# Autonomous Navigation Strategies for UGVs/UAVs

Autonomenavigatiestrategieën voor UGV's/UAV's

Thoa Mac Thi

Promotoren: prof. dr. ir. C.-M. Ionescu, dr. ir. C. Copot
Proefschrift ingediend tot het behalen van de graad van
Doctor in de ingenieurswetenschappen: werktuigkunde-elektrotechniek

Vakgroep Elektrische Energie, Metalen, Mechanische Constructies en Systemen
Voorzitter: prof. dr. ir. L. Dupré
Faculteit Ingenieurswetenschappen en Architectuur
Academiejaar 2017 - 2018

UNIVERSITEIT
GENT

ISBN 978-94-6355-113-7
NUR 950
Wettelijk depot: D/2018/10.500/31

## Members of the examination committee

**Chairman**
Prof. dr. ir. Daniel De Zutter, Department of Information Technology, UGent

**Secretary**
Prof. dr. ir. Guillaume Crevecoeur, EELAB-EA08, UGent

**Examination Committee**
Prof. dr. ir. Radu-Emil Precup, Politechnica University of Timisoara, Romania
Prof. dr. ir. Steve Vanlanduit, University of Antwerp, Belgium
Prof. dr. ir. Hendrik De Bie, Ghent University, Belgium

**Other members**
Prof. dr. ir. Clara Ionescu, Ghent University, *promotor*
Dr. ir. Cosmin Copot, University of Antwerp, *copromotor*
Prof. dr. ir. Robain De Keyser, Ghent University, (promoter until October 2016)

# Contents

# Permission for Usage

*Mac Thi Thoa, February 2018*

# Acknowledgments

This thesis is a journey in the labyrinth of knowledge where interaction with a large number of people, with whom I have had the pleasure to discuss a wide range of topics in robotics, optimization and control.

I would like to express my special appreciations and thanks to my promoters, Prof. Clara Ionescu, Prof. Robin De Keyser, Dr. ir. Cosmin Copot for their continuous guidance and very kind support, expertise, encouragement and freedom, which they offered during my research. It was my pleasure working in their research group. I gratefully acknowledge Prof. Tran Duc Trung for his advises for my research.

I am also very grateful to all other committee members for helping me to improve this thesis.

I would like to acknowledge the Eramus Mundus and Prof. Robin De Keyser for their financial support of my research; without their supports this work would not have been completed.

I would also like to thank my colleagues at school of mechanical engineering for their special supports. I am particularly grateful to Prof. Nguyen Dac Trung for his administrative help.

It has been fortunate for me to have very nice colleagues at Laboratory of Dynamical Systems and Control Research in Ghent. My gratitude to Kevin, Amélie, Dana, Andrada, Jasper, Ricardo, Shiquan, Dongfei, Andres, Maria, Mihaela for being very kind lab-mates. Thanks Bram and Arne for their help in technical works.

Living abroad far away would be much harder without the company of many Vietnamese friends. Many thanks for their helps and encouragements during my stay in Belgium, my special thanks to Tu Ha and Thoa Hai for their kindnesses.

Finally, this dissertation would not have come to a successful end without the emotional supports from my family. Very special thanks are given to my brother, Huyen and his wife, Lan for very nice moments we have spent together. Words can not express how grateful I am to my sister, my parents for their motivation and encourages to move forward. My special thanks to my sons, Kien and Nam, for the endless happiness and motivation they have brought to my life. Finally, my special thanks to my husband and my mother. I can't thank you enough for supporting me for everything, the understanding, the energy and the love they are giving me all the time.

*Mac Thi Thoa*
*Ghent, February 2018*

# Abstract

Unmanned Ground Vehicles (UGVs) and Unmanned Aerial Vehicles (UAVs) are ideal platforms for surveillance, search and rescue in confined environments due to their flexibility and superior mobility. In such missions, it is essential that the agents are capable of autonomous navigation in various environments. Despite recent successes in commercialization of GPS-based autonomous vehicles, autonomous navigation in complex and possibly GPS-denied environments gives rise to engineering challenges. These problems require an integrated approach for perception, estimation, planning, control and high level situational awareness. The objectives of this thesis is to address these issues by developing a low cost, consistently accurate navigation system using on-board sensors. As path planning is a crucial component of autonomous navigation, we emphasize that aspect.

A large amount of research has been performed and it is still ongoing in the area of autonomous navigation of mobile platforms but very few have reported convincing experimental results. There is a gap between theoretical and experimental results, although a great amount of theoretical work has been validated in simulations and off-line processing. The most current research activities provide effective solutions to the robot path planning in known static environments. The contemporary autonomous navigation approaches have not yet reached the level of robustness and reliability required for the real-world applications. Traveling in unknown dynamic environments is still a problematic aspect for research.

Moreover, no optimal solution exists for all the autonomous navigation applications, as each algorithm has its own advantages and disadvantages. For each scenario, a suitable solution is chosen based on the application's purposes, the system characteristics, the working spaces and the user's experiences. If

an application requires the knowledge of a human operator, a fuzzy control is the most suitable strategy. However, in cluttered environments, where the robot faces many different situations, integrated methods which combine several algorithms may achieve the better performances.

