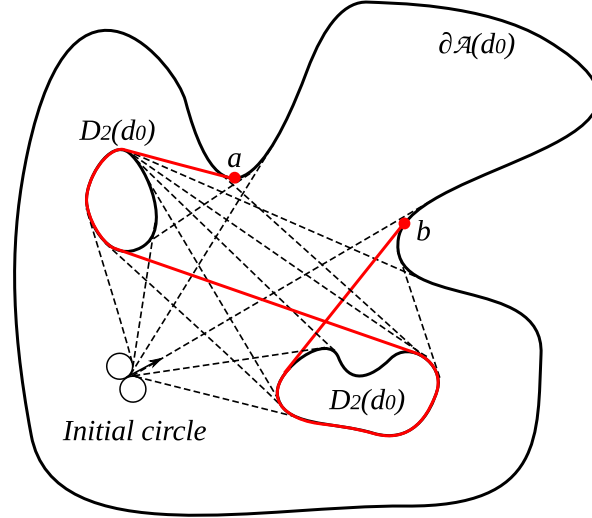
Figure 5.12: A viable path connecting the vertices a and b in the graph $\mathcal{G}$.

Assumption 5.2.3 *Let P_1 and P_2 be any two tangent points from any boundaries
$\partial D_i(d_0)$ or $\partial \mathcal{A}(d_0)$. Then there exists a viable path from P_1 to P_2 on the extreme
graph $\mathcal{G}$.*

Definition 5.2.4 *A tangent point on the boundary $\partial D_i(d_0)$ or $\partial \mathcal{A}(d_0)$ is said to be
non-circular if the other tangent point on the corresponding tangent segment does
not belong to the initial circles.*

Let $0 < q_0 < 1$ be a given number. To search and map the closed area $\mathcal{A}$ completely, the following probabilistic navigation algorithm is proposed:

A1: The robot starts to move along any of two initial circles.

A2: When the robot moving along an initial circle reaches a tangent point with the exit direction coinciding with the robot's current heading, it starts to move along the corresponding tangent segment.

A3: When the robot moves along the boundary $\partial D_i(d_0)$ or $\partial \mathcal{A}(d_0)$ and reaches a non-circular tangent point with the exit direction coinciding with the robot's current heading, with probability q_0 it starts to move along the corresponding tangent segment, and with probability $(1 - q_0)$ it continues to move along the boundary.

A4: When the robot moves along a tangent segment and reaches a tangent point on the boundary $\partial D_i(d_0)$ or $\partial \mathcal{A}(d_0)$, it starts to move along this boundary.

The navigation algorithm **A1**–**A4** is based on switching between following the initial circles, tangent lines and segments of boundaries of the extended obstacles or the reduced search area. In the step **A1**, the proposed algorithm makes the robot either move along the corresponding tangent segment (with probability q_0) or continue to move along the boundary (with probability $(1 - q_0)$). Hence, the algorithm **A1-A4** employs a degree of randomness as part of its logic and therefore belongs to the class of randomized algorithms.

Now the main theoretical result of this chapter is presented as follows.

Theorem 5.2.1 *Suppose that Assumptions 5.1.1 – 5.1.5, 5.2.1 – 5.2.3 hold and the robot is navigated by the algorithm* **A1**–**A4** *with some $q_0 \in (0, 1)$. Then for any initial position and heading of the robot, with probability 1 there exists a time $t_f \geq 0$ such that the trajectory $p(t) = (x(t), y(t)), t \in [0, t_f]$ of the robot (2.1) is complete map building with obstacle avoidance.*

Assumptions 5.1.1 – 5.1.5, 5.2.1 – 5.2.3 are needed for mathematically rigorous proof of the main theoretical result (Theorem 5.2.1). In practice, the proposed algorithm often performs well even when some of them do not hold.

The proof of Theorem 5.2.1 The first step in the proof of Theorem 5.2.1 is to prove that with probability 1 there exists a time $t_f \geq 0$ which is a map building completing time (see Definition 5.1.2). Indeed, there are the following two possible cases:

Case 1: The set $\mathcal{R}[\mathcal{A}, d_0]$ is non-convex.

Case 2: The set $\mathcal{R}[\mathcal{A}, d_0]$ is convex.

In **Case 1**, Assumption 5.2.2 implies that the algorithm **A1**−**A4** includes switching between tangent lines and the boundaries $\partial \mathcal{A}(d_0)$ and $\partial D_i(d_0)$ with some non-zero probabilities. Therefore, it follows from Assumption 5.2.3 that with probability 1 there exists a time $t_f \geq 0$ such that by this time the robot's trajectory includes all the points of the boundaries $\partial \mathcal{A}(d_0)$ and $\partial D_i(d_0)$ for all i (notice that the robot can travel along the boundaries in either clockwise or counter-clockwise direction). Now it immediately follows from Assumption 5.1.3 that the robot by the time t_f has seen every point of the boundaries of the region and the obstacles. Furthermore, the navigation algorithm **A1**−**A4** consists of moving along the boundaries $\partial D_i(d_0)$, $\partial \mathcal{A}(d_0)$, two initial circles and tangent lines between them. According to Assumption 2.5, the distance between both initial circles and obstacles and the boundary of the region is greater or equals d_0. Therefore, under the navigation algorithm **A1**−**A4**, the robot always moves along lines such that the distance between these lines and the obstacles and the boundary of the region is greater or equals d_0. Hence, the distance between the robot and the obstacles and the boundary of the region is greater or equals d_0.

In **Case 2**, there is no a tangent line connecting a tangent point on the boundary $\partial \mathcal{A}(d_0)$ with either a boundary $\partial D_i(d_0)$ or an initial circle since $\partial \mathcal{A}(d_0)$ does not have tangent points. In this case, the robot does not travel along $\partial \mathcal{A}(d_0)$. The algorithm **A1**−**A4** includes switching between tangent lines and the boundaries $\partial D_i(d_0)$ with some non-zero probabilities. Therefore, it follows from Assumption 5.2.3 that with probability 1 there exists a time $t_f \geq 0$ such that by this time the robot's trajectory includes all the points of the boundaries $\partial D_i(d_0)$ for all i. Assumption 5.1.3 imme-

diately implies that the robot by the time t_f has seen every point of the boundaries of the obstacles. Moreover, this and the convexity of the set $\mathcal{R}[\mathcal{A}, d_0]$ imply that the robot by the time t_f has seen every point of the boundary of the region $\mathcal{A}$ as well. Indeed, Let P be an arbitrary point of the boundary of $\mathcal{A}$. If there exists a straight line segment connecting P with a point P_1 of a boundary $\partial D_i(d_0)$ which does not cross any obstacle (see Fig. 5.13) than the robot sensed the point P when it was at P_1. If there is no such a straight line segment then the region $\mathcal{A}$ does not contain any obstacles. In this case, the algorithm **A1**−**A4** results in travelling along an initial circle and the map of the region will be built after one complete turn (see Fig. 5.14). The proof of the statement that the algorithm **A1**−**A4** guarantees that the distance between the robot and the obstacles and the boundary of the region is greater or equals d_0 is exactly the same as in **Case 1**. This completes the proof of Theorem 5.2.1.
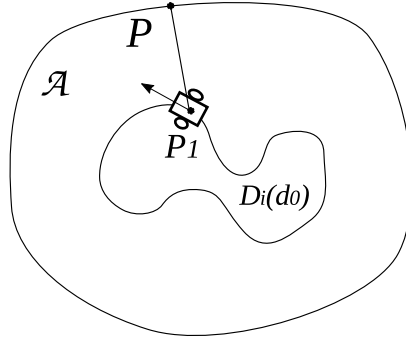
Figure 5.13: A straight line segment connecting P with a point P_1 of a boundary $\partial D_i(d_0)$.
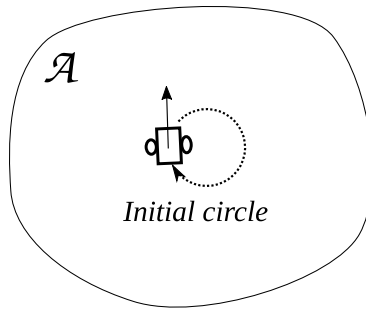
Figure 5.14: Map is built after a complete turn of an initial circle.

Remark 5.2.2 *It should be pointed out that the robot will stop at a time $t_f \geq 0$ that is a map building completing time (see Definition 5.1.2). It immediately follows from Remark 5.1.1 and Theorem 5.2.1 that trajectory $p(t) = (x(t), y(t)), t \in [0, t_f]$ of the robot (2.1) navigated by the algorithm **A1**–**A4** is complete map building with obstacle avoidance.*

5.3 Computer simulations

In this section, computer simulations of the collision free map building navigation algorithm **A1**–**A4** are presented. In these simulations, the odometry errors, range finder noise and dynamic environments are discussed. Finally, our algorithm is compared with another environment exploration and map building algorithm.

To follow tangent lines and boundaries of the reduced region and the extended obstacles, a sliding mode control law which is a modification of the law of [153] is used, that consists of switching between a boundary following approach proposed in [107], and the pure pursuit navigation approach; see e.g. [161]. Our sliding mode navigation law can be expressed as follows:

$$u(t) = \begin{cases} \pm u_M & R1 \\ \Gamma sgn\left[\phi_{tan}(t)\right] u_M & R2, \\ \Gamma sgn\left[\dot{d}_{min}(t) + X(d_{min}(t) - d_0)\right] u_M & R3 \end{cases} \quad (5.2)$$

where the function $sgn(x)$ is defined as follows:

$$sgn(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases} . \quad (5.3)$$

This navigation law defined three control strategies corresponding to three separate modes $R1 - R3$. There are three rules for switching the mode between three modes $R1 - R3$. Initially mode $R1$ is active, and transitions to other modes are determined as follows:

1. $R1 \to R2$: it occurs when the robot is at a tangent point on one of the initial circles and the exit direction at the tangent point coincides with the robot's current heading.

2. $R2 \to R3$: it occurs when the robot is at a tangent point on a tangent segment and reaches the boundary $\partial D_i(d_0)$ or $\partial \mathcal{A}(d_0)$.

3. $R3 \to R2$: with the probability q_0, it occurs when the robot is at a tangent point on the boundary $\partial D_i(d_0)$ or $\partial \mathcal{A}(d_0)$ and the exit direction at the tangent point coincides with the robot's current heading.

Mode $R1$ describes motion along the initial circle with maximal angular velocity. Mode $R2$ describes pursuit navigation, where $\phi_{tan}(t)$ is defined as the angle between the robot's heading and a line segment connecting the vehicle and the currently tracked tangent segment. Mode $R3$ describes boundary following navigation, where the control calculation is based on the minimum distance to the nearest obstacle, defined as $d_{min}(t)$. The control law (5.2) consisting of switching constructs a hybrid control system; see [102, 152, 159, 175].

This control law is subject to some restrictions which are inherited from [107]. The variable Γ is defined as $+1$ if the boundary followed is on the left of the tangent being tracked, -1 if it is on the right. A constant $d_{trig} > d_0$ is also introduced to determine when the control system transitions to boundary following mode [107]. The saturation function X is defined as follows:

$$
X(r) = \begin{cases} lr & |r| < k \\ lk\ sgn(r) & otherwise \end{cases} \tag{5.4}
$$

where l and k are tunable constants. Because of any potential chattering in the robot's heading caused by the sliding mode control, once it is decided to not pursue a tangent line, there is a short pause until tangent following can potentially be engaged again. It means, during this short pause, the mode $R3$ will not switch to $R2$ even if the robot's heading coincides with a tangent line again.

According to the system description, the robot is equipped with a range finder sensor and an odometry sensor. The range finder sensor can inform the distance from the mobile robot to the nearest obstacles in a finite number of directions. The measurements of the range finder sensor can be represented as a finite point set $\{W_i(\omega_i, \varsigma_i)|i = 1, 2 \ldots n\}$. ς_i is the distance from the robot to the nearest obstacle in the direction ω_i. ω_i is the direction angle relative to the robot's heading. Moreover, the odometry sensor measures the robot's speed and angular velocity. The robot can calculate its position and orientation by using odometry.

Notice that the robot will stop at a time $t_f \geq 0$ that is a map building completing time (see Definition 5.1.2). As it is pointed out in Remark 5.2.2, in this case, the trajectory $p(t) = (x(t), y(t)), t \in [0, t_f]$ of the robot (2.1) navigated by the algorithm **A1**−**A4** is complete map building with obstacle avoidance.

Now, the following approaches to implement the proposed algorithm **A1**−**A4** are presented.

With a small angular resolution $\Delta\omega$ of the range finder sensor, the variable $d_{min}(t)$ can be approximated by the minimum value ς_{min}. Similarly, $\dot{d}_{min}(t)$ can be approximated by $\frac{d_{min}(t+T) - d_{min}(t)}{T}$ with a short sampling interval T.

To determine when the mode $R2$ should be activated, the following steps are designed to check whether the robot's orientation coincides with a tangent line (see Fig. 5.15):

1. Calculate $|\Delta\varsigma_i|$ between successive ς_i and ς_{i+1} for any $i = 1, 2 \ldots n-1$. According to the Assumption 2.2, the minimum distance between any two obstacles is greater than $2d_0$. Therefore, the tangent lines on the boundary of obstacles are obtained by checking if $|\Delta\varsigma_i| > 2d_0$. When there exists any $|\Delta\varsigma_k| > 2d_0$, let ς'_k denote the smaller value between ς_k and ς_{k+1}. Then, the line segment Ψ_k from the pole to the point $(\omega'_k, \varsigma'_k)$ is a tangent line on an obstacle.

2. According to the geometry, let Ψ_k be a leg and construct a right triangle. The length of another leg which shares the common endpoint $(\omega'_k, \varsigma'_k)$ with Ψ_k is d_0. After this, the hypotenuse ξ_k can approximately represent the tangent

line on the corresponding safe boundary $\partial D(d_0)$. Note that if $\varsigma_k < \rho_{k+1}$, the polar angle of ξ_k is bigger than the polar angle of Ψ_k (see ξ_2 in Fig. 5.15). If $\varsigma_k > \rho_{k+1}$, the situation is opposite (see ξ_1 in Fig. 5.15).

3. Define a safe area Ω_k for each line segment ξ_k (see Fig. 5.16). The set Ω_k is formed by all points at the distance smaller than d_0 from the line segment ξ_k. If there is any point W_i within the area Ω_k, ξ_k is not a real tangent line because in geometry, it intersects with other $\partial D(d_0)$ and can not be pursued safely by the robot.

4. If there exist a ξ_k that the absolute value of its angle ω_k in the local polar coordinate system is smaller than a given threshold θ_{trig}, the robot's orientation approximately coincides with the tangent line ξ_k.

To calculate $\phi_{tan}(t)$ in the equation (5.2), a intermediate target P_T is defined. Assume that when the mode $R2$ is activated, the robot's orientation coincides with the tangent line ξ_k. Then, the temporary target P_T is the endpoint of ξ_k. After this, $\phi_{tan}(t)$ is the angle between the robot's heading and the vector from the robot's position $(x(t), y(t))$ to the intermediate target P_T.
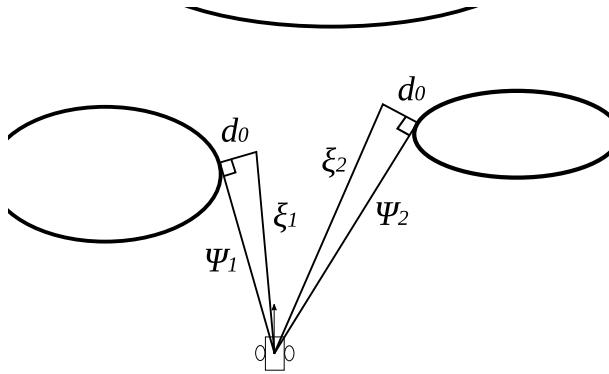
Figure 5.15: Calculation of the tangent line ξ_k.

A raster graphics map is used to represent the environment. Each cell of the map has three optional states (unknown, unoccupied and occupied). Initially, all the cells are unknown. At each sampling time, the measurements $W_i(\omega_i, \varsigma_i)$ are transformed from the local polar coordinate system to the global Cartesian coordinate system
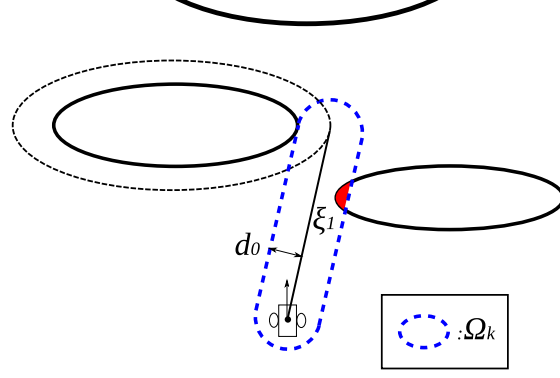
Figure 5.16: Definition of safe area Ω_k.

$W_i(x_{W(i)}, y_{W(i)})$. Then, the Bresenham's line algorithm [15] is used to determine the state of each cell intersecting with the line segments between robot's position $(x(t), y(t))$ and each $W_i(x_{W(i)}, y_{W(i)})$. The state of each cell is updated by the following strategy:

1. If an unknown cell or an unoccupied cell is occupied by $W_i(x_{W(i)}, y_{W(i)})$, the cell will be updated to occupied cell.

2. If an unknown cell intersects with a line segment between $(x(t), y(t))$ and $W_i(x_{W(i)}, y_{W(i)})$ and is not occupied by $W_i(x_{W(i)}, y_{W(i)})$, the cell will be updated to unoccupied cell.

In the implementations of the proposed algorithm, there are some parameters which should be determined as follows:

1. The maximal angular velocity u_M is a known physical parameter of a mobile robot.

2. For a real robot, the safety margin d_0 should be bigger than the maximum distance from the robot's center to its boundary. Moreover, it should be smaller than a half of the minimal distance between any obstacles (see Assumption 5.1.2).

3. The robot's speed v should be smaller than $u_M d_0$ because R_{min} should be smaller than d_0 (see Assumption 5.1.4).

4. The two thresholds θ_{trig} and d_{trig} should be tuned in different implementations to guarantee the correct mode transitions.

5. If the precision of a range finder sensor is $\pm e$, the resolution of the map is recommended to be greater than $(2e)^2/cell$ because a higher resolution than $(2e)^2/cell$ is unnecessary.