To perform online autonomous navigation for a complex system where highly dynamic characteristics and limited computation capacities are involved, an effective algorithm with a fast updating ability is required. The potential field method is particularly attractive as it has a simple structure, a low computational complexity and a straightforward implementation.

From these points of view, this thesis develops further path planning schemes in a diversity of environments, such as static, dynamic, known and unknown working spaces. The main theoretical contributions are:

- An intelligent fuzzy algorithm which allows the mobile platform to easily adapt with different terrains (e.g. a high speed where its road is smooth, a low speed where its road is rocky) and obstacles (e.g. speed up in case no obstacle and slow down otherwise) in outdoor environments. This algorithm has a modular structure that can be extended to incorporate new behaviors.

- A novel hierarchical global path planning method for mobile robots in cluttered environments based on the proposed constrained multi objective particle swarm optimization algorithm with an accelerated update methodology (MOPSO) and Pareto dominance principle. It obtains the optimal path in terms of the path length and the path smoothness considering the robot's constraints. The advantages include the computational efficiency.

- The development of an improved potential field method which performs a real-time obstacle avoidance in unknown/dynamic environments and solves the non-reachable goal problem.

Besides, this thesis offers a detailed investigation of other autonomous navigation problems, such as the localization, obstacle detection, velocity estimation of moving objects and optimal control algorithms. Other practical contributions of this thesis are:

- A position-estimation approach using a sensor fusion technique in a structured environment. This localization method presents how to get the UAV's states (the position and the orientation), dealing with data provided by an optical sensor and an inertial measurement unit (IMU). Such a data fusion scheme also considers the time delay of the camera signals because of the communication protocols.

- The development and implementation of an optimal proportional-integral-derivative (PID) controller based on the novel multi-objective particle swarm optimization with an accelerated update methodology. This MOPSO aims to facilitate convergence to the PID's optimal parameters.

- The design and implementation of a sensor fusion method which fuses Pozyx's signal and IMU's signal.

- A method to detect static/dynamic obstacles using information obtained from the UAV's camera.

- The velocity estimation methodology of a moving obstacle based on the UAV's vision system. This velocity information enables the UAV to perform the landing task in complex situations.

- The development of a low-cost, high-scalability and easy-to-use navigation system to assist the UAV's landing.

The content of this work is summarized hereafter.

*Chapter 1* addresses the problem description, motivation, organization and main contributions of this thesis.

*Chapter 2* presents the literature review of the mobile robot path planning as it is the most challenging task of autonomous navigation. The path planning techniques are classified into two main categories: classical methods and heuristic methods. The classical methods consist of cell decomposition, potential field method, sub-goal network and sampling-based methods. The approaches are simple; however, they commonly are computationally expensive and may possibly fail when the robot confronts with an uncertainty. The heuristic-based method comprises of neural network, fuzzy logic, nature-inspired and hybrid algorithms. The strengths and drawbacks of each algorithm are discussed.

*Chapter 3* considers an UGV's autonomous navigation in a static environment with an expectation to inherit the expertise of a human operator. Fuzzy logic control is used as a mechanism towards accomplishing this goal. A test case study is an autonomous geophysical measurement platform (AGMP) since this mobile agent is expected to mimic the archaeologist's skills. To account for a situation of the AGMP working in outdoor environments consisting of different terrains, such as smooth, rough or rocky, the AGMP should also be able to adjust its speed with changes in the terrains. The fuzzy logic rules that govern the AGMP's motion are simpler and more understandable. In addition, the behavior-based strategy has a modular structure that can be extended to incorporate new behaviors.

*Chapter 4* presents the development of a multi-objective optimal path considering the abilities of mobile robots in a cluttered environment. Unfortunately, planners such as, the fuzzy-based approach presented in chapter 3 or neural networks, produce solutions that are not optimal. The nature-inspired algorithms can deal with multi-objective robot path planning optimization problems. However, they are time consuming and may fail to create a feasible path in some scenarios. Consequently, a novel hierarchical approach is proposed which combines: 1) a triangular decomposition; 2) a Dijkstra's algorithm; and 3) a proposed constrained multi-objective particle swarm optimization (CMOPSO) with an accelerated update methodology based on Pareto dominance principle. The aim of the approach is to generate the optimal collision-free paths focused on minimizing the path length and maximizing the smoothness regarding the mobile robots' constraints. The main theoretical contributions are verified through simulations.

*Chapter 5* investigates an autonomous methodology for a low-cost commercial Ar.Drone 2.0 in partly unknown indoor flight. Firstly, a localization based on a sensor fusion technique is developed, which permits a robust estimation of the UAV's position and orientation in an unpredictable environment. Secondly, an improved potential field method is presented to successfully guide the UAV reach the target and avoid all the unknown/dynamic obstacles. The method has several advantages, such as a simple structure, a low computational complexity and an ease of implementation. Thirdly, after obtaining the UAV's dynamic models by the identification process, the optimal controllers are designed to accurately track the UAV's paths. The experimental results demonstrate the feasibility of the proposed strategy, which opens new possibilities for the autonomous mobile agents in complex environments.