In the following parts, five groups of computer simulations are carried out to confirm and discuss the performance of the proposed algorithm.

Simulation 1: The proposed algorithm was simulated in two 112.5m $\times$ 112.5m cluttered environments with different types of obstacles. In the first scene, there are some non-convex obstacles. The parameters used for the simulation are listed in Table. 5.1. The result of the simulation in the first scene is shown in Fig. 5.17. It can be seen that the robot searched the whole area successfully and built a complete map of the environment in 920.5s. The control input satisfies the bound constraint.

Table 5.1: Parameters for Simulation 1

Speed of robot	v	1m/s
Maximum angular velocity	u_M	1.4rad/s
Minimum turning radius	R_{min}	0.71m
Safety margin	d_0	1.7m
Sampling interval	T	0.25s
Angle threshold	θ_{trig}	0.08rad
Distance threshold	d_{trig}	0.3m
Probability	q_0	0.4
Resolution of the map		20cm^2/cell

In the second scene (a maze), the safety margin was changed to 4m to make the result clear to see. Thus, the speed and angular velocity used for this simulation were changed to 0.8m/s and 0.5rad/s. As the result shows in Fig. 5.18, the robot started from a initial position and searched the whole area of this maze without any collision. Finally, it took 1168s to build a complete map of this environment.

Simulation 2: In simulation 2, the range finder sensor is considered as a laser range finder and the noise model proposed in [63] was used to simulate the laser

(a) $t = 125$s

(b) $t = 920.5$s
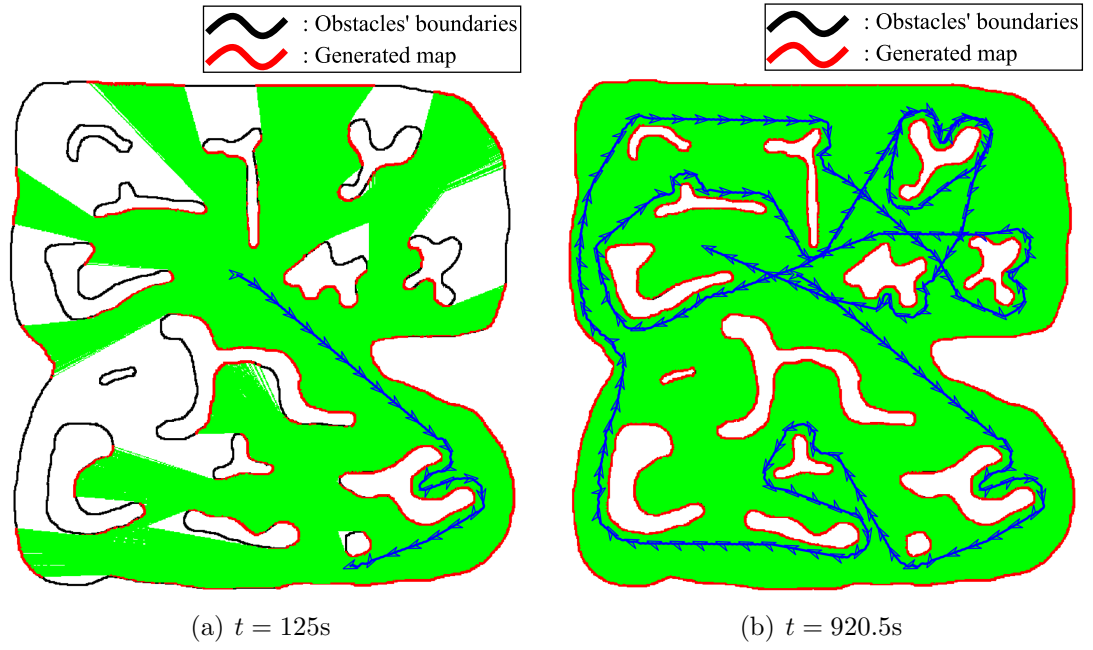
Figure 5.17: Result of the simulation in the first scene of Simulation 1.

(a) $t = 250$s

(b) $t = 1186$s
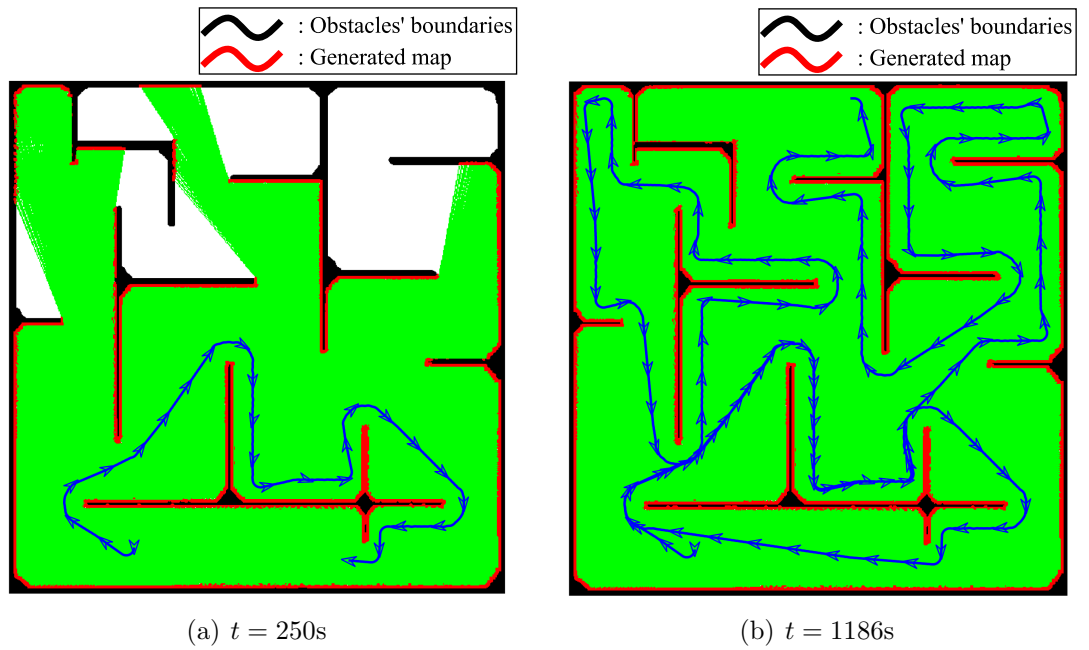
Figure 5.18: Result of the simulation in the second scene of Simulation 1.

range finder noise. The modelled noise is a combination of flicker noise and white noise with standard deviation of 1.77023mm and 3.03856mm (see Fig. 5.19). In this simulation, the modelled noise was increased by 5 times to make the effects of the noise more obvious in the result of the simulation. The size of the environment is 11.25m × 11.25m and the resolution of the map is $1\text{cm}^2/cell$. Fig. 5.20 shows the complete map in this simulation with the laser range finder noise. It can be seen that the proposed algorithm is robust with the range finder errors, although the map quality is influenced (i.e. rough boundaries in generated maps).
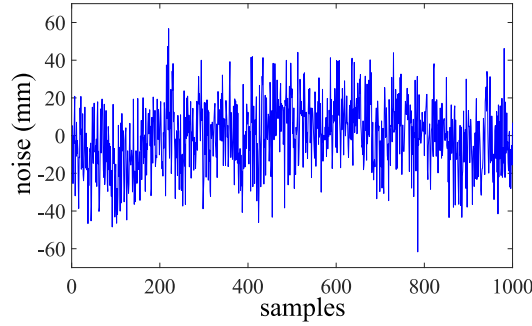
Figure 5.19: Modelled laser range finder noise.

(a) Obstacles' boundaries   (b) Generated map
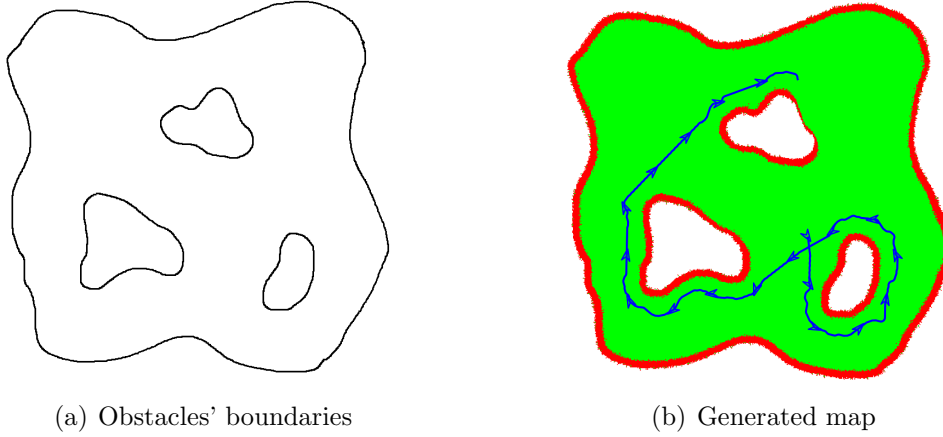
Figure 5.20: Result of Simulation 2 with range finder noise.

Simulation 3: In the proposed algorithm, odometry is used to estimate the position and orientation of the robot. The odometry errors are caused by many reasons [13]. In this simulation, only the odometry errors caused by the approximation in the odometry equations [13] are considered. The sampling interval in this simulation is 0.5s, which caused an obvious odometry errors. The size of the

environment is 11.25m × 11.25m. The result of the simulation (see Fig. 5.21) shows that the estimation of the robot's position diverged from the real position gradually with the increase of time. As a result of odometry errors, there are obvious errors on the generated map.
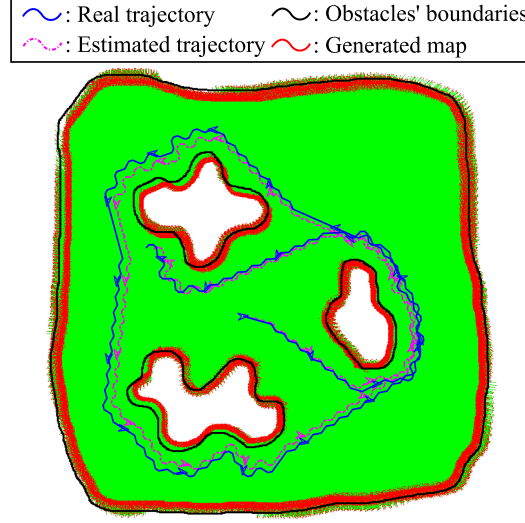
Figure 5.21: Result of Simulation 3 with odometry errors.

To alleviate the odometry errors, an improved estimation model with the help of an accelerometer is proposed here. The accelerometer measures the acceleration of the robot in the direction vertical to the robot's heading (see Fig. 5.22). Let $\hat{v}$ and $\hat{u}$ be the speed and angular velocity measured by odometry sensors, respectively. Let $\hat{a}$ be the acceleration measured by the accelerometer. In the improved estimation model, $\hat{v}$, $\hat{u}$ and $\hat{a}$ are approximated as constant during the sampling interval T. Based on the approximation, the improved estimation model is proposed as follows:

$$x(k+1) = x(k) + Q\cos\theta(k) + \frac{\hat{a}T^2}{2}\cos(\theta(k) + \frac{\pi}{2})$$
$$y(k+1) = y(k) + Q\sin\theta(k) + \frac{\hat{a}T^2}{2}\sin(\theta(k) + \frac{\pi}{2}) \quad , \qquad (5.5)$$
$$\theta(k+1) = \theta(k) + \hat{u}(k)$$

where

$$Q = \int_0^T \sqrt{\hat{v}^2 - (\hat{a}t)^2}\, \mathrm{d}t = (\frac{T}{2}\sqrt{\hat{v}^2 - (\hat{a}T)^2} + \frac{\hat{v}^2}{2\hat{a}}\arcsin(\frac{\hat{a}T}{\hat{v}^2})). \qquad (5.6)$$

To confirm the performance of the improved estimation model (5.5), a new simulation is carried out with this improved model in a same environment. The result of
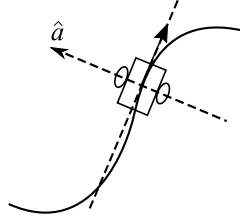
Figure 5.22: Direction of the acceleration measured by the accelerometer.

the new simulation is shown in Fig. 5.23. Comparing with Fig. 5.21, the odometry errors are alleviated significantly by the improved estimation model (5.5). Additionally, for other types of odometry errors, there are many methods focusing on the alleviation of odometry errors e.g. [34, 35, 148].
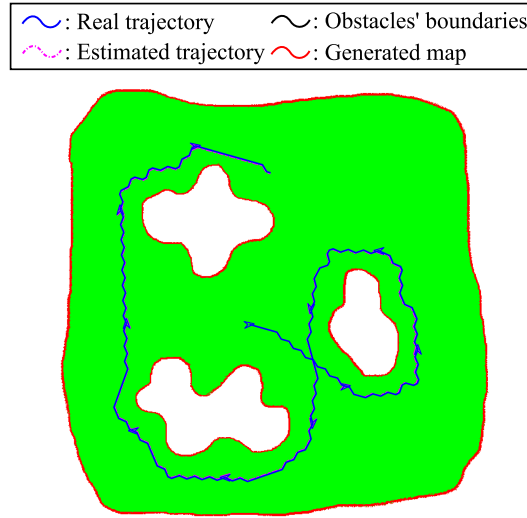
Figure 5.23: Result of the simulation with the improved estimation model (5.5). Notice that the obstacles' boundaries nearly coincide with the generated map and the real trajectory of the robot nearly coincides with the estimated trajectory.

Simulation 4: In Simulation 4, a dynamic environment with both steady and moving obstacles is considered. In the dynamic environment, the proposed algorithm should be modified to explore the environment safely and build the complete map correctly. In this simulation, the robot navigation algorithm contains a simple motion detector block to distinguish moving obstacles and avoid collisions with them. There are many methods focusing on the moving obstacle detection e.g. [111, 140, 204]. Moreover, in our future research, more advanced methods of avoiding collisions with moving obstacles can be integrated with the proposed algorithm

(see e.g. [56, 106]). With the motion detector block, the robot can avoid pursuing a tangent segment which is tangent to a moving obstacles (see algorithm **A3**). With the update of the map during the exploration, the boundaries of moving obstacles on the map will be eliminated completely and it will end up with a correct map of the steady environment (the region and all its steady obstacles).

With the above modifications, Simulation 4 is carried out with two moving obstacles in a 5m × 5m environment (see Fig. 5.24). The result of the simulation is shown in Fig. 5.25. It can be seen that the robot successfully explored the area without pursuing any tangent line belonging to moving obstacles. Moreover, it can be seen that, at the time 47.1s, the robot avoided the moving obstacle when pursing a tangent line. According to the update of the map, it can be seen that the boundaries of moving obstacles are eliminated gradually. Finally, the robot built a correct complete map of the region and all its steady obstacles.
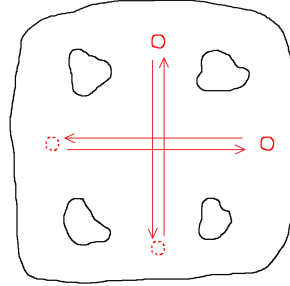
Figure 5.24: Dynamic environment in Simulation 4.

Simulation 5: In recent researches, the frontier-based exploration algorithms are the most common exploration strategies. In this simulation, a frontier-based exploration algorithm proposed in [190] was selected to compare the exploration time with the proposed algorithm. The selected algorithm has a better performance than current exploration methods. It keeps the map information integrated during exploration and uses a multi criteria decision method to overcome the drawbacks of the traditional weighted average method to improve the efficiency of the exploration.

Notice that, although the Growing Neural Gas network topological map is used to represent the environment, which is different from geometrical map, it does not affect the exploration time. In this simulation, the algorithm of [190] was simulated

(a) $t = 6.1$s

(b) $t = 47.1$s

(c) $t = 112.2$s
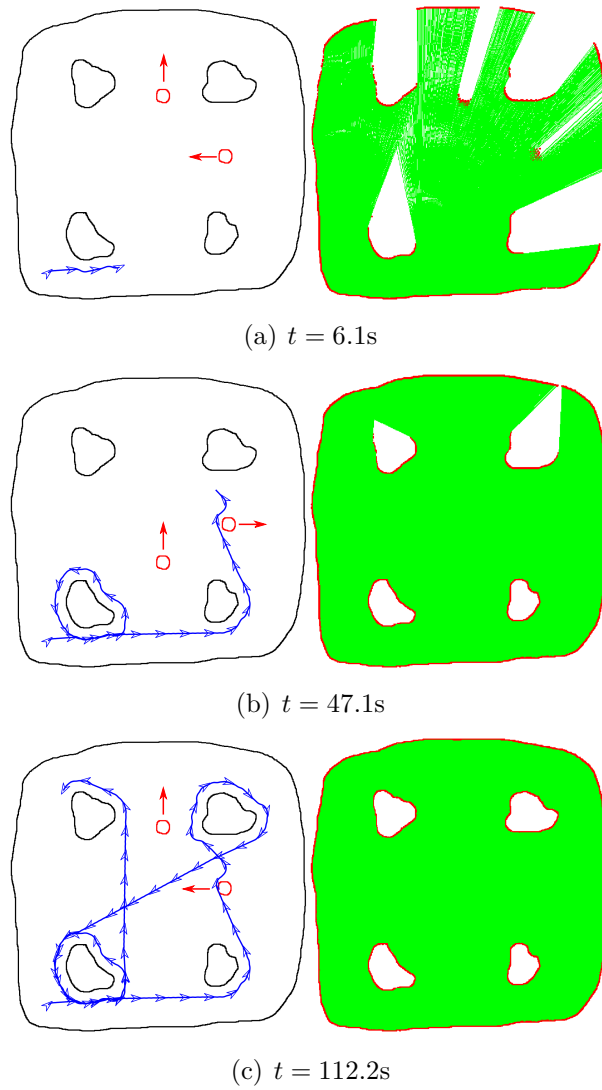
Figure 5.25: Result of Simulation 4 in the dynamic environment.

and run in some new examples to compare the exploration time with the proposed algorithm.

In these new examples, the exploration time of the two algorithms is compared in different environment with different number of obstacles. Notice that the algorithm of [190] refers to a front exploration area which is a semi-circle at the front of the robot. Therefore, different radius of the front exploration area was considered in the simulations.