*Chapter 6* provides a low-cost, highly scalable and easy-to-use landing approach of the UAV for rescue missions. A sensor fusion method which fuses IMU's signal and Poxyz's signal for localization process is developed in GPS-denied environments. An obstacle avoidance algorithm, which effectively detects unknown static/dynamic obstacles, is also investigated. Besides, a velocity estimation approach for a moving obstacle based on the UAV's vision system is proposed to assist the UAV landing on the optimal position which has the shortest distance to the target.

*Chapter 7* provides an overview of this thesis's contributions and several suggestions of the potential autonomous navigation directions for the future research.

# Nederlandse samenvatting

Onbemande landvoertuigen (EN: unmanned ground vehicles (UGV) ) en onbemande luchtvaartuigen (EN: unmanned aerial vehicles (UAV)) zijn door hun flexibiliteit en mobiliteit de ideale platformen voor bewakings-, opsporings- en reddingsoperaties in besloten omgevingen. In deze toepassingen is autonome navigatie van het voertuig van uitermate belang. Ondanks de succesvolle commercialisering van op GPS gebaseerde autonome voertuigen, blijft het navigeren in complexe dynamische omgevingen - soms zonder bruikbare GPS ontvangst – zorgen voor zware technologische uitdagingen. Deze uitdagingen vergen een geïntegreerde aanpak van perceptie, schatting, planning en regeling van het voertuig samen met een gedegen situatiekennis. In dit proefschrift wordt dit vraagstuk te lijf gegaan door het ontwikkelen van een goedkoop accuraat navigatiesysteem via gebruik van ingebouwde (EN: on board) sensoren. Verder wordt er ook nadruk gelegd op automatische traject planning.

Ondanks het groot aantal onderzoeksprojecten op het vlak van autonome navigatie, blijven overtuigende experimentele resultaten uit. Hoewel een groot aandeel van het theoretisch onderzoek reeds gevalideerd is in simulaties en offline berekeningen, blijft er een kloof tussen theorie en praktijk in online en dynamische omgevingen. Het huidig onderzoek is veelal gericht op optimale route planning in gekende, statische omgevingen. Deze aanpak heeft nog niet de gewenste robuustheid en betrouwbaarheid bereikt, nodig voor praktische real-world toepassingen. Navigeren in onbekende, dynamische omgevingen blijft een uitdaging voor vele onderzoekers.

Bovendien bestaat er geen optimale oplossing voor elke autonome navigatie toepassing, want elk gebruikt algorithme heeft zijn voordelen en nadelen. Elke toepassing vergt een individuele aanpak gebaseerd op de doelstelling, de systeemeigenschappen, de werkruimte, de beschikbare informatie en

de kennis van obstakels. Zo is het in een scenario dat de kennis van een menselijke gebruiker benut, dikwijls aangewezen om een vage (EN: fuzzy) regeling te hanteren. Anderzijds, in een weinig-gestructureerde rommelige omgeving (EN: cluttered environment) kan het autonoom voertuig voor zeer veel verschillende situaties komen te staan. Hier zal eerder een gecombineerde aanpak van verschillende algorithmes voor de beste resultaten zorgen.

Voor online autonome navigatie in complexe systemen en met gelimiteerde rekenkracht is een snel en efficiënt algorithme nodig, om zo optimaal de dynamische omgeving te verkennen. Hiervoor wordt de potentiaalveld methode (EN: potential field method) aangewend. Deze methode heeft een simpele structuur, vergt weinig berekeningen en is makkelijk te implementeren.

In het kader van de bovengenoemde punten, zal dit proefschrift de route planning verder ontwikkelen in een varia van omgevingen: statisch, dynamisch, gekende en ongekende omgevingen. De voornaamste theoretische bijdragen zijn:

- Een intelligent algorithme - gebaseerd op vage logica - om te realiseren dat mobiele platformen zich vlug aanpassen aan verschillende terreinen (bv. hogere snelheid bij vlakke weg en lagere snelheid naast rotsen) en aan het al-dan-niet aanwezig zijn van obstakels in de nabijheid. Dit algorithme heeft een modulaire structuur die kan uitgebreid worden indien ander gedrag gewenst is.

- Een vernieuwende hiërarchische globale routeplanner voor mobiele robots in een rommel omgeving, gebaseerd op het constrained Multi-Objective Particle Swarm Optimalisatie (MOPSO) algorithme, gebruik makend van een versnelde update methodologie en het Pareto-dominantie principe.

- Een verbeterde potentiaalveld methode met real-time ontwijking van obstakels in een onbekende en dynamische omgeving; deze methode pakt ook het probleem van een onbereikbaar doel (EN: non-reachable goal) aan.