In the comparison, the size of the environment is 37.5m × 37.5m and the robot's speed is 0.2m/s. Fig. 5.26 shows the different exploration time of two algorithms with different number of obstacles. The performance of the proposed algorithm refers to Line A. The performance of the algorithm of [190] with different radius of 5m, 6m, 7m and 8m refers to Line B1 to B4, respectively.
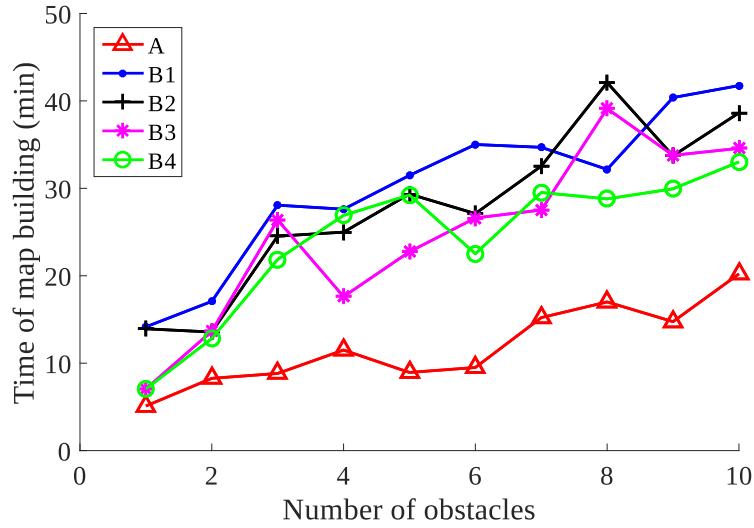
Figure 5.26: Exploration time of the two algorithms with different number of obstacles.

According to the comparison, it can be seen that our randomized search algorithm has a strong advantage because our method builds a complete map of the environment nearly twice faster than the algorithm of [190]. Furthermore, our algorithm is designed for a non-holonomic model (see Equation (2.1)), whereas the algorithm of [190] and the most of works in this fields do not consider any non-holonomic constraint of the robot's motion. Moreover, the algorithm of [190] may

fail to search some part of the area when the diameter of the front exploration area is smaller than the minimum distance between any obstacles. Additionally, the performance of our algorithm is verified by a mathematically rigorous proof. Finally, the developed randomized algorithm requires a smaller computational load than the exploration algorithm proposed in [190].

5.4   Experiments with real mobile robot

In this section, the real implementation of the proposed collision free map building algorithm **A1**–**A4** are proposed with a Pioneer-3DX wheeled mobile robot. The robot has an on-board computer with a Linux operating system. It can be programmed to control the speed of each wheel separately and process the data of the mounted sensors. The Pioneer-3DX robot calculates its real-time position and orientation by odometry.

In the experiments, the robot is equipped with a SICK LMS-200 laser range finder (LRF) and an odometry sensor. This LRF's angular resolution, scanning range and measurement range are $0.5°$, $180°$ and 80m respectively. The LRF's precision is $±15$mm. Notice that, to guarantee that the minimum distance to obstacles can be measured correctly and any point on obstacles' boundaries can be detected by the range finder, the scanning range of the LRF should be greater than or equal to $180°$. If the scanning range is smaller than $180°$, the minimum distance cannot be measured correctly and some parts of obstacles' boundaries may never be detected (see Fig. 5.27). The change of the scanning range does not affect the final result of the map building and safe navigation. However, with the decrease of the scanning range, the map building completing time will increase. Moreover, it is noticed that the scene we built in the experiments is not quite large. Thus, the odometry errors of the robot is quite small and the generated map is accurate. If the proposed algorithm is implemented in a larger scene, the approach proposed in Simulation 3 can be used to reduce odometry errors. Other technologies can also be used to reduce the localization error of the robot such as the use of Global Positioning

System (GPS) in outdoor environments.
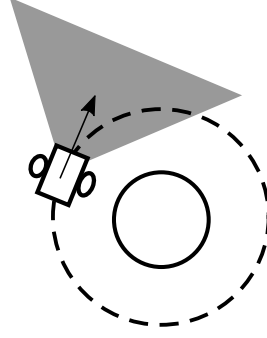
Figure 5.27: The case when scanning range is smaller than 180°.

A closed area in an indoor environment was built by blocking the hallway and doors. Seven obstacles were placed on the floor. Each obstacle was higher than the robot to guarantee that the obstacle can be detected by the LRF. Considering the measurement range of the LRF, the maximum distance from any point to any boundary should be smaller than 80m.

The robot was programmed with the proposed algorithm and put into the environment. When the robot started running, its mode R1 was activated and it moved along the initial circle firstly (see **A1**). When its heading coincided with a tangent line ξ (See Fig. 5.15), its mode was transitioned to R2 from R1 or R3 (see **A2** and **A3**). When the robot reached the endpoint of the line segment ξ, its mode was transitioned back to R3 (see **A4**) and it started tracking $\partial D_i(d_0)$ or $\partial \mathcal{A}(d_0)$. During the period of the safe navigation, the robot was continuously updating the map and checked whether the map building is completed (see Definition 5.1.2). Finally, the robot stopped when the complete map was built.

Remark 5.4.1 *The saturation function (5.4) is used to replace the sign function to reduce control chattering, which is undesired in experiments with a real robot [181].*

In the following parts, five experiments are carried out with different probability q_0 in the preset environment to confirm the performance of the proposed algorithm. The parameters used in these experiments are listed in Table. 5.2. The probability q_0 for the five experiments are 0.3, 0.4, 0.5, 0.6 and 0.7. The control input satisfies the bound constraint.

Table 5.2: Parameters for Experiment 1

Speed of robot	v	0.15m/s
Maximum angular velocity	u_M	0.4rad/s
Minimum turning radius	R_{min}	0.375m
Safety margin	d_0	0.55m
Sampling interval	T	0.2s
Angle threshold	θ_{trig}	0.1rad
Distance threshold	d_{trig}	0.1m
Resolution of the map		$3\text{cm}^2/cell$

At the beginning, the robot selected one of two initial circles randomly to move. When the robot's heading coincided with a tangent segment, the robot left the initial circle and reached the enlarged boundary by pursuing this tangent segment. When reached the enlarged boundary, the robot started patrolling this boundary and randomly selected the next tangent segment to pursue. Finally the robot explored the closed area completely and built a correct complete map of this environment. This procedure is indicated in Fig. 5.28 and 5.29. Fig. 5.28 shows the generated maps and the robot's trajectory at different time in the experiment with probability $q_0 = 0.6$. Fig. 5.29 shows some pictures of the corresponding experiment.

The results of the five experiments including the generated complete maps, map building completing time t_f, the robot trajectories and corresponding probability q_0 are indicated in Fig. 5.30. It can be seen that, in the five experiments, the robot successfully avoided all the obstacles and built the complete maps of the closed area in a finite time. Moreover, according to the comparison of the experimental results, it can be seen that the map building completing time t_f and the length of the robot trajectory are influenced by different values of the parameter q_0. Furthermore, it is easy to see that the accuracy of the generated map is influenced by the length of the robot trajectory because a longer robot trajectory involves bigger odometry errors and reduces the accuracy of the map (e.g. Fig. 5.30(a)).

(a) $t = 10$s

(b) $t = 40$s

(c) $t = 100$s

(d) $t = 174.2$s
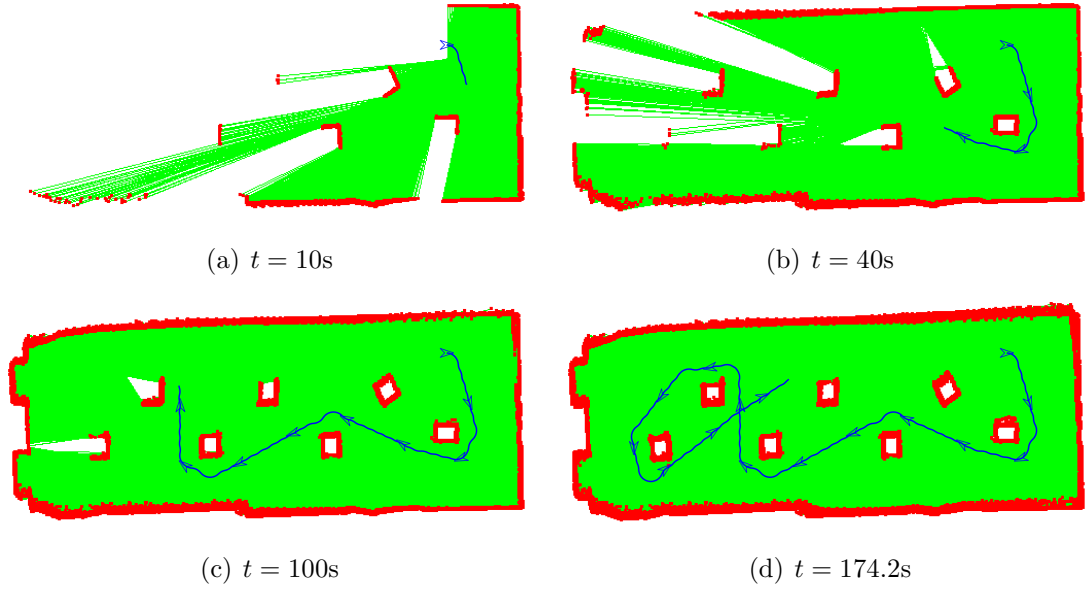
Figure 5.28: Generated maps at different time with probability $q_0 = 0.6$.

(a) $t = 51$s

(b) $t = 97$s

(c) $t = 143$s

(d) $t = 174.2$s
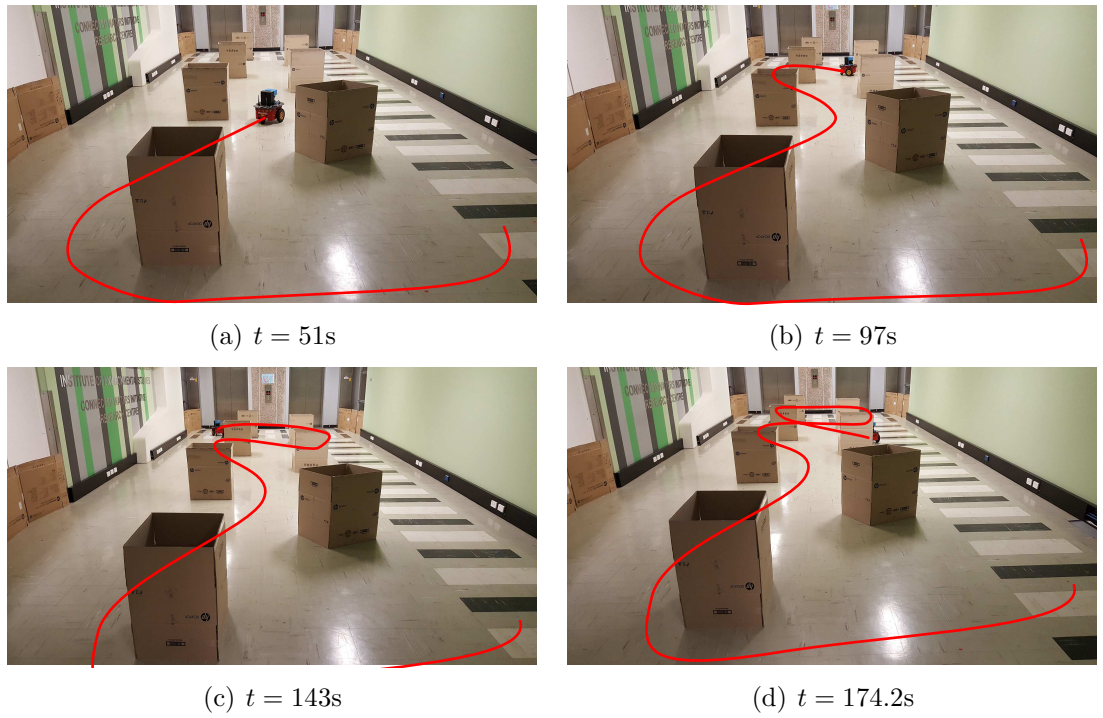
Figure 5.29: Experiment pictures at different time with probability $q_0 = 0.6$.
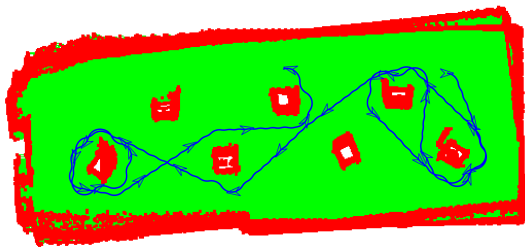
(a) $q_0 = 0.3$ and $t_f = 313.6$s
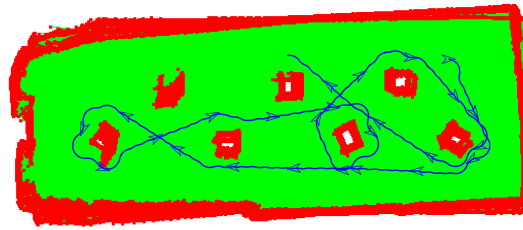
(b) $q_0 = 0.4$ and $t_f = 298.6$s
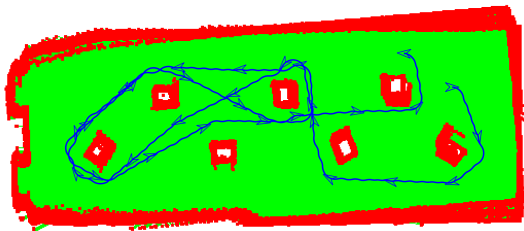
(c) $q_0 = 0.5$ and $t_f = 336.2$s
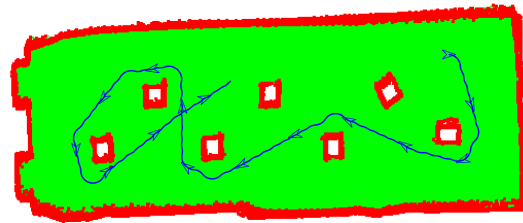
(d) $q_0 = 0.6$ and $t_f = 174.2$s
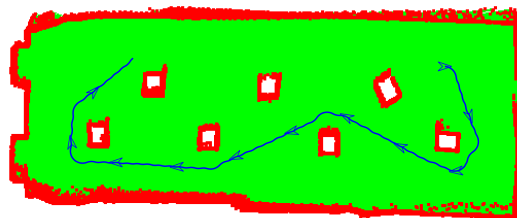
(e) $q_0 = 0.7$ and $t_f = 130.2$s

Figure 5.30: Generated maps with different probability q_0 and different map building completing time t_f.

5.5 Summary

In this chapter, a safe area search and map building algorithm for a mobile ground robot in a closed unknown 2D environment with obstacles was proposed. The obstacles are not required to be convex. The developed algorithm belongs to the class of randomized algorithms. It was proved that with probability 1 the robot finishes searching the area and builds a complete map of the environment in a finite time. Moreover, the odometry errors, range finder noise and dynamic environments are considered and discussed in the computer simulations.

The proposed method has a number of advantages. First, the area search and safe navigation were combined together in our method. Second, the proposed algorithm is relatively simple and computationally efficient. Moreover, unlike many other publications in the area, a non-holonomic robot model was considered in the proposed method. Finally, the performance of the proposed algorithm is proved by mathematically rigorous analysis. The usefulness of the method in real applications is based on the fact that the proposed algorithm is computationally simple and easy to implement. Moreover, the proposed mathematically rigorous theoretical framework describes cluttered environments in which a good performance of the algorithm will be guaranteed.

While the simulations and experiments have confirmed the performance and advantages of the proposed algorithm, there are still some limitations on our method. In particular, in the proposed method, only the range finder sensor is considered. In the future work, the proposed methodology will be generalized and extended to different types of sensors. Moreover, the algorithm will be extended to 3D environments. An interesting direction of our future research will be to find a method to select a better value of the parameter q_0 in the step **A3** of the algorithm depending on the environment to be explored. Implementation of the proposed method in real applications will be another important direction of our future research. Potential real applications include environment map building in planetary development, seabed map building in oceans, and surveillance and patrolling in sensitive and

contaminated regions.

Chapter 6

Safe Area Search and 3D Map Building of a Ground Mobile Robot in Indoor Environments

This chapter is based on the the publications [156] and [155]. In this chapter, we improve the area search and map building algorithm presented in Chapter 5. The map building problem is extended into 3D environment in this chapter. In this chapter, we consider a non-holonomic ground mobile robot equipped with two 2D range finder sensors. One range finder sensor is used to detect the obstacles on the ground plane and another range finder sensor is used to capture and scan the 3D structure of an indoor environment above the ground plane. In our method, we propose a probabilistic navigation algorithm that performs a safe area search and exploration in the unknown indoor environment and builds a 3D map of the environment in a finite time. The 3D map can be presented by an octree and indicates the occupied area and unoccupied area in the indoor environment. The control law we present in this method consists of switching between different control strategies with different conditions to construct a hybrid system; see [102, 152, 159, 175].

6.1 Problem description

The Robot model we considered is the unicycle model presented in Chapter 2; see Equation (2.1). In this chapter, $(x(t), y(t))$ is the Cartesian coordinates of the vehicle and $\theta(t)$ is the robot's heading at time t. The robot's minimum turning radius is

$$R_{\min} = \frac{v_r}{u_M}. \tag{6.1}$$
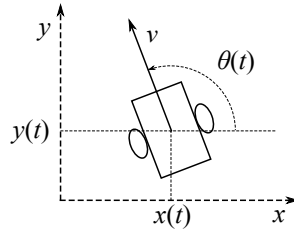
Figure 6.1: Unicycle model of the robot.

The wheeled robot travels on the flat ground of an unknown indoor environment $\mathcal{V} \subset \mathbb{R}^3$ (see Fig. 6.2). The boundary $\partial \mathcal{V}$ of the set $\mathcal{V}$ consists of the ground and the boundaries of several obstacles, like furnitures and walls. The flat ground can be modelled as a closed and bounded planar unknown area $\mathcal{A}$ with several disjoint planar obstacles $D_1, \ldots, D_k$ (see Fig. 6.3). Let $d_s > 0$ be a given safety margin. The requirement is to drive the ground mobile robot through the collision-free part of the area $\mathcal{A}$ while keeping the safety margin d_s from both the boundaries of area $\mathcal{A}$ and obstacles $D_1, \ldots, D_k$ on the plane of the ground.
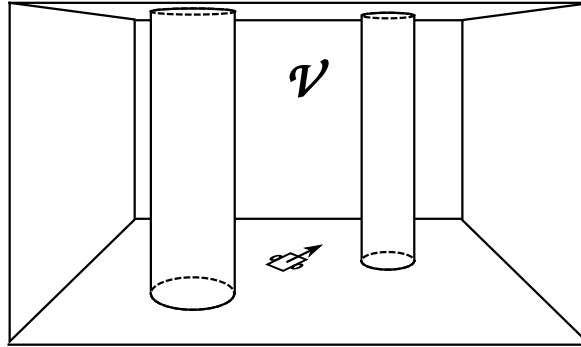
Figure 6.2: Closed indoor environment $\mathcal{V}$.
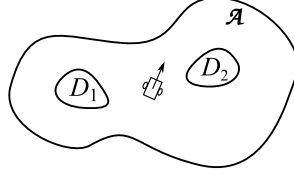
Figure 6.3: Planar area $\mathcal{A}$ with obstacles.

Definition 6.1.1 *Area $\mathcal{V}$ is a closed, connected and bounded point set in $\mathbb{R}^3$.*

Remark 6.1.1 *The boundary $\partial \mathcal{V}$ of $\mathcal{V}$ is a closed surface that any point on $\partial \mathcal{V}$ belongs to an obstacle or the ground. Other points inside $\mathcal{V}$ are unoccupied points.*

Definition 6.1.2 *The area $\mathcal{A}$ and obstacle D_i for any i are closed, connected and bounded point sets in $\mathbb{R}^2$.*

Available Measurements: The robot is equipped with two 2D range finder sensors. The first range finder sensor **S1** has a scanning range of 360° and is mounted on the robot's centre with the scanning plane parallel to the flat ground. At any time $t \geq 0$, it measures the distance to the nearest object in all directions on the ground; i.e. for any angle $0 \leq \omega \leq 2\pi$, the sensor measures the distance $d_h(\omega, t)$ to the nearest boundary of an obstacle or the area $\mathcal{A}$ in the direction given by ω on the ground (see Fig. 6.4). The second range finder sensor **S2** has a scanning range of 180° and is mounted on the robot's centre with the scanning plane orthogonal to the flat ground and the robot's heading. At any time $t \geq 0$, it measures the distance to the nearest object on the sensor's scanning plane above the ground; i.e. for any angle $0 \leq \omega \leq \pi$ on the scanning plane, the sensor measures the distance $d_v(\omega, t)$ to the nearest boundary of an obstacle in the direction given by ω above the ground (see Fig. 6.5). Furthermore, the robot is equipped with an odometry type sensor which measures the robot's location and heading relative to the initial location and heading.

The objective of the robot is to travel through the collision-free part of the unknown area $\mathcal{A}$ and build a complete 3D map of the unknown indoor environment $\mathcal{V}$.
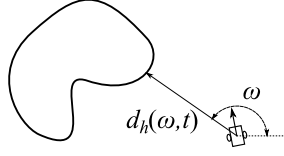
Figure 6.4: Distance $d_h(\omega, t)$ measurement by the first range finder sensor **S1**.
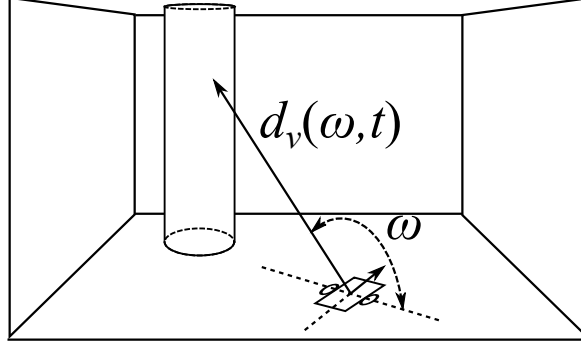
Figure 6.5: Distance $d_v(\omega, t)$ measurement by the second range finder sensor **S2**.

Definition 6.1.3 *Let $\mathcal{M}$ be a point set in $\mathbb{R}^3$ and $\mathcal{F}$ be a binary map $\mathcal{M} \to \{0, 1\}$. The pair $(\mathcal{M}, \mathcal{F})$ is said to be a complete map of the area $\mathcal{V}$ if $\mathcal{M}$ and $\mathcal{V}$ are congruent, i.e. one can be transformed into the other by an isometry, and for any point $p \in \mathcal{M}$, $\mathcal{F}(p) = 1$ if and only if p corresponds to a point of the boundary $\partial\mathcal{V}$ under such isometry.*

The complete map $(\mathcal{M}, \mathcal{F})$ indicates the unoccupied parts and the boundary of the closed indoor environment. To build the complete map, we are here to propose the following map building algorithm, which is a modification of a map building algorithm in 2D planar environment proposed in [157].

A1: At any time $t \geq 0$, the robot builds a planar set $\mathcal{M}_0(t)$ and a binary map $\mathcal{F}_0(t)$ as follows: the set $\mathcal{M}_0(t)$ is formed by all the points p that can be seen by the range finder sensor **S2** and $\mathcal{F}_0(t)(p) = 1$ if p belongs to either obstacle's boundary or the ground; otherwise, $\mathcal{F}_0(t)(p) = 0$ (see Fig. 6.6). The pair $(\mathcal{M}_0(t), \mathcal{F}_0(t))$ is called instant map at time t.

A2: At any time $t \geq 0$, the robot builds a set $\mathcal{M}(t) \subset \mathbb{R}^3$ and a binary map $\mathcal{F}(t)$ by fusing all the instant maps $(\mathcal{M}_0(\tau), \mathcal{F}_0(\tau))$ for $\tau \in [0, t]$ with the help of

odometry sensor. The pair $(\mathcal{M}(t), \mathcal{F}(t))$ is called the total map at time t.
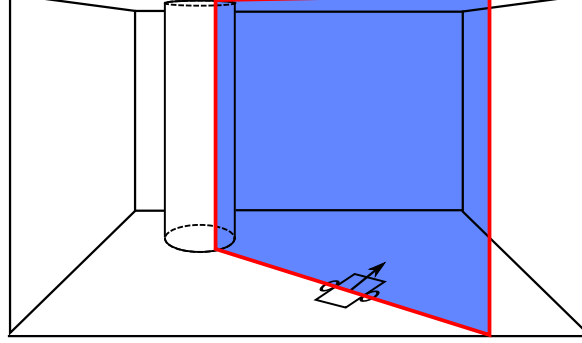
Figure 6.6: Instant map $(\mathcal{M}_0(t), \mathcal{F}_0(t))$.

Now, we are here to give some definitions as follows.

Definition 6.1.4 *Consider a robot trajectory $p(t) = (x(t), y(t)), t \in [0, t_f]$ with the map building algorithm **A1**–**A2**. The time $t_f > 0$ is called a map building completing time if for all the points p on the boundary of the set $\mathcal{M}(t_f)$, $\mathcal{F}(t_f)(p) = 1$. It means the boundary of the set $\mathcal{M}(t_f)$ does not include any point p such that $\mathcal{F}(t_f)(p) = 0$ (see Fig. 6.7).*
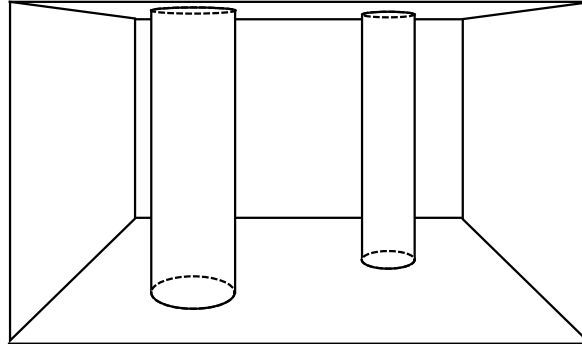
Figure 6.7: Complete map $(\mathcal{M}(t_f), \mathcal{F}(t_f))$. For all points p on the boundary of the set $\mathcal{M}(t_f)$, $\mathcal{F}(t_f)(p) = 1$. For other points in the set $\mathcal{M}(t_f)$, $\mathcal{F}(t_f)(p) = 0$.

If a robot completes the map building at the map building completing time t_f, the map $(\mathcal{M}(t_f), \mathcal{F}(t_f))$ is a complete map showing the whole unoccupied parts of the area $\mathcal{V}$ and the obstacles.

Definition 6.1.5 *Let t_f be a map building completing time for a robot trajectory $p(t) = (x(t), y(t)), t \in [0, t_f]$. The trajectory is said to be collision-free complete map building if the distance between the robot's position $p(t)$ and the boundary of $\mathcal{A}$ or D_i is greater than or equal to d_s for any $t \in [0, t_f]$.*

Notation 6.1.1 *Let p be an arbitrary point in the area $\mathcal{A}$. For any closed set $D \subset \mathcal{A}$, the minimum distance $\rho(D, p)$ between p and D is introduced as*

$$\rho(D, p) := \min_{q \in D} \|p - q\|. \tag{6.2}$$

Definition 6.1.6 *For the safety margin $d_s > 0$, the d_s-enlargement of a closed set $D \subset \mathbb{R}^2$ is a set $\mathcal{E}[D, d_s]$ defined as follows:*

$$\mathcal{E}[D, d_s] := \{p \in \mathbb{R}^2 : \rho(D, p) \le d_s\}. \tag{6.3}$$

Definition 6.1.7 *For the safety margin $d_s > 0$, the d_s-reduction of a closed set $D \subset \mathbb{R}^2$ is a set $\mathcal{R}[D, d_s]$ defined as follows:*

$$\mathcal{R}[D, d_s] := \{p \in D : \rho(\partial D, p) \ge d_s\}, \tag{6.4}$$

where ∂D is the boundary of set D.

Assumption 6.1.1 *The planar sets $\mathcal{R}[\mathcal{A}, d_s]$ and $\mathcal{E}[D_i, d_s]$, $i = 1, \ldots, k$ are closed, bounded, connected and linearly connected sets.*

Assumption 6.1.2 *The sets $\mathcal{E}[D_i, d_s]$ and $\mathcal{E}[D_j, d_s]$ do not overlap for any $i \ne j$. Any set $\mathcal{E}[D_i, d_s]$ is a subset of $\mathcal{R}[\mathcal{A}, d_s]$.*

If Assumption 6.1.2 does not hold, the robot may not be able to reach some parts of the planar area $\mathcal{A}$ with obstacle avoidance and build the complete map of the area $\mathcal{V}$.

Notation 6.1.2 *Let $\partial D_i(d_s)$ denote the boundary of the set $\mathcal{E}[D_i, d_s]$, and $\partial \mathcal{A}(d_s)$ denote the boundary of the set $\mathcal{R}[\mathcal{A}, d_s]$.*

Assumption 6.1.3 *All boundaries $\partial D_i(d_s)$ and $\partial \mathcal{A}(d_s)$ are non-self-intersecting, closed smooth curve. The absolute value of the curvature at any point of $\partial D_i(d_s)$ or $\partial \mathcal{A}(d_s)$ is smaller than or equal to $\frac{1}{R_{\min}}$.*

Definition 6.1.8 *The robot has two initial circles at the robot's initial position. The initial circles are tangent to the robot's initial heading θ_0 and cross the robot's initial position (x_0, y_0). The radius of each initial circle is equal to $R_{\min}$ (see Fig. 6.8).*
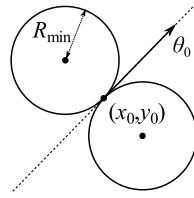
Figure 6.8: The two initial circles.

Assumption 6.1.4 *Both of the two initial circles lie in the set $\mathcal{R}[\mathcal{A}, d_s]$ and do not intersect with any set $\mathcal{E}[D_i, d_s]$.*

6.2 Safe navigation algorithm

Here, we propose a collision-free navigation algorithm for the mobile robot to explore the planar area $\mathcal{A}$ and build a complete map of area $\mathcal{V}$. Firstly, several definitions and assumptions are introduced.

Definition 6.2.1 *Let L be a straight line segment which lies in the set $\mathcal{R}[\mathcal{A}, d_s]$ and does not cross any set $\mathcal{E}[D_i, d_s]$, $\bar{L}$ be the straight line which coincides with L. The straight line segment L is said to be a common tangent line segment if it satisfies any one of the following conditions:*

1. *$\bar{L}$ is simultaneously tangent to $\partial D_i(d_s)$ and $\partial D_j(d_s)$, $i \neq j$ at the two distinct end points of L.*

2. *$\bar{L}$ is tangent to $\partial D_i(d_s)$ at two distinct end points of L.*

3. $\bar{L}$ is simultaneously tangent to $\partial D_i(d_s)$ and $\partial \mathcal{A}(d_s)$ at the two distinct end points of L.

4. $\bar{L}$ is simultaneously tangent to $\partial D_i(d_s)$ and an initial circle at the two distinct end points of L.

5. $\bar{L}$ is simultaneously tangent to $\partial \mathcal{A}(d_s)$ and an initial circle at the two distinct end points of L.

6. $\bar{L}$ is tangent to $\partial \mathcal{A}(d_s)$ at two distinct end points of L.

Assumption 6.2.1 *Any end point of a common tangent line segment does not belong to other distinct common tangent line segments.*

Now, we are here to introduce a planar graph $\mathcal{G}$ as follows.

Definition 6.2.2 *The graph $\mathcal{G}$ is defined as follows. The vertices of $\mathcal{G}$ are the robot's initial position (x_0, y_0) and the end points of all the common tangent line segments. The edges of $\mathcal{G}$ are all the common tangent line segments, arcs of the initial circles and the segments of all the boundaries $\partial D_i(d_s)$ and $\partial \mathcal{A}(d_s)$. The edges connects all the vertices (see Fig. 6.9).*
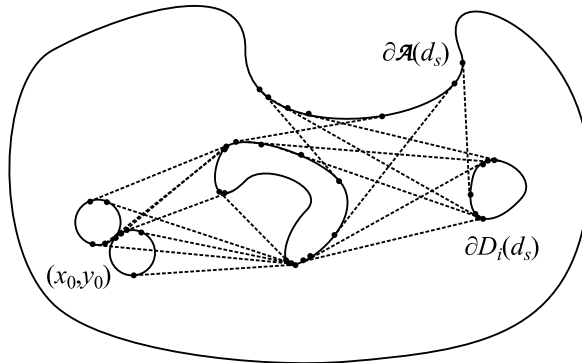
Figure 6.9: Graph $\mathcal{G}$ with vertices and edges.

With the presented graph $\mathcal{G}$, some definitions and assumptions are proposed as follows.

Definition 6.2.3 *Let L be a common tangent line segment on graph $\mathcal{G}$. For any end point p of L, the direction from p to another end point of L is called the exit direction of the vertex p on graph $\mathcal{G}$ (see Fig. 6.10).*
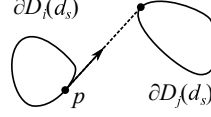
Figure 6.10: Exit direction of the vertex p.

Definition 6.2.4 *A path on the graph $\mathcal{G}$ connecting two vertices is said to be viable if the heading at the end of each edge of the path coincides with the heading at the beginning of the next edge of the path (see Fig. 6.11).*
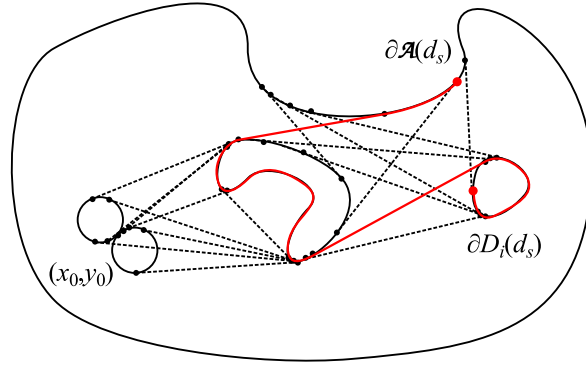
Figure 6.11: A viable path on graph $\mathcal{G}$.

Assumption 6.2.2 *Let p_1 and p_2 be any two distinct vertices on any boundary $\partial D_i(d_s)$ or $\partial \mathcal{A}(d_s)$. From the exit direction of p_1 or another direction opposite to the exit direction of p_1, there exists at least one viable path from p_1 and p_2 on the graph $\mathcal{G}$.*

Definition 6.2.5 *A vertex on the boundary $\partial D_i(d_s)$ or $\partial \mathcal{A}(d_s)$ is said to be non-circular if another vertex on the corresponding common tangent line segment does not belong to the initial circles. Otherwise, the vertex is said to be circular.*

Definition 6.2.6 *If the boundary $\partial \mathcal{A}(d_s)$ is convex, the edges $\partial D_i(d_s)$ and the com-mon tangent line segments which do not include circular vertex are called map build-ing edges. If $\partial \mathcal{A}(d_s)$ is non-convex, $\partial \mathcal{A}(d_s)$ is also called map building edge. See Fig. 6.12.*
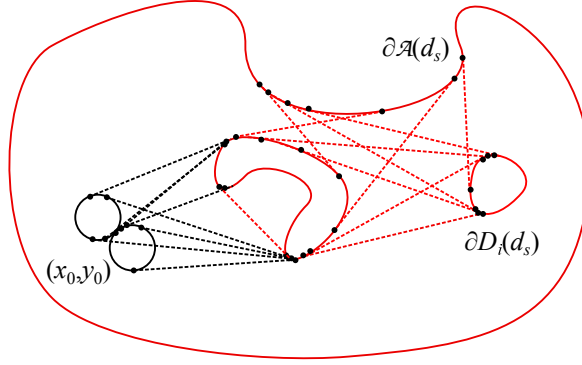
Figure 6.12: Map building edges on graph $\mathcal{G}$.

For the case of simplicity, the following assumption is introduced.

Assumption 6.2.3 *On the graph $\mathcal{G}$, the set of map building edges is not empty.*

Assumption 6.2.4 *For any point $p \in \partial \mathcal{V}$, there exists at least a point q on map building edges of the graph $\mathcal{G}$ such that the straight line segment between p and q belongs to the set $\mathcal{V}$, does not cross the boundary $\partial \mathcal{V}$ and is orthogonal to the edge on point q.*

Assumption 6.2.3 and 6.2.4 guarantees that any point on the boundary $\partial \mathcal{V}$ of the environment $\mathcal{V}$ can be detected by the sensor **S2** when the robot searches all the map building edges on the graph $\mathcal{G}$.

Let $q_c \in (0, 1)$ be a given probability. To explore the ground area $\mathcal{A}$ and build the complete map of the indoor environment $\mathcal{V}$, a probabilistic navigation algorithm is introduced as follows:

B1: The robot starts to move along one of two initial circles.