Daarnaast biedt dit proefschrift een gedetailleerd onderzoek naar andere problemen die gepaard gaan met autonome navigatie zoals lokalisatie, detecteren van obstakels, het schatten van snelheid van obstakels en optimale controle. Hieruit volgen deze bijdragen:

- Positieschatting via sensorfusie in een gestructureerde omgeving. Als toepassing worden positie en oriëntatie van een UAV gehaald uit data van zijn optische sensor en de inertiale meeteenheid (EN: IMU). Er wordt ook rekening gehouden met de vertragingen in het camerasignaal tengevolge van communicatieprotocollen.

- De ontwikkeling en het implementeren van een optimale PID regelaar (EN: Proportional-Integral-Derivative controller) gebaseerd op het constrained Multi-Objective Particle Swarm Optimalisatie (MOPSO) algorithme met een versnelde update methodologie. The MOPSO verbetert de convergentie tijdens het optimaliseren.

- Het ontwerp en de implementatie van een sensorfusie methode die gebruik maakt van *Pozyx* en ingebouwde sensoren.

- Een methode voor het detecteren van statische en dynamische obstakels uit de informatie afkomstig van een reguliere camera.

- Het schatten van de snelheid van bewegende obstakels met een visiesysteem. Dit wordt toegepast bij het landen van een UAV in complexe situaties.

- De ontwikkeling van een goedkoop, schaalbaar en gebruiksvriendelijk navigatiesysteem ter assistentie bij het landen van een UAV.

# Used Abbreviations

| | |
|---|---|
| CD | Cell Decomposition Method |
| PFM | Potential Field Method |
| MPFM | Modified Potential Field Method |
| SG | Subgoal Method |
| SBP | Sampling-Based Motion Planning |
| PRM | Probabilistic Road-Map |
| VG | Visibility Graph |
| VD | Voronoi Diagram |
| RRT | Rapidly-exploring Random Trees |
| HeAT-RT | Heuristic Arrival Time Field biased Random Tree |
| NN | Neural Network |
| FL | Fuzzy Logic |
| FC | Fuzzy Control |
| PCA | Principal Component Analysis |
| PAFARTNA | Prune-Able Fuzzy Adaptive Resonance Theory Neural Architecture |
| GA | Genetic Algorithms |
| PSO | Particle Swarm Optimization |
| NPSO | Negative Particle Swarm Optimization |
| ACO | Ant Colony Optimization |
| ACO-MH | Ant Colony Optimization Meta-Heuristic |
| ATV | Autonomous Terrain Vehicle |
| AGMP | Autonomous Geophysical Measurement Platform |
| MNHS | Multi-Neuron Heuristic Search |
| CMOPSO | Constrained Multi-objective Particle Swarm Optimization |
| MOPSO | Multi-objective Particle Swarm Optimization |
| SISO | Single Input Single Output |
| MIMO | Multiple Inputs Multiple Outputs |
| MISO | Multiple Inputs Single Output |

| | |
|---|---|
| LTI | Linear Time Invariant |
| PRBS | Pseudo-Random Binary Signal |
| IMU | Inertial Measurement Unit |
| RGB | Red Green Blue |
| ROS | Robot Operating System |
| UGV | Unmanned Ground Vehicle |
| UAV | Unmanned Aerial Vehicle |
| WiFi | Wireless Fidelity |
| FOV | Field Of View |
| PID | Proportional-Integral-Derivative |
| SSE | Steady State Error |
| SHV | Hue Saturation Value |
| GPS | Global Positioning System |
| LIDAR | Light Detection and Ranging |
| VTOL | Vertical Take-Off and Landing |
| DOF | Degrees-Of-Freedom |

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Problem Description

In various industries and real-life situations, robots are widely used to replace humans for dangerous, dirty and repetitive works. Among these robots, UAVs (Unmanned Aerial Vehicles) and UGVs (Unmanned Ground Vehicles) are the most important families, because of their capability to perform tasks that are difficult or costly to accomplish. UGVs and UAVs are the ultimate agents for surveillance, search and rescue in confined environments due to their flexibility and superior mobility [13]. A robot autonomous navigation in complex and GPS-denied environments is a really challenging task for engineering, therefore this problem attracts a lot of attention from researchers.

For humans, the navigation ability is eminent. For mobile platforms, however, the navigation in real-world environments is an extremely complicated task. Such environments are characterized by their complex structure and movement of the objects. A problem of a navigating robot in crowded environments is that it can be blocked by obstacles. In such cases, the robot becomes immobilized and it is not able to reach the goal position. Another difficulty that may appear is the collision with moving obstacles, as the mobile robot cannot sense or predict the motions of the obstacles [14].