B2: When the robot moving along an initial circle reaches a vertex, which is not the robot's initial position, with the heading direction coinciding with the

exit direction of the vertex, the robot starts to move along the corresponding common tangent line segment.

B3: When the robot moving along a common tangent line segment reaches a vertex, the robot starts to move along the corresponding boundary $\partial D_i(d_s)$ or $\partial \mathcal{A}(d_s)$ which the vertex belongs to.

B4: When the robot moving along a boundary $\partial D_i(d_s)$ or $\partial \mathcal{A}(d_s)$ reaches a non-circular vertex with the heading direction coinciding with the exit direction of the vertex, with the probability q_c the robot starts to move along the corresponding common tangent line segment and with the probability $(1 - q_c)$ the robot continues to move along the boundary.

Now, we are here to propose the main theoretical result as follows.

Theorem 6.2.1 *Suppose that Assumptions 6.1.1–6.1.4, 6.2.1–6.2.4 hold, the robot is navigated by the algorithm **B1**–**B4** with probability $q_c \in (0, 1)$ and, simultaneously, builds the map by the algorithm **A1**–**A2**. Then for any initial position and heading of the robot on the ground of the indoor environment, with probability 1 there exists a time $t_f > 0$ such that the trajectory $p(t) = (x(t), y(t)), t \in [0, t_f]$ of the robot (2.1) is collision-free complete map building.*

Proof of Theorem 6.2.1: According to Assumption 6.2.3 and the algorithm **B1**–**B4**, the robot starts from its initial position, moves along an initial circle and a common tangent line segment to a vertex belonging to a map building edge. Then the robot travels along the map building edges on the graph $\mathcal{G}$. According to Assumption 6.2.1 and 6.2.2, with a non-zero probability the robot can reach any vertex b on the map building edges from any other vertex a on the map building edges with any starting direction on vertex a. Therefore, it is easy to known that there exist a time $t_f > 0$ such that the robot has travelled all the points on any map building edge.

According to Assumption 6.2.4, it can be seen that the time t_f is a map building completing time. According to Assumption 6.1.4, the robot moves along any initial

circle while keeping the given safety margin d_s. According to Definition 6.2.1, the distance between any point on any common tangent line segment on the graph $\mathcal{G}$ and any obstacle D_i or the boundary of $\mathcal{A}$ is $\geq d_s$. According to Assumption 6.1.1 and 6.1.2, it is easy to known that the robot moving along the boundaries $\partial D_i(d_s)$ and $\partial \mathcal{A}(d_s)$ also keeps the given safety margin d_s. Therefore, with the navigation of the algorithm **B1**–**B4**, the distance between the robot and any obstacle D_i or the boundary of $\mathcal{A}$ is $\geq d_s$ at any time. The Assumption 6.1.3 guarantees the trajectory of the robot satisfies the non-holonomic constraint (6.1).

According to the above proof. It can be indicated that the robot's trajectory $p(t) = (x(t), y(t)), t \in [0, t_f]$ of the robot model (2.1) is collision-free complete map building. This completes the proof of Theorem 6.2.1.

6.3   Computer simulations

We carry out computer simulations to confirm the proposed map building algorithm **A1**–**A2** and navigation algorithm **B1**–**B4**.

To apply the proposed navigation algorithm **B1**–**B4** on the robot, a sliding mode control law which is a modification of the law of [153] is used, that consists of switching between a boundary following approach proposed in [107], and the pure pursuit navigation approach; see e.g. [161]. The control law is described as follows with three separate modes $R1$, $R2$ and $R3$:

$$u(t) = \begin{cases} \pm u_M & R1 \\ \operatorname{sgn}\left[\phi_m(t)\right] u_M & R2 \\ \Gamma \operatorname{sgn}\left[\dot{d}_m(t) + H(d_m(t) - d_s)\right] u_M & R3 \end{cases}, \qquad (6.5)$$

where the sign function $\operatorname{sgn}(x)$ is defined as follows:

$$\operatorname{sgn}(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases} \qquad (6.6)$$

and the saturation function $H(x)$ is defined as follows with two tunable parameters l and r:

$$H(x) = \begin{cases} lx & |x| < r \\ lr\,\mathrm{sgn}(x) & \text{otherwise} \end{cases}. \tag{6.7}$$

When mode $R1$ is active, the robot is moving along an initial circle. Mode $R1$ is the initial mode. When mode $R2$ is active, the robot is moving along a common tangent line segment. When mode $R3$ is active, the robot is moving along a boundary $\partial D_i(d_s)$ or the boundary $\partial \mathcal{A}(d_s)$. The three modes $R1$–$R3$ corresponds to the above three separate control strategies and transitions to each other by the following mode transition rules:

$R1 \rightarrow R2$: It occurs when the robot is at a vertex on one of the initial circles and the exit direction at the vertex coincides with the robot's current heading.

$R2 \rightarrow R3$: It occurs when the robot is at a vertex on a common tangent line segment and reaches the boundary $\partial D_i(d_s)$ or $\partial \mathcal{A}(d_s)$.

$R3 \rightarrow R2$: With the probability q_c, it occurs when the robot is at a vertex on the boundary $\partial D_i(d_s)$ or $\partial \mathcal{A}(d_s)$ and the exit direction at the tangent point coincides with the robot's current heading.

In the control law (6.5), when mode $R3$ is active, the variable $d_m(t)$ is the minimum distance from the robot to the obstacles $D_1, \ldots, D_k$ and the area boundary $\partial \mathcal{A}$ on the ground plane. It is measured directly by the range finder sensor **S1**. When mode $R2$ is active, the variable $\phi_m(t)$ is the angle between the robot's heading $\theta(t)$ and the bearing of the end point which the robot is moving forwards from the robot. It can be calculated with the help of the odometry sensor and the range finder sensor **S1**. When mode $R3$ is activated at a vertex on boundary $\partial D_i(d_s)$ or $\partial \mathcal{A}(d_s)$, the variable Γ is defined as $+1$ if the boundary is on the left of the robot, -1 if it is on the right. It can be calculated with the help of the range finder sensor **S1**. The robot stops at the map building completing time t_f. The control law (6.5) consisting of switching constructs a hybrid control system; see [102, 152, 159, 175].

Here, an approach is proposed to determine if the robot's heading coincides with the exit direction of a vertex. Suppose the range finder sensor **S1** measures distances $d_h(\omega_i, t)$ in finite directions ω_i, $i = 0, \pm 1, \ldots, \pm n$, where ω_0 is the robot's heading and the angular intervals between any two successive direction ω_i and ω_{i+1} are equal and small. Then the steps of calculating tangent line segment on boundary $\partial D_i(d_s)$ or $\partial \mathcal{A}(d_s)$ crossing the robot's position are described as follows:

1. Search any direction pairs ω_i and ω_{i+1} satisfying $|(d_h(\omega_i, t) - d_h(\omega_{i+1}, t)| \geq 2d_s$. Between the pairs ω_i and ω_{i+1} satisfying the inequality, denote the direction corresponding to the shorter distance by $\hat{\omega}_j$, $j = 1, 2, \ldots, m$.

2. For any j, let $d_h(\hat{\omega}_j, t)$ be a leg and construct a right triangle. The length of another leg is d_s sharing the common end point with $d_h(\hat{\omega}_j, t)$ on measurement point. Then the hypotenuse denoted by ξ_j can approximately represent the tangent line segment on the corresponding boundary $\partial D_i(d_s)$ or $\partial \mathcal{A}(d_s)$ (see Fig. 6.13). Notice that if there exist any other measurement point p such that the minimum distance between point p and line segment ξ_j is smaller than d_s, ξ_j cannot represent the tangent line segment since it cannot be tracked with the safety margin d_s (see Fig. 6.14).
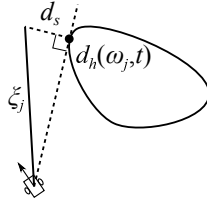
Figure 6.13: Tangent line segment ξ_j.
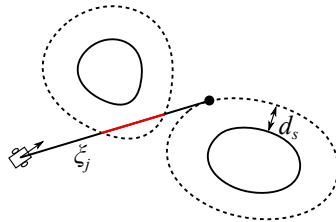
Figure 6.14: Unsafe tangent line segment ξ_j.

If there exist a line segment ξ_j such that the angle between direction ω_0 and line segment ξ_j is smaller than a threshold, ξ_j is said to coincide with the robot's heading. When mode transition $R1 \rightarrow R2$ occurs, the end point of the corresponding tangent line segment ξ_j is defined as a intermediate target and is used to calculated the variable $\phi_m(t)$.

In the following part, several computer simulations are carried out to confirm the performance of the proposed algorithms. To avoid the oscillation of the robot's heading involved by sign function $\mathrm{sgn}(x)$, saturation function with upper bound 1, lower bound -1 and a suitable gradient is used to replace the sign function $\mathrm{sgn}(x)$. Moreover, when the robot is moving on mode $R3$ and decides not to activate mode $R2$, there is a following short time interval that the robot will not activate mode $R2$ in any case. It is used to avoid repeatedly checking the coinciding of the robot's heading with a same common tangent line segment. Moreover, the octree can be used to represent the 3D map of the environment.

In the presented computer simulation, an indoor environment was built with several obstacles with the size of $25\mathrm{m} \times 25\mathrm{m} \times 3\mathrm{m}$. A ground mobile robot moved on the flat ground of the room and was equipped with two laser range finder sensors. The mobile robot was navigated by the proposed algorithm **B1**–**B4** in the indoor environment with obstacle avoidance. At a map building completing time, the robot stopped and completed the map building. The trajectory of the robot is shown in Fig. 6.15 that the robot kept the safety margin from the boundary of the room and any other obstacles. Fig. 6.16 shows the map building of the robot while the robot was moving. Finally, the robot stopped with a complete 3D map of the indoor environment indicated in Fig. 6.17. The control input satisfies the bound constraint.

6.4 Experiments with real mobile robot

We carry out an experiment with a real ground mobile robot. In the experiment, a Pioneer3-DX robot is used and there are two SICK LMS-200 laser range finders mounted on the robot (see Fig. 6.18). In a closed room, there were some folding
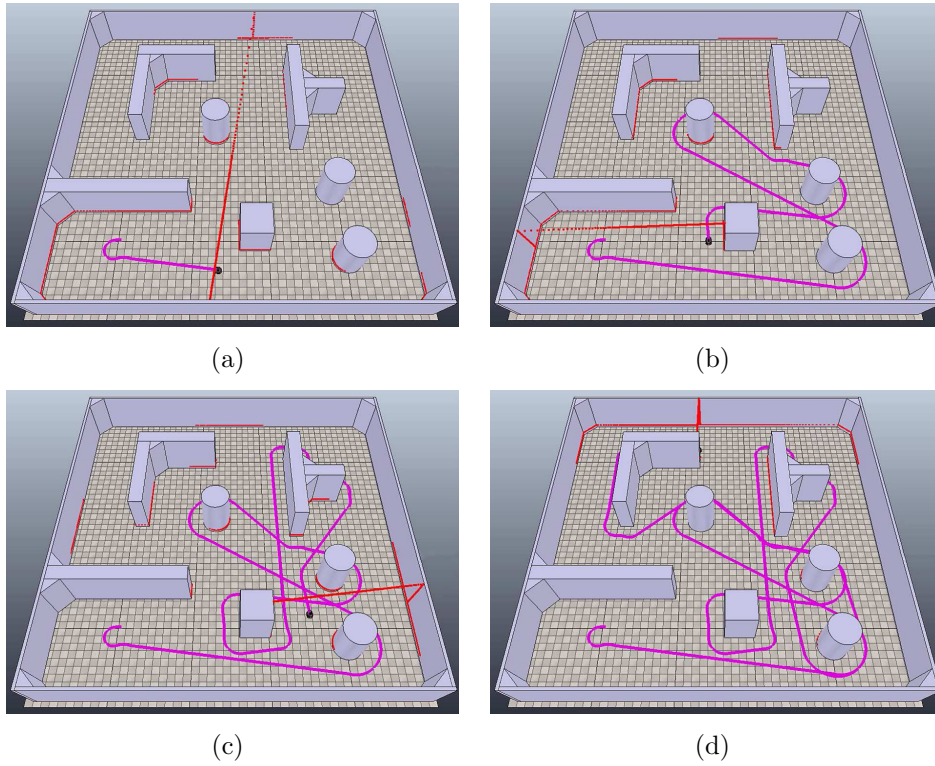
Figure 6.15: Robot trajectory in computer simulation.
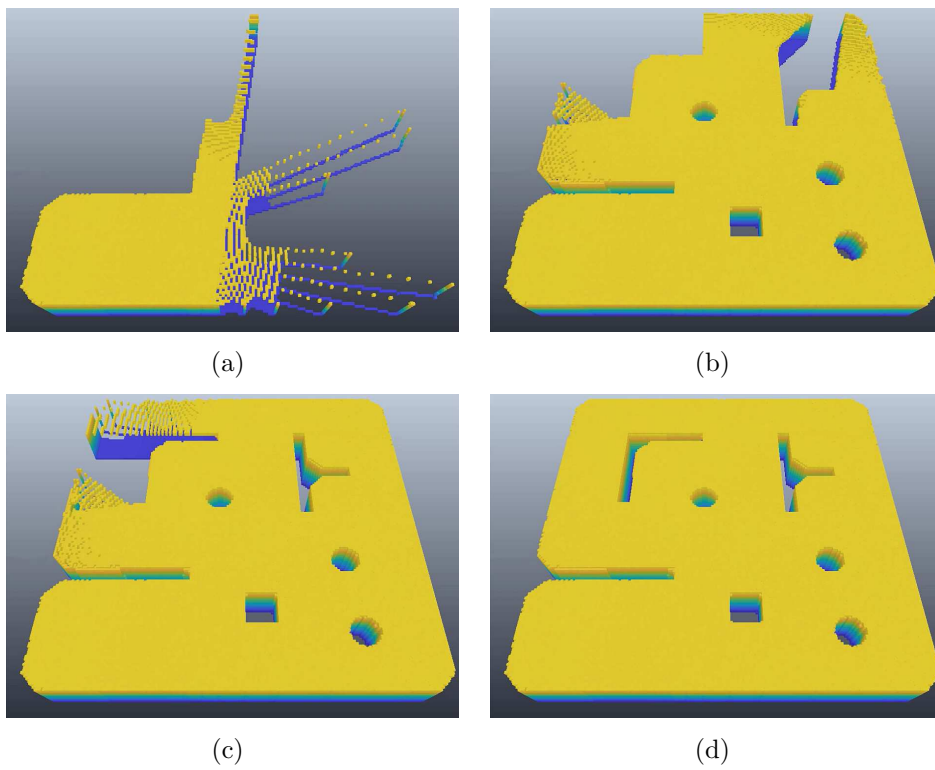
Figure 6.16: Map building in computer simulation. Different colors represent different height.
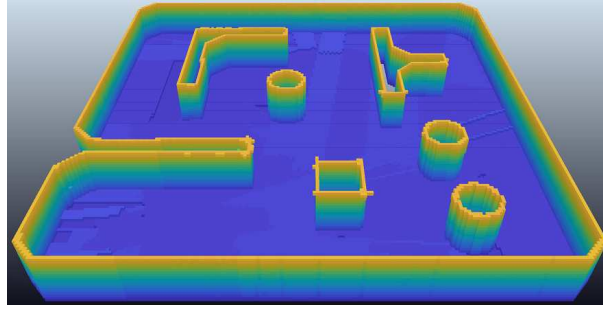
Figure 6.17: Complete 3D map of the environment in computer simulation.

cartons put on the ground as the obstacles. The robot started from an initial circle and was navigated by the proposed algorithm to avoid collision with any obstacle. Simultaneously, the robot scanned the 3D environment by the second laser range finder and used the measurements to build the 3D map of the environment (see Fig. 6.19 and 6.20). Finally, the robot stopped when the complete map was built at the map building completing time (see Fig. 6.21). The control input satisfies the bound constraint.
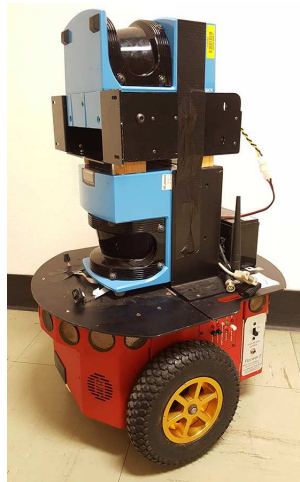
Figure 6.18: Pioneer3-DX robot with two SICK LMS-200 laser range finders.

6.5 Summary

In this chapter, we consider a non-holonomic ground mobile robot equipped with two 2D range finder sensors and propose a probabilistic safe navigation algorithm to search and explore an unknown indoor environment and build a complete 3D map.

(a)   (b)

(c)   (d)
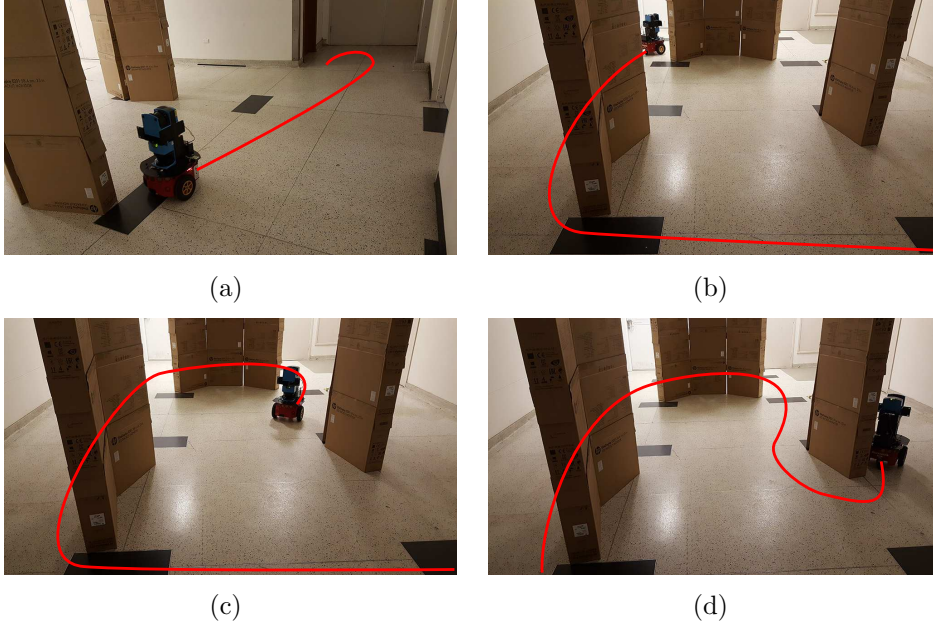
Figure 6.19: Pictures of the experiment.