A fully autonomous robot can gain information about environments, work for an extended period without the human's intervention, move either all or part of itself without the human's assistance, avoid situations that are

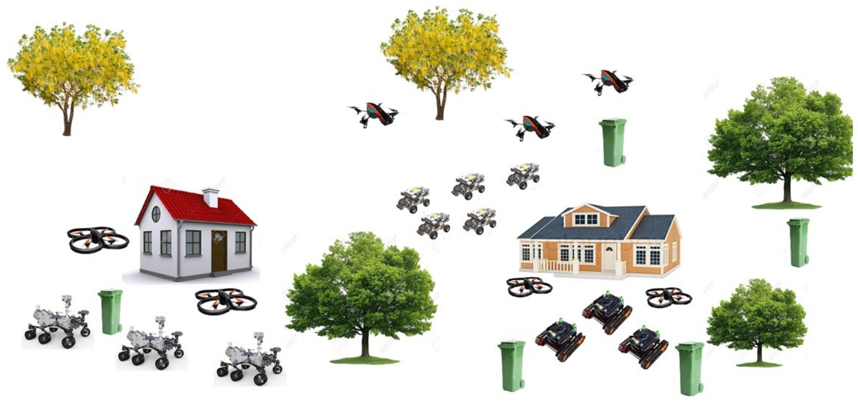


**Figure 1.1:** Large-scale homogeneous/heterogeneous autonomous navigation system.

harmful to environmental property. An autonomous robot may also adapt to changing surroundings. Therefore, the mobile robots need to have the capabilities of autonomy and intelligence, and to design algorithms that allow the agents to function autonomously in unstructured, unknown, and GPS-denied environments. The autonomous navigation system uses information from various subsystems to accomplish four essential tasks: (*i*) to estimate the position and orientation of the mobile agent (Localization); (*ii*) to identify obstacles in the surroundings and operate in order to avoid them (Obstacle detection and avoidance); (*iii*) to generate reasonable trajectory to the goal position (Path planning) and (*iv*) to regulate its motion to accomplish the desired trajectory (Motion control). This poses a challenge to solve key issues such as uncertainty (in both sensing and action), reliability, and real-time response. Mobile agents are inherently uncertain about the state of their environments. An uncertainty occurs from sensor limitations, noise and the fact that the real-world environments are unpredictable. The uncertainty requests the development of flexible approaches in navigation [15, 16].

Figure 1.1 presents a large-scale heterogeneous autonomous navigation system in a real life application. This large scale system includes six homogeneous subsystems which may have different dynamic characteristics and unique working environments. Therefore, it is necessary to design suitable autonomous navigation strategies, especially path planning algorithms in different scenarios, for different mobile platforms. Consequently, this thesis concentrates on designing multi-objective optimal/real-time path plannings in cluttered or dynamic environments. In each situation, based on the various purposes, the system's characteristics, the working space's features, and the user's experiences, a proper methodology is proposed to fulfill the robot's missions.

Firstly, the robot path planning is investigated in static environments that highly mimic the skills of expert operators (Chapter 3). Secondly, the multi-objective optimization path planing problem considering the constraints of the mobile agents is studied in cluttered environments (Chapter 4). Lastly, the autonomous navigation approach is developed in unpredictable environments, thus enriching the value of the path planning algorithms in practical applications (Chapter 5, Chapter 6).

## 1.2 Motivation

All robotic systems have complex dynamics and the requirements on the performance of the autonomous navigation strategies are usually quite demanding. The main motivation behind this work is the development of fully autonomous low-cost navigation processes using on-board sensors, which effectively solve the localization, sensing environment, path planning and control problems in different scenarios.

As path planning is a core function for an autonomous robot, this problem is deeply investigated in all chapters and it is the main contribution of this thesis. The path planning in a diversity of environments, such as static (Chapter 3), cluttered (Chapter 4) and unknown/dynamic (Chapter 5, Chapter 6) environments are considered. In each application, a proper solution is proposed to fulfill the robot's assignments based on the dynamic characteristics, as well as the working spaces. Firstly, the path planning is studied in an application that highly desires to capture the skills of an expert operator and uses his knowledge to develop an autonomous strategy (Chapter 3). Secondly, the multi-objective optimization path planing problem considering the constraints of the mobile agents is also investigated (Chapter 4). Lastly, this thesis studies this topic in unpredictable environments (Chapter 5, Chapter 6).

An indispensable aspect of the mobile platform's localization is how to fuse data from different sensors to improve the common estimation accuracy. The data fusion algorithms are key issues and their performances have a strong interdependence with the performance accuracy of the agent position in a working environment. It is an important requirement that the robust method should be validated in use.

Furthermore, the mobile platform includes cameras and other sensors to capture information about the environment. The image processing is able to provide better information about the surroundings. Thus, this algorithm is developed during the observation process .

As UGVs/ UAVs are complex systems, in which electromechanical dynamic characteristics are involved, a robust controller is a challenging task to explore.

For UAVs, autonomous landing is an essential requirement. Hitherto, UAVs have to land on prepared area with prior knowledge of obstacle-free trajectories. This thesis provides an original approach for autonomous landing in presence of uncertain obstacles.

The challenges and proposed solutions of the localization, perception and control approaches are validated via real-time experiments in Chapter 5 and Chapter 6.

## 1.3 Organization and Main Contributions

This dissertation is organized as follows.

### Chapter 2: Literature Review

This chapter introduces the state-of-the-art autonomous navigation in the recent years, especially in path planning. The first part describes the big picture of autonomous robots, presenting the main achievements, the potential applications and the main technical problems. The second one summarizes the different approaches. The third one explains what the planning problem is, discusses the various methods of planning and presents research works related to the problem of partial observation ability. The advantages and disadvantages of each method are investigated.

This chapter concludes that many studies have been implemented in autonomous navigation but very few of them have reported convincing experimental results. The current robot path planning algorithms have not yet reached the level of robustness and reliability required for real-world applications. Autonomous navigation in unknown/dynamic environments is still a challenge and indispensable aspect for researchers.

The results presented in this chapter have been published in:

T. T. Mac, C. Copot, T. D. Tran, R. De Keyser, Heuristic Approaches in Robot Path Planning: A Survey, *Robotics and Autonomous Systems*, vol. 86, pp.13-28, 2016. (A1, rank: Q1, impact factor: 1.950).

## Chapter 3: Fuzzy-Based Autonomous Navigation of a Mobile Robot in Outdoor Environments

In this chapter, a robot path planning problem in an unknown static environment is developed. Fuzzy-based autonomous navigation is introduced as an intelligent means of a mobile robot. It is considered for expressing the subjective uncertainties in the human mind. The human has a noticed ability to perform navigation tasks without any exact measurements or computations. In the FC algorithm, the decision making of robot navigation can be phrased by a set of IF-THEN rules. To allow a straightforward implementation, the navigation problem is divided into simpler tasks and behaviors.

Given this need for designing safe path planning that highly desires to capture the expertise of an operator and use his knowledge to develop an autonomous strategy, this chapter concentrates on designing a practical adaptation mechanism to make an intelligent autonomous system accomplish this goal.

Since an autonomous geophysical measurement platform (AGMP) is extremely expected to use the archaeologist's knowledge, it is an ideal case study for this chapter. To account for a situation of the AGMP working in outdoor environments with different terrains such as smooth, rough or rocky ones, the AGMP should also be able to adjust its speed with changes in terrain. The proposed fuzzy-based speed control strategy has major advantages such as: *i*) the fuzzy logic rules that lead the AGMP's motion are simpler and more understandable and *ii*) the behavior-based strategy has a modular structure that can be extended to incorporate new behaviors.

The results presented in this chapter have been published in:

1. T. T. Mac, C. Copot, R. De Keyser, T. D. Tran, T.Vu, MIMO Fuzzy Control for Autonomous Mobile Robot, *Journal of Automation and Control Engineering*, Vol. 4 (1), pp. 65-70, 2016.
2. T. T. Mac, C. Copot, L. Verdonck, R. De Keyser, Speed control strategy of geophysical measurement platform for archaeological prospection: a conceptual study, *IEEE International Conference on Systems, Man, and Cybernetics*, Budapest, Hungary, pp. 3748 - 3753, 2016. (P1-ISI)
3. T. T. Mac, C. Copot, R. De Keyser, T. D. Tran, T.Vu, MIMO fuzzy control for autonomous mobile robot, *7th International conference on Computer Research and Development*, Ho Chi Minh, Vietnam, pp. 277- 282, 2015.

## Chapter 4: Hierarchical Global Path Planning Approach on Multi-Objective Particle Swarm Optimization

In Chapter 4, a multi-objective optimal path taking into account the abilities of mobile robots in a cluttered environment is introduced. The quality of the robot's pathway is one of most important aspects in autonomous navigation applications. However, planners, such as fuzzy control or neural networks generate paths that are not optimal. From literature review in Chapter 2, one recognizes that each of the robot path planning approach has its own limitations and so far, one individual method cannot perfectly solve the robot path planning problem. Therefore, integrated methods are better choices for this task. The aim of those methods is to figure out an optimal collision-free path from a starting position to a goal position under certain constraints. As there are many types of robots with different characteristics, constraints and applications, it is nearly impossible to introduce an exact definition for the term "optimal path". However, it can be focused on several aspects, such as the safety, the smoothness, the shortest distance and the less energy consumption with respect to constraints of changing direction and velocity.

Consequently, this chapter proposes a novel hierarchical approach consisting of: (*i*) a triangular decomposition; (*ii*) a Dijkstra's algorithm and (*iii*) a proposed constrained multi objective particle swarm optimization (CMOPSO) with an accelerated update methodology based on Pareto dominance principle. The considered problem is the global path planning of a mobile robot evolving in cluttered environments. The purpose of the approach is to obtain optimal collision-free paths, focusing on minimizing the path length and maximizing the smoothness considering the mobile robots' constraints.

The results presented in this chapter have been published in:

1. T. T. Mac, C. Copot, T. D. Tran, R. De Keyser, A Hierarchical Global Path Planning for Mobile Robot based on Multi Objective Particle Swarm Optimization, *Applied Soft Computing*, vo. 59, pp. 68-76, 2017. (A1, rank: Q1, impact factor: 3.541).
2. T. T. Mac, C. Copot, T. D. Tran, R. De Keyser, A Hierarchical Global Path Planning based on Multi-Objective Particle Swarm Optimization, *IEEE 21st International Conference on Methods and Models in Automation and Robotics*, Miedzyzdroje, Poland, 29th August-1st September, pp. 930-935, 2016. (P1-ISI).