(a)   (b)

(c)   (d)
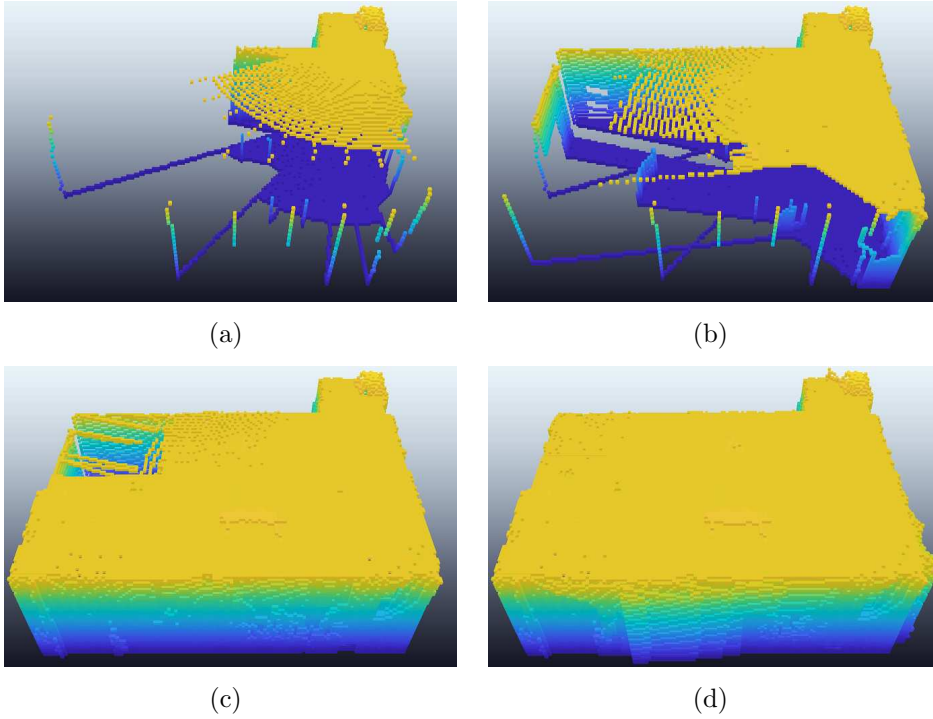
Figure 6.20: Map building of the environment in the experiment. Different colors represent different height.

The proposed method has several advantages. Firstly, the collision-free area search problem and complete 3D map building problem are solved simultaneously by the proposed algorithm. Secondly, comparing with the previously works, we consider the 3D map building of the environment, which is more useful than 2D map in
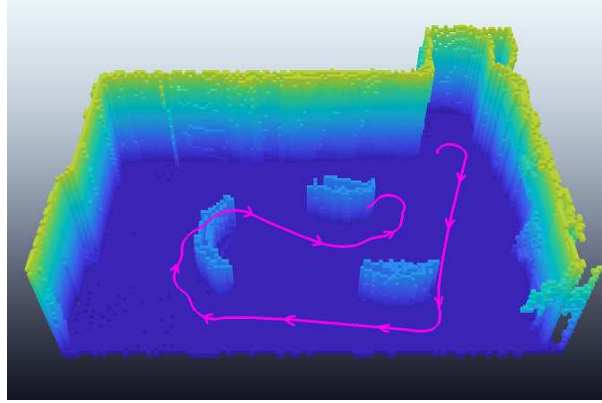
Figure 6.21: Complete 3D map of the environment and robot trajectory in the experiment.

practical implementations. Furthermore, the proposed navigation algorithm is relatively simple and fast. Therefore, it can be implemented in small-sized robots with low power supply and poor computer performance. Moreover, the non-holonomic constraint of the robot's motion is considered in the proposed algorithm, which is more practical than other works. According to the mathematical analysis, it can be proved that the complete 3D map can be built in a finite time by the ground mobile robot.

The proposed method can be used in several commercial and military tasks, like rescue and planetary exploration. In future work, flying robot can be considered for the 3D map building task.

Chapter 7

Conclusion and Future Work

In this chapter, we give the conclusion of this thesis and present the future work as follows.

7.1 Conclusion

In this thesis, we focus on two specific problems of robot navigation. In Chapter 1, the problem statements of the two specific problems are presented based on the literature review. One is the sensor-network based robot navigation problem and another is the area search and map building problem. In Chapter 2, the first stage of the design of sensor network based navigation algorithm is presented and an artificial potential field based path planning algorithm is designed. In Chapter 3, the presented path planning algorithm is developed and combined with a novel graph search algorithm to perform the real-time robot navigation. Furthermore, we extend the presented path planning algorithm into 3D environment with the probabilistic roadmap to navigate the micro flying robots in Chapter 4. Moreover, we study and present an area search and map building algorithm for ground mobile robot in a closed environment in Chapter 5. Based on the presented area sear algorithm, we develop another algorithm for 3D map building in Chapter 6.

Firstly, we present a path planning algorithm for ground mobile robots in dynamic environments. Then we consider the sensor network and develop the path

planning algorithm to a sensor network based navigation algorithm for ground mobile robots. In the presented method, a sensor network consists of several 2D range finder sensors. The sensor network detects and monitors any obstacles in dynamic environments. With the measurements, the sensor network guides each ground mobile robot from its initial position to the target with a relatively short trajectory by the navigation algorithm. The computer simulations and experiments with a real mobile robot confirm the performance of the presented method. Furthermore, we consider the implementations of small-sized flying robots in industries and extend the presented 2D navigation algorithm into 3D environments. In the 3D environments, the sensor network consists of several time-of-flight cameras. Each of the sensor nodes detects the dynamic obstacles in the field of view. The modified algorithm navigates the micro flying robots to avoid any detected obstacles and undetected areas.

Comparing with other navigation algorithms, the main feature of the presented sensor network based navigation algorithms is that the navigation task for any mobile robots in the workspace is transferred to the sensor network from the mobile robots. Each robot in the workspace with the presented sensor network is only required for a simple low-level path tracking controller. There is not any sensor for obstacle detection on the robots. It is a economical solution for multi-robot navigation in smart factories.

Furthermore, in this thesis, the safe area search and map building algorithms for single ground mobile robot is presented. Firstly, we consider a 2D map building and present a collision-free area search navigation algorithm for a non-holonomic ground mobile robot to explore a 2D bounded area. Simultaneously, the robot builds a complete 2D map of the unknown 2D area that indicates the boundary of the bounded area and the obstacles in the area. Then we consider an extra 2D range finder sensor mounted on the robot and develop the algorithm for 3D map building. The computer simulations and experiments confirm the performance of the presented area search and map building algorithms. The presented algorithms

belong to probabilistic navigation algorithm and the mathematical analysis proves that with probability 1 the robot can build a complete map in a finite time by the presented algorithms.

7.2   Future work

With the development of the sensor network and robotics, the mobile robots and sensor networks will be implemented widely in both industries and military. Therefore, the methods presented in this thesis can be applied in a variety of fields.

In Chapter 2, 3 and 4, we developed a sensor network based navigation framework for both ground mobile robots and flying robots. In this navigation framework, different micro-sized ground mobile robots and flying robots can be navigated with obstacle avoidance by the sensor network consisting of some 2D and 3D range finder sensors. In the future work of this topic, firstly, the completely distributed navigation strategies should be focused on with the presented sensor network. In a distributed navigation framework, a central computer is not used and each sensor node works individually and equally to perform the navigation for the mobile robots within its coverage, like other distributed sensor network [80, 132, 40, 151]. The main advantage of a completely distributed navigation algorithm is that the system can keep working when a part of the sensor nodes are faulty. Another interesting problem is the robot formation with collision avoidance in the presented sensor network framework. Mobile robot formation is a significant type of the multi-robot cooperation [46, 192, 160, 164, 165]. It can be combined with the presented sensor network based navigation framework in dynamic smart factories or indoor environments to complete some multi-robot cooperation tasks like object transport [7] with collision avoidance. Moreover, the presented sensor network based navigation algorithm can be combined with the target or source search problem [62, 88, 48] in dynamic indoor environments with moving obstacles. The target search combined with the safe indoor sensor network based navigation can be implemented in gas source or radioactive source searching in chemical plants or nuclear power plants

with obstacle avoidance.

In Chapter 5 and 6, we developed the collision-free area search and map building algorithms for a single ground mobile robot in 2D and 3D environments. With the presented algorithms, a non-holonomic ground mobile robot can be navigated with low computational cost to explore an unknown environment and build the complete 2D or 3D map. In the future work of this topic, collaborative multiple robot map building, like [169, 19, 141], can be considered based on the presented single robot area search and map building algorithm by combining the presented algorithm with some multi-robot cooperation strategies, such as [55, 18]. Another significant problem is the map building and area search on uneven terrain with obstacle avoidance. The path planning and navigation problem is more difficult than on uneven terrain than on a flat ground, such as [135, 134, 174]. Therefore, it can be a challenge to solve the safe area search and complete map building problem by combining the presented area search strategy with other uneven terrain navigation algorithms, such as [172, 60, 101]. Moreover, the presented research works can be combined with robust Kalman filtering approach [130] to build a more sophisticated map of the environment; see an example of the robust Kalman filtering approach in [128]. Furthermore, the presented area search and map building algorithm can be extended into 3D environments with flying robots; see [116]. A flying robot can be considered to search a 3D environment with complex structure safely and build a 3D map for the environment by a 2D or 3D range finder sensor. It can be implemented in many fields, such as indoor 3D structure scanning, cave exploration and tunnel rescue; see [177, 58, 59].

Bibliography

[1] F. Abdessemed, K. Benmahammed, and E. Monacelli. A fuzzy-based reactive controller for a non-holonomic mobile robot. *Robotics and Autonomous Systems*, 47(1):31–46, 2004.

[2] M. Al Khawaldah and A. Nüchter. Enhanced frontier-based exploration for indoor environment with multiple robots. *Advanced Robotics*, 29(10):657–669, 2015.

[3] M. S. Alam and M. U. Rafique. Mobile robot path planning in environments cluttered with non-convex obstacles using particle swarm optimization. In *2015 International Conference on Control, Automation and Robotics (IC-CAR)*, pages 32–36, Singapore, Singapore, 2015. IEEE.

[4] A. AlDahak, L. Seneviratne, and J. Dias. Frontier-based exploration for unknown environments using incremental triangulation. In *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–6, Linköping, Sweden, 2013. IEEE.

[5] G. Alenyà, S. Foix, and C. Torras. Using ToF and RGBD cameras for 3D robot perception and manipulation in human environments. *Intelligent Service Robotics*, 7(4):211–220, 2014.

[6] S. Almansa-Valverde, J. C. Castillo, and A. Fernández-Caballero. Mobile robot map building from time-of-flight camera. *Expert Systems with Applications*, 39(10):8835–8843, 2012.

[7] J. Alonso-Mora, S. Baker, and D. Rus. Multi-robot formation control and object transport in dynamic environments via constrained optimization. *The International Journal of Robotics Research*, 36(9):1000–1021, 2017.

[8] A. Babinec, F. Duchoň, M. Dekan, P. Pásztó, and M. Kelemen. VFH* TDT (VFH* with Time Dependent Tree): A new laser rangefinder based obstacle avoidance method designed for environment with non-static obstacles. *Robotics and autonomous systems*, 62(8):1098–1115, 2014.

[9] D. Bacciu, C. Gallicchio, A. Micheli, M. Di Rocco, and A. Saffiotti. Learning context-aware mobile robot navigation in home environments. In *IISA 2014, The 5th International Conference on Information, Intelligence, Systems and Applications*, pages 57–62, Chania Crete, Greece, 2014.

[10] N. Basilico and F. Amigoni. Exploration strategies based on multi-criteria decision making for searching environments in rescue operations. *Autonomous Robots*, 31(4):401–417, 2011.

[11] P. Beeson, J. Modayil, and B. Kuipers. Factoring the mapping problem: Mobile robot map-building in the hybrid spatial semantic hierarchy. *The International Journal of Robotics Research*, 29(4):428–459, 2010.

[12] F. Bohdanowicz, H. Frey, R. Funke, D. Mosen, F. Neumann, and I. Stojmenović. RSSI-based localization of a wireless sensor node with a flying robot. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, SAC '15, pages 708–715, New York, USA, 2015. ACM.

[13] J. Borenstein and L. Feng. Measurement and correction of systematic odometry errors in mobile robots. *IEEE Transactions on robotics and automation*, 12(6):869–880, 1996.

[14] P. Brass, I. Vigan, and N. Xu. Shortest path planning for a tethered robot. *Computational Geometry*, 48(9):732–742, 2015.

[15] J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems journal*, 4(1):25–30, 1965.

[16] H. Cao, J. Gao, F. Zhao, J. Zhao, C. Liu, Y. Liu, and X. Shi. The 3D map building of the mobile robot. In *2016 IEEE International Conference on Mechatronics and Automation*, pages 2576–2581, Harbin, China, Aug 2016.

[17] Y. Cao, W. Yu, W. Ren, and G. Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, 9(1):427–438, 2013.

[18] J. Capitan, M. T.J. Spaan, L. Merino, and A. Ollero. Decentralized multi-robot cooperation with auctioned POMDPs. *The International Journal of Robotics Research*, 32(6):650–671, 2013.

[19] S. Carpin. Fast and accurate map merging for multi-robot systems. *Autonomous Robots*, 25(3):305–316, Oct 2008.

[20] P. Chanak, I. Banerjee, J. Wang, and R. S. Sherratt. Obstacle avoidance routing scheme through optimal sink movement for home monitoring and mobile robotic consumer devices. *IEEE Transactions on Consumer Electronics*, 60(4):596–604, Nov 2014.

[21] C. K. Chang, C. Siagian, and L. Itti. Mobile robot vision navigation amp; localization using gist and saliency. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4147–4154, Taipei, Taiwan, 2010.

[22] H. Chang and T. Jin. Adaptive tracking controller based on the pid for mobile robot path tracking. In *Intelligent Robotics and Applications: 6th International Conference (ICIRA 2013)*, pages 540–549, Berlin, Heidelberg, 2013. Springer.

[23] Y. Cheng, P. Jiang, and Y. F. Hu. A snake-based scheme for path planning and control with constraints by distributed visual sensors. *Robotica*, 32(4):477–499, 2014.

[24] S. H. Cho and S. Hong. Map based indoor robot navigation and localization using laser range finder. In *2010 11th International Conference on Control Automation Robotics Vision*, pages 1559–1564, Singapore, Singapore, 2010.

[25] R. G. Colares and L. Chaimowicz. The next frontier: combining information gain and distance cost for decentralized multi-robot exploration. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 268–274, Pisa, Italy, 2016. ACM.

[26] M. A. Contreras-Cruz, V. Ayala-Ramirez, and U. H. Hernandez-Belmonte. Mobile robot path planning using artificial bee colony and evolutionary programming. *Applied Soft Computing*, 30:319–328, 2015.

[27] P. Corke, R. Peterson, and D. Rus. Networked robots: Flying robot navigation using a sensor net. In *Robotics Research. The Eleventh International Symposium*, pages 234–243, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[28] P. J. Costa, N. Moreira, D. Campos, J. Gonçalves, J. Lima, and P. L. Costa. Localization and navigation of an omnidirectional mobile robot: The robotfactory case study. *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, 11(1):1–9, Feb 2016.

[29] M. S. Couceiro, R. P. Rocha, and N. M. F. Ferreira. A novel multi-robot exploration approach based on particle swarm optimization algorithms. In *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, pages 327–332, Kyoto, Japan, Nov 2011.

[30] A. R. de Carvalho, A. D. Ribas, V. F. da Camara Neto, E. F. Nakamura, and C. M. Figueiredo. An RSSI-based navigation algorithm for a mobile robot in

wireless sensor networks. In *37th Annual IEEE Conference on Local Computer Networks*, pages 308–311, Clearwater Beach, USA, Oct 2012.

[31] N. Deshpande, E. Grant, M. Draelos, and T. C. Henderson. Received signal strength based bearing-only robot navigation in a sensor network field. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4618–4623, Chicago, USA, Sept 2014.

[32] N. Deshpande, E. Grant, and T. C. Henderson. Target localization and autonomous navigation using wireless sensor networks–a pseudogradient algorithm approach. *IEEE Systems Journal*, 8(1):93–103, March 2014.

[33] K. Dewangan, A. Saha, K. Vaiapury, and R. Dasgupta. 3D environment reconstruction using mobile robot platform and monocular vision. In *Advanced Computing and Communication Technologies: Proceedings of the 9th ICACCT, 2015*, pages 213–221, Singapore, 2016.

[34] N. Doh, H. Choset, and W. K. Chung. Accurate relative localization using odometry. In *IEEE International Conference on Robotics and Automation, 2003*, pages 1606–1612, Taipei, Taiwan, 2003. IEEE.

[35] N. L. Doh, H. Choset, and W. K. Chung. Relative localization using path odometry information. *Autonomous Robots*, 21(2):143–154, 2006.

[36] X. Dong, J. Xi, G. Lu, and Y. Zhong. Formation control for high-order linear time-invariant multiagent systems with time delays. *IEEE Transactions on Control of Network Systems*, 1(3):232–240, 2014.

[37] X. Dong, Y. Zhou, Z. Ren, and Y. Zhong. Time-varying formation control for unmanned aerial vehicles with switching interaction topologies. *Control Engineering Practice*, 46:26–36, 2016.

[38] L. E. Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79(3):497–516, 1957.

[39] G. Enriquez, S. Park, and S. Hashimoto. Wireless sensor network and RFID fusion approach for mobile robot navigation. *ISRN Sensor Networks*, 2013, 2013.

[40] G. L. Foresti and L. Snidaro. A distributed sensor network for video surveillance of outdoor environments. In *2002 International Conference on Image Processing*, volume 1, pages I–525–I–528, Rochester, USA, 2002.

[41] A. Franchi, L. Freda, G. Oriolo, and M. Vendittelli. A randomized strategy for cooperative robot exploration. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 768–774, Roma, Italy, April 2007.

[42] L. Freda and G. Oriolo. Frontier-based probabilistic strategies for sensor-based exploration. In *2005 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3881–3887, Hong Kong, China, 2005. IEEE.