## Chapter 5: Potential Field Method of an UAV's Autonomous Navigation

A large number of studies have emerged in the literature on UAVs. Some examples of their applications can be found in precision agriculture, formation control of UGVs using UAVs and habitat mapping. The current UAVs' implementations still require collision avoidance, adaptive path-planing and optimal control techniques. There exists a need to design methodologies to cope with these requirements, to increase the degree of intelligence, and therefore, the autonomy of UAVs.

In Chapter 5, an adapted path planning algorithm in unknown/dynamic environments is studied. Besides, a real-time implementation for an Ar. Drone 2.0 autonomous navigation is proposed to trigger its identification, localization, detection of obstacles, path planning and optimal control processes. The novelties are: (*i*) a position-estimation method based on sensor fusion method; (*ii*) a solution of path planning in unknown/dynamic environment and (*iii*) multi-objective optimization for PID controllers based on a proposed multi-objective particle swarm optimization (MOPSO) with an accelerated update methodology.

The results presented in this chapter have been published in:

1. T. T. Mac, C. Copot, T. D. Tran, R. De Keyser, C. M. Ionescu, The Development of an Autonomous Navigation System with Optimal Control of an UAV in Partly Unknown Indoor Environment, *Mechatronics*, vol. 49, pp. 187-196, 2018. (A1, rank: Q1, impact factor: 2.496).
2. T. T. Mac, C. Copot, T. D. Tran, R. De Keyser, Ar.Drone UAV control parameters tuning based on particle swarm optimization algorithm, *IEEE international Conference on Automation, Quality and Testing, Robotics*, Cluj-Napoca, Romania, May 19-21, 2016. (P1-ISI).
3. T. T. Mac, C. Copot, A. Hernandez, R. De Keyser, Improved Potential Field Method for Unknown Obstacle Avoidance Using UAV in Indoor Environment, *IEEE 14th International Symposium on Applied Machine Intelligence and Informatics*, January 21-23, Herlany, Slovakia, pp. 345-350, 2016. (P1-ISI).
4. C. Copot, A. Hernandez, T. T. Mac, R. De Keyser, Collision-Free Path Planning in Indoor Environment Using a Quadrotor, *IEEE 21st International Conference on Methods and Models in Automation and Robotics*, Miedzyzdroje, Poland, 29th August-1st September, 2016. (P1-ISI).

## Chapter 6: UAV Autonomous Landing in Presence of Uncertain Obstacles and GPS-Denied Environment

Autonomous landing is the essential step after autonomous navigation of an UAV. In order to land on a location at which the safety requirement is not confirmed (e.g., presence of static obstacles or moving obstacles), the UAV should immediately detect the environment through its sensors and decide its actions for landing.

In Chapter 6, an optimal landing algorithm for the UAV in unknown/dynamic environments is proposed. Additionally, there was developed an algorithm to estimate moving obstacle's velocity based on vision system. This velocity information assists the UAV to effectively perform the landing task. The main contributions of this chapter are:

- a low-cost, high-scalability and easy-to-use navigation system is developed to assist the UAV's landing process.
- a sensor fusion technique is investigated for localization process.
- an estimation of moving obstacle's velocity based on vision approach is proposed to interpret the landing task in complex situations.

The results presented in this chapter have been published in:

T. T. Mac, C. Copot, C. M. Ionescu, A Vision Based Approach for UAV Autonomous Landing in Presence of Uncertain Obstacles and GPS Denied Environment, *IEEE/ASME Transactions on Mechatronics*, revision. (A1, rank: Q1, impact factor: 4.357)

## Chapter 7: Concluding Remarks and Perspectives

This chapter summarizes the contributions made by this thesis and outlines directions for future research.

# Chapter 2

# Literature review

Autonomous navigation is one of the most important requirements of an intelligent vehicle, and a process designed towards a target position while avoiding obstacles. There are four basic components of this process [13]: i) *perception*, the robot uses its sensors to extract meaningful information; ii) *localization*, the robot determines its location in the working space; iii) *cognition and path planning*, the robot decides how to steer to achieve its goal and iv) *motion control*, the robot regulates its motion to accomplish the desired trajectory. Among them, the path planning is the most challenging task, therefore, it brings attention to researchers.

This chapter has reviewed the current-state-of-the art heuristic approaches and potential field method in robot path planning. The heuristic methods analyzed in this chapter consist of neural network, fuzzy logic and nature-inspired methods (genetic algorithm, particle swarm optimization and ant colony optimization). The strengths and drawbacks of each algorithm are discussed.

The results of this chapter are published as follows.

T. T. Mac, C. Copot, T. D. Tran, R. De Keyser, Heuristic Approaches in Robot Path Planning: A Survey, *Robotics and Autonomous Systems*, vol. 86, pp.13-28, 2016. (A1, rank: Q1, impact factor: 1.950).