[43] S. M. George, W. Zhou, H. Chenji, M. Won, Y. O. Lee, A. Pazarloglou, R. Stoleru, and P. Barooah. Distressnet: a wireless ad hoc and sensor network architecture for situation management in disaster response. *IEEE Communications Magazine*, 48(3):128–136, March 2010.

[44] M. Guerra, D. Efimov, G. Zheng, and W. Perruquetti. Finite-time obstacle avoidance for unicycle-like robot subject to additive input disturbances. *Autonomous Robots*, pages 1–12, 2016.

[45] M. S. Güzel, M. Kara, and M. S. Beyazkılıç. An adaptive framework for mobile robot navigation. *Adaptive Behavior*, 25(1):30–39, 2017.

[46] H. M. Güzey, T. Dierks, S. Jagannathan, and L. Acar. Hybrid consensus-based control of nonholonomic mobile robot formation. *Journal of Intelligent & Robotic Systems*, Apr 2017.

[47] D. Hai, Y. Li, H. Zhang, and X. Li. Simultaneous localization and mapping of robot in wireless sensor network. In *2010 IEEE International Conference*

on Intelligent Computing and Intelligent Systems, volume 3, pages 173–178, Xiamen, China, Oct 2010.

[48] J. Han and Y. Chen. Multiple uav formations for cooperative source seeking and contour mapping of a radiative signal field. *Journal of Intelligent & Robotic Systems*, 74(1):323–332, Apr 2014.

[49] A. D. Haumann, K. D. Listmann, and V. Willert. Discoverage: A new paradigm for multi-robot exploration. In *2010 IEEE International Conference on Robotics and Automation*, pages 929–934, Anchorage, USA, May 2010.

[50] D. Held, J. Levinson, S. Thrun, and S. Savarese. Robust real-time tracking combining 3D shape, color, and motion. *The International Journal of Robotics Research*, 35(1-3):30–49, 2016.

[51] S. A. Hiremath, G. W. A. M. van der Heijden, F. K. van Evert, A. Stein, and C. J.F. ter Braak. Laser range finder model for autonomous navigation of a robot in a maize field using a particle filter. *Computers and Electronics in Agriculture*, 100:41–50, 2014.

[52] G. M. Hoffmann and C. J. Tomlin. Mobile sensor network control using mutual information methods and particle filters. *IEEE Transactions on Automatic Control*, 55(1):32–47, Jan 2010.

[53] J. Hoog, S. Cameron, and A. Visser. Autonomous multi-robot exploration in communication-limited environments. In *Proceedings of the 11th Conference Towards Autonomous Robotic Systems (Taros 2010)*, pages 68–75, Plymouth, UK, 2010. University of Plymouth, School of Computing and Mathematics.

[54] M. A. Hossain and I. Ferdous. Autonomous robot path planning in dynamic environment using a new optimization technique inspired by bacterial foraging technique. *Robotics and Autonomous Systems*, 64:137–141, 2015.

[55] M. Hoy, A. S. Matveev, and A. V. Savkin. Collision free cooperative navigation of multiple wheeled robots in unknown cluttered environments. *Robotics and Autonomous Systems*, 60(10):1253–1266, 2012.

[56] M. Hoy, A. S. Matveev, and A. V. Savkin. Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey. *Robotica*, 33(3):463–497, 2015.

[57] C. Hsu, H. E. Chang, and Y. Lu. Map building of unknown environment using PSO-tuned enhanced iterative closest point algorithm. In *2013 International Conference on System Science and Engineering (ICSSE)*, pages 279–284, Wroclaw, Poland, 2013. IEEE.

[58] Y. Hwang, D. Lee, and J. Lee. 3D map building for a moving based on mobile robot. In *2014 Proceedings of the SICE Annual Conference (SICE)*, pages 474–479, Sapporo, Japan, Sept 2014.

[59] Y. S. Hwang and J. M. Lee. Robust 3D map building for a mobile robot moving on the floor. In *2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 1388–1393, Busan, Korea, July 2015.

[60] K. Iagnemma, S. Shimoda, and Z. Shiller. Near-optimal navigation of high speed mobile robots on uneven terrain. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4098–4103, Nice, France, Sept 2008.

[61] G. Indiveri, S. Cretí, and A. A. Zizzari. A proof of concept for the guidance of 3D underactuated vehicles subject to constant unknown disturbances. *IFAC Proceedings Volumes*, 45(27):307–312, 2012.

[62] H. Ishida, H. Tanaka, H. Taniguchi, and T. Moriizumi. Mobile robot navigation using vision and olfaction to search for a gas/odor source. *Autonomous Robots*, 20(3):231–238, Jun 2006.

[63] S. Jain, S. Nandy, G. Chakraborty, C. S. Kumar, R. Ray, and S. N. Shome. Error modeling of laser range finder for robotic application using time domain technique. In *2011 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pages 1–5, Xi'an, China, 2011. IEEE.

[64] M. Ji, S. Azuma, and M. Egerstedt. Role-assignment in multi-agent coordination,. *International Journal of Assistive Robotics and Mechatronics*, 7(1):32–40, 2006.

[65] Y. Ji, A. Yamashita, and H. Asama. Automatic calibration and trajectory reconstruction of mobile robot in camera sensor network. In *2015 IEEE International Conference on Automation Science and Engineering (CASE)*, pages 206–211, Gothenburg, Sweden, Aug 2015.

[66] Q. Jia, M. Wang, S. Liu, J. Ge, and C. Gu. Research and development of mecanum-wheeled omnidirectional mobile robot implemented by multiple control methods. In *2016 23rd International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, pages 1–4, Nanjing, China, Nov 2016.

[67] J. Jin, Y. Kim, S. Wee, D. Lee, and N. Gans. A stable switched-system approach to collision-free wheeled mobile robot navigation. *Journal of Intelligent & Robotic Systems*, pages 1–18, 2017.

[68] D. Karaboga, S. Okdem, and C. Ozturk. Cluster based wireless sensor network routing using artificial bee colony algorithm. *Wireless Networks*, 18(7):847–860, Oct 2012.

[69] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, Aug 1996.

[70] E. K. Kim, H. Cho, E. Jang, M. K. Park, and S. Kim. Map building of indoor environment using laser range finder and geometrics. In *2015 IEEE*

International Conference on Advanced Intelligent Mechatronics (AIM), pages 1259–1264, Busan, Korea, 2015. IEEE.

[71] S. J. Kim and B. K. Kim. Dynamic ultrasonic hybrid localization system for indoor mobile robots. *IEEE Transactions on Industrial Electronics*, 60(10):4562–4573, 2013.

[72] Y. Kim and S. Kwon. A heuristic obstacle avoidance algorithm using vanishing point and obstacle angle. *Intelligent Service Robotics*, 8(3):175–183, 2015.

[73] W. Klingenberg. *A course in differential geometry*, volume 51. Springer Science & Business Media, New York, USA, 2013.

[74] C. Ko, K. Young, and Y. Hsieh. Optimized trajectory planning for mobile robot in the presence of moving obstacles. In *2015 IEEE International Conference on Mechatronics (ICM)*, pages 70–75, Nagoya, Japan, 2015. IEEE.

[75] S. Koenig and M. Likhachev. Fast replanning for navigation in unknown terrain. *IEEE Transactions on Robotics*, 21(3):354–363, June 2005.

[76] C. Kownacki. A concept of laser scanner designed to realize 3D obstacle avoidance for a fixed-wing UAV. *Robotica*, 34(2):243–257, 002 2016.

[77] S. Kubota, Y. Ando, and M. Mizukawa. Navigation of the autonomous mobile robot using laser range finder based on the non quantity map. In *2007 International Conference on Control, Automation and Systems*, pages 2329–2333, Seoul, Korea, 2007.

[78] A. S. Kundu, O. Mazumder, A. Dhar, and S. Bhaumik. Occupancy grid map generation using 360° scanning xtion pro live for indoor mobile robot navigation. In *2016 IEEE First International Conference on Control, Measurement and Instrumentation (CMI)*, pages 464–468, Kolkata, India, Jan 2016.

[79] L. Lapierre, R. Zapata, and P. Lepinay. Combined path-following and obstacle avoidance control of a wheeled robot. *The International Journal of Robotics Research*, 26(4):361–375, 2007.

[80] J. H. Lee and H. Hashimoto. Controlling mobile robots in distributed intelligent sensor network. *IEEE Transactions on Industrial Electronics*, 50(5):890–902, Oct 2003.

[81] G. Li, S. Tong, F. Cong, A. Yamashita, and H. Asama. Improved artificial potential field-based simultaneous forward search method for robot path planning in complex environment. In *2015 IEEE/SICE International Symposium on System Integration (SII)*, pages 760–765, Nagoya, Japan, 2015. IEEE.

[82] H. Li. Global shortest path planning for a wheeled mobile robot navigated by a sensor network in dynamic cluttered environments. In *2016 35th Chinese Control Conference (CCC)*, pages 5525–5530, Chengdu, China, July 2016.

[83] H. Li and A. V. Savkin. An algorithm for safe navigation of mobile robots by a sensor network in dynamic cluttered industrial environments. *Robotics and Computer-Integrated Manufacturing*. Submitted.

[84] H. Li and A. V. Savkin. Wireless sensor network based navigation of micro flying robots in the industrial internet of things. *IEEE Transactions on Industrial Informatics*. Submitted.

[85] H. Li and A. V. Savkin. A method for collision free sensor network based navigation of flying robots among moving and steady obstacles. In *2017 37th Chinese Control Conference (CCC)*, pages 6079–6083, Dalian, China, July 2017.

[86] H. Li and A. V. Savkin. A navigation algorithm for a non-holonomic mobile robot navigated by a sensor network in dynamic cluttered environments. In *2017 37th Chinese Control Conference (CCC)*, pages 5801–5804, Dalian, China, July 2017.

[87] J. Li, S. Liu, B. Zhang, and X. Zhao. RRT-A* motion planning algorithm for non-holonomic mobile robot. In *2014 Proceedings of the SICE Annual Conference (SICE)*, pages 1833–1838, Sapporo, Japan, Sept 2014.

[88] J. G. Li, Q. H. Meng, Y. Wang, and M. Zeng. Odor source localization using a mobile robot in outdoor airflow environments with a particle filter algorithm. *Autonomous Robots*, 30(3):281–292, Apr 2011.

[89] X. Li, S. Li, S. Jia, and C. Xu. Mobile robot map building based on laser ranging and kinect. In *2016 IEEE International Conference on Information and Automation (ICIA)*, pages 819–824, Ningbo, China, Aug 2016.

[90] F. Liao, S. Lai, Y. Hu, J. Cui, J. L. Wang, R. Teo, and F. Lin. 3D motion planning for UAVs in GPS-denied unknown forest environment. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 246–251, Gothenburg, Sweden, June 2016.

[91] J. Lim, S. J. Lee, G. Tewolde, and J. Kwon. Ultrasonic-sensor deployment strategies and use of smartphone sensors for mobile robot navigation in indoor environment. In *2014 IEEE International Conference on Electro/Information Technology*, pages 593–598, Milwaukee, USA, 2014.

[92] Y. Lin and S. Saripalli. Path planning using 3D Dubins curve for unmanned aerial vehicles. In *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 296–304, May 2014.

[93] T. Liu and D. M. Lyons. Leveraging area bounds information for autonomous decentralized multi-robot exploration. *Robotics and Autonomous Systems*, 74(Part A):66–78, 2015.

[94] Y. Liu and Y. Sun. Mobile robot instant indoor map building and localization using 2D laser scanning data. In *2012 International Conference on System Science and Engineering (ICSSE)*, pages 339–344, Coventry, UK, 2012. IEEE.

[95] I. R. Manchester and A. V. Savkin. Circular-navigation-guidance law for precision missile/target engagements. *Journal of Guidance, Control, and Dynamics*, 29(2):314–320, 2006.

[96] E. Masehian and N. Mohamadnejad. Path planning of nonholonomic flying robots using a new virtual obstacle method. In *2015 3rd RSI International Conference on Robotics and Mechatronics (ICROM)*, pages 612–617, Tehran, Iran, Oct 2015.

[97] S. Mastellone, D. M. Stipanović, C. R. Graunke, K. A. Intlekofer, and M. W. Spong. Formation control and collision avoidance for multi-agent nonholonomic systems: Theory and experiments. *The International Journal of Robotics Research*, 27(1):107–126, 2008.

[98] A. S. Matveev, M. Hoy, and A. V. Savkin. A globally converging algorithm for reactive robot navigation among moving and deforming obstacles. *Automatica*, 54:292–304, 2015.

[99] A. S. Matveev, M. C. Hoy, and A. V. Savkin. The problem of boundary following by a unicycle-like robot with rigidly mounted sensors. *Robotics and Autonomous Systems*, 61(3):312–327, 2013.

[100] A. S. Matveev, M. C. Hoy, and A. V. Savkin. 3D environmental extremum seeking navigation of a nonholonomic mobile robot. *Automatica*, 50(7):1802–1815, 2014.

[101] A. S. Matveev, K. S. Ovchinnikov, and A. V. Savkin. A method of reactive 3D navigation for a tight surface scan by a nonholonomic mobile robot. *Automatica*, 75:119–126, 2017.

[102] A. S. Matveev and A. V. Savkin. *Qualitative Theory of Hybrid Dynamical Systems*. Birkhäuser, Boston, 2000.

[103] A. S. Matveev and A. V. Savkin. The problem of state estimation via asynchronous communication channels with irregular transmission times. *IEEE Transactions on Automatic Control*, 48(4):670–676, April 2003.

[104] A. S. Matveev and A. V. Savkin. The problem of lqg optimal control via a limited capacity communication channel. *Systems & Control Letters*, 53(1):51–64, 2004.

[105] A. S. Matveev and A. V. Savkin. *Estimation and Control over Communication Networks*. Birkhäuser, Boston, 2009.

[106] A. S. Matveev, A. V. Savkin, M. Hoy, and C. Wang. *Safe Robot Navigation among Moving and Steady Obstacles*. Elsevier, Oxford, 2015.

[107] A. S. Matveev, H. Teimoori, and A. V. Savkin. A method for guidance and control of an autonomous vehicle in problems of border patrolling and obstacle avoidance. *Automatica*, 47(3):515–524, 2011.

[108] A. S. Matveev, H. Teimoori, and A. V. Savkin. Navigation of a unicycle-like mobile robot for environmental extremum seeking. *Automatica*, 47(1):85–91, 2011.

[109] A. S. Matveev, H. Teimoori, and A. V. Savkin. Range-only measurements based target following for wheeled mobile robots. *Automatica*, 47(1):177–184, 2011.

[110] A. S. Matveev, C. Wang, and A. V. Savkin. Real-time navigation of mobile robots in problems of border patrolling and avoiding collisions with moving and deforming obstacles. *Robotics and Autonomous systems*, 60(6):769–788, 2012.

[111] C. Mertz, L. E. Navarro-Serment, R. MacLachlan, P. Rybski, A. Steinfeld, A. Suppé, C. Urmson, N. Vandapel, M. Hebert, C. Thorpe, D. Duggins, and J. Gowdy. Moving object detection with laser scanners. *Journal of Field Robotics*, 30(1):17–43, 2013.

[112] N. Michael, M. M. Zavlanos, V. Kumar, and G. J. Pappas. Distributed multi-robot task assignment and formation control. In *2008 IEEE International Conference on Robotics and Automation*, pages 128–133, Pasadena, USA, May 2008.

[113] H. Mo and L. Xu. Research of biogeography particle swarm optimization for robot path planning. *Neurocomputing*, 148:91–99, 2015.

[114] O. Montiel, U. Orozco-Rosas, and R. Sepúlveda. Path planning for mobile robots using bacterial potential field for avoiding static and dynamic obstacles. *Expert Systems with Applications*, 42(12):5177–5191, 2015.

[115] D. Murray and J. J. Little. Using real-time stereo vision for mobile robot navigation. *Autonomous Robots*, 8(2):161–171, Apr 2000.

[116] F. Nex and F. Remondino. UAV for 3D mapping applications: a review. *Applied Geomatics*, 6(1):1–15, Mar 2014.

[117] I. Nishitani, T. Matsumura, M. Ozawa, A. Yorozu, and M. Takahashi. Human-centered X–Y–T space path planning for mobile robot in dynamic environments. *Robotics and Autonomous Systems*, 66:18–26, 2015.

[118] S. W. Noh, N. Y. Ko, and J. H. Han. Integrating elementary functions for autonomous navigation of a mobile robot. In *2014 11th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 591–593, Kuala Lumpur, Malaysia, 2014. IEEE.

[119] R. O'Flaherty and M. Egerstedt. Optimal exploration in unknown environments. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5796–5801, Hamburg, Germany, 2015. IEEE.

[120] K. K. Oh, M. C. Park, and H. S. Ahn. A survey of multi-agent formation control. *Automatica*, 53:424–440, 2015.

[121] R. Olfati-Saber. Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420, 2006.

[122] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, Jan 2007.

[123] P. Olivka, M. Mihola, P. Novák, T. Kot, and J. Babjak. The design of 3D laser range finder for robot navigation and mapping in industrial environment with point clouds preprocessing. In *Modelling and Simulation for Autonomous Systems (MESAS 2016)*, pages 371–383, Rome, Italy, 2016. Springer International Publishing.

[124] S. Oßwald, M. Bennewitz, W. Burgard, and C. Stachniss. Speeding-up robot exploration by exploiting background information. *IEEE Robotics and Automation Letters*, 1(2):716–723, 2016.

[125] J. Park, H. Kwak, Y. Kang, and D. W. Kim. Advanced fuzzy potential field method for mobile robot obstacle avoidance. *Computational Intelligence and Neuroscience*, 2016:1–13, 2016.

[126] G. Parlangeli and G. Indiveri. Single range observability for cooperative underactuated underwater vehicles. *Annual Reviews in Control*, 40(Supplement C):129–141, 2015.

[127] G. Parlangeli, P. Pedone, and G. Indiveri. Relative pose observability analysis for 3D nonholonomic vehicles based on range measurements only. *IFAC Proceedings Volumes*, 45(27):182–187, 2012.

[128] P. N. Pathirana, N. Bulusu, A. V. Savkin, and S. Jha. Node localization using mobile robots in delay-tolerant sensor networks. *IEEE Transactions on Mobile Computing*, 4(3):285–296, May 2005.

[129] Y. Pei, M. W. Mutka, and N. Xi. Coordinated multi-robot real-time exploration with connectivity and bandwidth awareness. In *2010 IEEE Interna-*

tional Conference on Robotics and Automation, pages 5460–5465, Anchorage, USA, May 2010.

[130] I. R. Petersen and A. V. Savkin. *Robust Kalman filtering for signals and systems with large uncertainties.* Birkhäuser, Boston, 1999.

[131] S. T. Pfister, S. I. Roumeliotis, and J. W. Burdick. Weighted line fitting algorithms for mobile robot map building and efficient data representation. In *2003 IEEE International Conference on Robotics and Automation*, volume 1, pages 1304–1311, Taipei, Taiwan, Sept 2003.

[132] S. R. B. Prabhu, C. V. Dhasharathi, R. Prabhakaran, M. R. Kumar, S. W. Feroze, and S. Sophia. Environmental monitoring and greenhouse control by distributed sensor network. *International Journal of Advanced Networking and Applications*, 5(5):2060–2065, Mar 2014.

[133] G. Pudics, M. Zsolt Szabó-Resch, and Z. Vámossy. Safe robot navigation using an omnidirectional camera. In *2015 16th IEEE International Symposium on Computational Intelligence and Informatics (CINTI)*, pages 227–231, Hangzhou, China, 2015. IEEE.

[134] S. Pütz, T. Wiemann, J. Sprickerhof, and J. Hertzberg. 3d navigation mesh generation for path planning in uneven terrain. *IFAC-PapersOnLine*, 49(15):212–217, 2016. 9th IFAC Symposium on Intelligent Autonomous Vehicles IAV 2016.

[135] R. Raja, A. Dutta, and K.S. Venkatesh. New potential field method for rough terrain path planning using genetic algorithm for a 6-wheel rover. *Robotics and Autonomous Systems*, 72:295–306, 2015.

[136] A. M. Rao, K. Ramji, B. S. K. Sundara Siva Rao, V. Vasu, and C. Puneeth. Navigation of non-holonomic mobile robot using neuro-fuzzy logic with integrated safe boundary algorithm. *International Journal of Automation and Computing*, 14(3):285–294, Jun 2017.

[137] M. N. Rastgoo, B. Nakisa, and M. Z. A. Nazri. A hybrid of modified PSO and local search on a multi-robot search system. *International Journal of Advanced Robotic Systems*, 12(86):1–15, 2015.

[138] P. Rawat, K. D. Singh, H. Chaouchi, and J. M. Bonnin. Wireless sensor networks: a survey on recent developments and potential synergies. *The Journal of Supercomputing*, 68(1):1–48, 2014.

[139] R. Ray, V. Kumar, D. Banerji, and S. N. Shome. Simultaneous localisation and image intensity based occupancy grid map building–a new approach. In *2012 Third International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, pages 143–148, Kingston, Canada, 2012. IEEE.

[140] K. Rebai, A. Benabderrahmane, O. Azouaoui, and N. Ouadah. Moving obstacles detection and tracking with laser range finder. In *2009 International Conference on Advanced Robotics (ICAR)*, pages 1–6, Munich, Germany, 2009. IEEE.

[141] I. Rekleitis, G. Dudek, and E. Milios. Multi-robot collaboration for robust exploration. *Annals of Mathematics and Artificial Intelligence*, 31(1):7–40, Oct 2001.

[142] H. Rezaee and F. Abdollahi. Mobile robots cooperative control and obstacle avoidance using potential field. In *2011 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 61–66, Budapest, Hungary, July 2011.

[143] H. Rezaee and F. Abdollahi. A decentralized cooperative control scheme with obstacle avoidance for a team of mobile robots. *IEEE Transactions on Industrial Electronics*, 61(1):347–354, Jan 2014.

[144] J. J. Robles, M. Deicke, and R. Lehnert. 3D fingerprint-based localization for wireless sensor networks. In *2010 7th Workshop on Positioning, Navigation and Communication*, pages 77–85, Dresden, Germany, March 2010.

[145] C. Rösmann, F. Hoffmann, and T. Bertram. Integrated online trajectory planning and optimization in distinctive topologies. *Robotics and Autonomous Systems*, 88:142–153, 2017.

[146] E. Royer, M. Lhuillier, M. Dhome, and J. M. Lavest. Monocular vision for mobile robot localization and autonomous navigation. *International Journal of Computer Vision*, 74(3):237–260, Sep 2007.

[147] J. C. L. Barreto S., A. G. S. Conceição, C. E. T. Dórea, L. Martinez, and E. R. de Pieri. Design and implementation of model-predictive control with friction compensation on an omnidirectional mobile robot. *IEEE/ASME Transactions on Mechatronics*, 19(2):467–476, April 2014.

[148] C. T. Sauer, H. Brugger, E. P. Hofer, and B. Tibken. Odometry error correction by sensor fusion for autonomous mobile robot navigation. In *Proceedings of the 18th IEEE Instrumentation and Measurement Technology Conference (IMTC 2001)*, pages 1654–1658, Budapest, Hungary, 2001. IEEE.

[149] A. V. Savkin. Coordinated collective motion of groups of autonomous mobile robots: analysis of vicsek's model. *IEEE Transactions on Automatic Control*, 49(6):981–982, 2004.

[150] A. V. Savkin. Analysis and synthesis of networked control systems: Topological entropy, observability, robustness and optimal control. *Automatica*, 42(1):51–62, 2006.

[151] A. V. Savkin. The problem of coordination and consensus achievement in groups of autonomous mobile robots with limited communication. *Nonlinear Analysis: Theory, Methods & Applications*, 65(5):1094–1102, 2006.

[152] A. V. Savkin and R. J. Evans. *Hybrid Dynamical Systems. Controller and Sensor Switching Problems*. Birkhäuser, Boston, 2002.

[153] A. V. Savkin and M. Hoy. Reactive and the shortest path navigation of a wheeled mobile robot in cluttered environments. *Robotica*, 31(02):323–330, 2013.

[154] A. V. Savkin and H. Huang. The problem of minimum risk path planning for flying robots in dangerous environments. In *2016 35th Chinese Control Conference (CCC)*, pages 5404–5408, Chengdu, China, July 2016.

[155] A. V. Savkin and H. Li. A collision-free area search and 3D map building algorithm for a ground mobile robot in unknown indoor environments. In *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. Accepted.

[156] A. V. Savkin and H. Li. Simultaneous collision free area exploration and 3D map building of unknown indoor environments by a ground mobile robot. *Robotics and Autonomous Systems*. Submitted.

[157] A. V. Savkin and H. Li. Collision free navigation of a non-holonomic ground robot for search and building maps of unknown areas with obstacles. In *2016 35th Chinese Control Conference (CCC)*, pages 5409–5414, Chengdu, China, July 2016.

[158] A. V. Savkin and H. Li. A safe area search and map building algorithm for a wheeled mobile robot in complex unknown cluttered environments. *Robotica*, 2017. Advance online publication.

[159] A. V. Savkin, E. Skafidas, and R. J. Evans. Robust output feedback stabilizability via controller switching. *Automatica*, 35(1):69–74, 1999.

[160] A. V. Savkin and H. Teimoori. Decentralized formation flocking and stabilization for networks of unicycles. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 984–989, Shanghai, China, Dec 2009.

[161] A. V. Savkin and H. Teimoori. Bearings-only guidance of a unicycle-like vehicle following a moving target with a smaller minimum turning radius. *IEEE Transactions on Automatic Control*, 55(10):2390–2395, 2010.

[162] A. V. Savkin and C. Wang. A simple biologically inspired algorithm for collision-free navigation of a unicycle-like robot in dynamic environments with moving obstacles. *Robotica*, 31(06):993–1001, 2013.

[163] A. V. Savkin and C. Wang. Seeking a path through the crowd: Robot navigation in unknown dynamic environments with moving obstacles based on an integrated environment representation. *Robotics and Autonomous Systems*, 62(10):1568–1580, 2014.

[164] A. V. Savkin, C. Wang, A. Baranzadeh, Z. Xi, and H. T. Nguyen. A method for decentralized formation building for unicycle-like mobile robots. In *2013 9th Asian Control Conference (ASCC)*, pages 1–5, Istanbul, Turkey, June 2013.

[165] A. V. Savkin, C. Wang, A. Baranzadeh, Z. Xi, and H. T. Nguyen. Distributed formation building algorithms for groups of wheeled mobile robots. *Robotics and Autonomous Systems*, 75, Part B:463–474, 2016.

[166] P. Schmuck, S. A. Scherer, and A. Zell. Hybrid metric-topological 3D occupancy grid maps for large-scale mapping. *IFAC-PapersOnLine*, 49(15):230–235, 2016.

[167] J. Sfeir, M. Saad, and H. Saliah-Hassane. An improved artificial potential field approach to real-time mobile robot path planning in an unknown environment. In *2011 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, pages 208–213, Montreal, Canada, Sept 2011.

[168] A. Sgorbissa and R. Zaccaria. Planning and obstacle avoidance in mobile robotics. *Robotics and Autonomous Systems*, 60(4):628–638, 2012.

[169] W. Sheng, Q. Yang, J. Tan, and N. Xi. Distributed multi-robot coordination in area exploration. *Robotics and Autonomous Systems*, 54(12):945–955, 2006.

[170] X. Shi, G. Junyao, Y. Liu, J. Zhao, F. Zhao, H. Cao, and C. Liu. Low-cost map building and obstacle avoidance of miniature reconnaissance robot. In *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 208–213, Qingdao, China, Dec 2016.

[171] C. Y. Shih and P. J. Marrón. COLA: Complexity-reduced trilateration approach for 3D localization in wireless sensor networks. In *2010 Fourth International Conference on Sensor Technologies and Applications*, pages 24–32, Mestre, Italy, July 2010.

[172] S. Shimoda, Y. Kuroda, and K. Iagnemma. Potential field navigation of high speed unmanned ground vehicles on uneven terrain. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 2828–2833, Barcelona, Spain, April 2005.

[173] C. Siagian, C. K. Chang, and L. Itti. Mobile robot navigation system in outdoor pedestrian environment using vision-based road recognition. In *2013 IEEE International Conference on Robotics and Automation*, pages 564–571, Karlsruhe, Germany, 2013.

[174] A. K. Singh and K. M. Krishna. Feasible acceleration count: A novel dynamic stability metric and its use in incremental motion planning on uneven terrain. *Robotics and Autonomous Systems*, 79:156–171, 2016.

[175] E. Skafidas, R. J. Evans, A. V. Savkin, and I. R. Petersen. Stability results for switched controller systems. *Automatica*, 35(4):553–564, 1999.

[176] X. Sun, X. Zhao, E. Li, H. Li, and Z. Liang. Mobile robot navigation using RSSI potential field in wireless sensor network. *Journal of Computational Information Systems*, 6(14):4751–4759, 2010.

[177] H. Surmann, A. Nüchter, and J. Hertzberg. An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. *Robotics and Autonomous Systems*, 45(3):181–198, 2003.

[178] N. K. Suryadevara and S. C. Mukhopadhyay. Wireless sensor network based home monitoring system for wellness determination of elderly. *IEEE Sensors Journal*, 12(6):1965–1972, June 2012.

[179] L. Tang, S. Dian, G. Gu, K. Zhou, S. Wang, and X. Feng. A novel potential field method for obstacle avoidance and path planning of mobile robot. In *2010 3rd International Conference on Computer Science and Information Technology*, volume 9, pages 633–637, Chengdu, China, 2010.

[180] S. Tatsukawa, T. Nakanishi, R. Nagao, T. Wada, M. Fujimoto, and K. Mutsuura. New moving control of mobile robot without collision with wall and obstacles by passive RFID system. In *2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–7, Calcary, Canada, 2015. IEEE.

[181] H. Teimoori and A. V. Savkin. A biologically inspired method for robot navigation in a cluttered environment. *Robotica*, 28(05):637–648, 2010.

[182] H. Teimoori and A. V. Savkin. Equiangular navigation and guidance of a wheeled mobile robot based on range-only measurements. *Robotics and Autonomous Systems*, 58(2):203–215, 2010.

[183] F. Tian, K. M. Chao, Z. Feng, K. Xing, and N. Shah. Using a wireless visual sensor network to harmonically navigate multiple low-cost wheelchairs in an indoor environment. *Journal of Network and Computer Applications*, 62:88–99, 2016.

[184] J. Ureña, D. Gualda, Á. Hernández, E. García, J. M. Villadangos, M. C. Pérez, J. C. García, J. J. García, and A. Jiménez. Ultrasonic local positioning system for mobile robot navigation: From low to high level processing. In *2015 IEEE International Conference on Industrial Technology (ICIT)*, pages 3440–3445, Seville, Spain, 2015.

[185] J. Vallvé and J. Andrade-Cetto. Potential information fields for mobile robot exploration. *Robotics and Autonomous Systems*, 69:68–79, 2015.

[186] J. Vilca, L. Adouane, and Y. Mezouar. A novel safe and flexible control strategy based on target reaching for the navigation of urban vehicles. *Robotics and Autonomous Systems*, 70:215–226, 2015.

[187] C. Wang, A. V. Savkin, R. Clout, and H. T. Nguyen. An intelligent robotic hospital bed for safe transportation of critical neurosurgery patients along crowded hospital corridors. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 23(5):744–754, Sept 2015.

[188] C. Wang, A. V. Savkin, and M. Garratt. Collision free navigation of flying robots among moving obstacles. In *2016 35th Chinese Control Conference (CCC)*, pages 4545–4549, Chengdu, China, 2016.

[189] C. Wang, A. V. Savkin, and M. Garratt. A strategy for safe 3D navigation of non-holonomic robots among moving obstacles. *Robotica*, 2017.

[190] D. Wang, Y. Duan, and J. Wang. Environment exploration and map building of mobile robot in unknown environment. *International Journal of Simulation and Process Modelling*, 10(3):241–252, 2015.

[191] M. Wang, W. Wang, J. Xiong, and L. Yan. A consistent map building method based on surf loop closure detection. In *2013 IEEE 3rd Annual International Conference on Cyber Technology in Automation, Control and Intelligent Systems (CYBER)*, pages 92–95, Nanjing, China, 2013. IEEE.

[192] Q. Wang, Z. Chen, P. Liu, and Q. Hua. Distributed multi-robot formation control in switching networks. *Neurocomputing*, 2017. Advance online publication.

[193] W. T. Wang and K. F. Ssu. Obstacle detection and estimation in wireless sensor networks. *Computer Networks*, 57(4):858–868, 2013.

[194] Y. Wang, T. Mai, and J. Mao. Adaptive motion/force control strategy for non-holonomic mobile manipulator robot using recurrent fuzzy wavelet neural networks. *Engineering Applications of Artificial Intelligence*, 34:137–153, 2014.

[195] T. Wattanavekin, T. Ogata, T. Hara, and J. Ota. Mobile robot exploration by using environmental boundary information. *ISRN Robotics*, 2013:1–11, 2013.

[196] G. Wen, Z. Duan, G. Chen, and W. Yu. Consensus tracking of multi-agent systems with lipschitz-type node dynamics and switching topologies. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 61(2):499–511, 2014.

[197] Z. Wu, Z. Guan, X. Wu, and T. Li. Consensus based formation control and trajectory tracing of multi-agent robot systems. *Journal of Intelligent and Robotic Systems*, 48(3):397–410, Mar 2007.

[198] F. Xiao, L. Wang, J. Chen, and Y. Gao. Finite-time formation control for multi-agent systems. *Automatica*, 45(11):2605–2611, 2009.

[199] S. Xie and Y. Wang. Construction of tree network with limited delivery latency in homogeneous wireless sensor networks. *Wireless Personal Communications*, 78(1):231–246, Sep 2014.

[200] H. Yang, X. Fan, P. Shi, and C. Hua. Nonlinear control for tracking and obstacle avoidance of a wheeled mobile robot with nonholonomic constraint. *IEEE Transactions on Control Systems Technology*, 24(2):741–746, March 2016.

[201] J. Yang, Y. Shi, and H. Rong. Random neural Q-learning for obstacle avoidance of a mobile robot in unknown environments. *Advances in Mechanical Engineering*, 8(7):1–15, 2016.

[202] Y. Yao, Q. Ni, Q. Lv, and K. Huang. A novel heterogeneous feature ant colony optimization and its application on robot path planning. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 522–528, Sendai, Japan, 2015. IEEE.

[203] Z. Yao and K. Gupta. Distributed roadmaps for robot navigation in sensor networks. *IEEE Transactions on Robotics*, 27(5):997–1004, Oct 2011.

[204] M. Yokoyama and T. Poggio. A contour-based moving object detection and tracking. In *2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 271–276, Beijing, China, 2005. IEEE.

[205] A. Yorozu and M. Takahashi. Obstacle avoidance with translational and efficient rotational motion control considering movable gaps and footprint for autonomous mobile robot. *International Journal of Control, Automation and Systems*, 14(5):1352–1364, Oct 2016.

[206] W. Yuan, Z. Cao, M. Tan, and J. Liu. Topological mapping and navigation based on visual sensor network. In *2014 IEEE International Conference on Mechatronics and Automation*, pages 1686–1690, Tianjin, China, 2014.

[207] A. Zakhar'eva, A. S. Matveev, M. C. Hoy, and A. V. Savkin. Distributed control of multiple non-holonomic robots with sector vision and range-only measurements for target capturing with collision avoidance. *Robotica*, 33(2):385–412, 2015.

[208] Y. Zhang, S. Wang, G. Fan, and J. Zhou. Navigation for indoor mobile robot based on wireless sensor network. In *Wireless Algorithms, Systems, and Applications: 8th International Conference (WASA 2013)*, pages 325–336, Zhangjiajie, China, Aug 2013. Springer Berlin Heidelberg.

[209] Z. Zhang, Z. Li, D. Zhang, and J. Chen. Path planning and navigation for mobile robots in a hybrid sensor network without prior location information. *International Journal of Advanced Robotic Systems*, 10(3):1–12, 2013.

[210] N. Zhou, X. Zhao, and M. Tan. RSSI-based mobile robot navigation in grid-pattern wireless sensor network. In *2013 Chinese Automation Congress*, pages 497–501, Changsha, China, Nov 2013.

[211] Z. Ziaei, R. Oftadeh, and J. Mattila. Global path planning with obstacle avoidance for omnidirectional mobile robot using overhead camera. In *2014 IEEE International Conference on Mechatronics and Automation*, pages 697–704, Tianjin, China, Aug 2014.