## 2.1 Introduction

Path planning of a robot can be considered as a sequence of translation and rotation actions from a starting position to a destination while avoiding obstacles in its working environment. There are two suggested techniques covering all approaches in robot path planning algorithms: i) *global path planning* or off-line path planning and ii) *local path planning* or on-line path planning [17][18]. A global path planner usually generates a low-resolution high-level path based on a known environmental map or its current and past perceptive information of the environment. The method is valuable for producing an optimized path; however, it is inadequately reacting to unknown or dynamic obstacles. On the other hand, local path planning algorithm does not need *a priori* information about the environment. It usually gives a high-resolution low-level path only over a fragment of global path based on information from on-board sensors. It works effectively in dynamic environments. The method is inefficient when the target is long distance away or the environment is cluttered. Normally, the combination of both methods is advised to enhance their advantages and eliminates some of their weaknesses [19–21]. The robot path planning problem can be divided into classical methods and heuristic methods [1, 22] as shown in Figure 2.1.

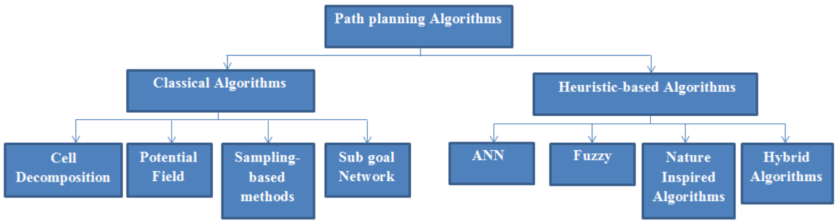


**Figure 2.1:** The classification of robot path planning algorithms.

The most important classical methods consist of cell decomposition method (CD), potential field method (PFM), subgoal method (SG) and sampling-based methods. In the Cell Decomposition method, the free space of the robot's configuration is divided into small regions called "cells". The goal is to provide a collision-free path to reach the target. The applications of this approach can be found in [23][24]. In potential field method, the obstacles and the goal are assigned repulsive and attractive forces respectively so that the robot is able to move towards the target while avoiding the obstacles [25]. A new formula of repelling potential is performed in the interest of reducing oscillations and avoiding conflicts when obstacles locate near a target [26]. Instead of a single robot, the method is also extended successfully to navigate multi robots to perform complex tasks [27][28]. D.H. Kim proposed a new framework to escape from a local minimum location of robot path based on PFM [29] [30]. To solve path planning problem in dynamics environments, modifications on classical PFM are introduced in [31, 32]. The Sub-goal method uses a list of reachable configurations from the starting position to the goal position while avoiding all the obstacles. The sub-goal method applications for robot navigation are presented in [33–35].

Planning schemes based on sampling-based motion planning (SBP) algorithms have received considerable attention due to their capability in complex and/or time-critical real-world planning problems. Arguably, the most persuasive SBP methods to date include probabilistic road-map (PRM) and rapidly-exploring random trees (RRT) [36]. Although the idea of connecting points randomly sampled is fundamental in both approaches, these two methods are different in the manner that they construct a graph connecting the points [37]. A comprehensive survey of work in SBP is given in [38]. PRM algorithm has been recorded to be well implemented in high-dimensional state spaces. The PRM is created by curves or straight lines that enable the robot to go anywhere in its free space. The two well-known road-map methods, namely, visibility graph (VG) and Voronoi diagram (VD) have achieved very good results with dramatically different types of roads. A visibility graph is a graph that comes as close as possible to the obstacles. As a result, the shortest path is found by applying this method; however, the path touches obstacles at the vertices or edges and thus is dangerous for the robot. Contrary, Voronoi diagram creates a road that tends to maximize the distance between the robot and the obstacles. Therefore, the obtained paths based on Voronoi diagram are not optimal with respect to the path length. The advantage of this method is that only a limited number of sensors is used in the robot navigation task. Path planning of a robot swarm using road-map technique is proposed in [39–41]. Several improvements are proposed by [42] [43]. RRT has received a considerable amount of attention, because of its computational efficiency and effectiveness and its ability to find a feasible motion plan relatively quickly, even in high-dimensional space [44][45].

In [46][47], the navigation approach, which consists of four separate modules: localization, path planning, path execution and obstacles avoidance, is proposed for autonomous urban service mobile robots. To avoid obstacles, the authors combine a local planner with a slightly modified dynamic window method. The local planner is implemented using RRT. RRT explores the robot's working space by incrementally building a tree, creating new branches by randomly generating points and linking them to the closest point for which an obstacle-free path is obtained. A problem in RRT is that it produces a path with many branches in the work space by using the randomized technique. To overcome this problem, a novel path planning approach for a mobile robot in dynamic and cluttered environments with kinodynamic constraints is presented in [48]. The algorithm called Heuristic Arrival Time Field biased Random Tree (HeAT-RT), that takes advantage of the high-exploration ability of a randomized tree, is combined with an arrival time field and heuristics to achieve the path optimality, safety, and applicability to the real robot. Instead of choosing a random point from the entire work space, like the basic RRT algorithm does, this approach selects a random point using the bias from the arrival time field so that the tree grows in a favorable direction towards the target. The kinodynamic RRT*, an incremental sampling-based approach for asymptotically optimal motion planning for robots with linear dynamics, is introduced in [49].

The ability of SBP to provide valid paths for constrained high dimensional problems is advantageous. Despite the hit-or-miss sampling approach being the core of the SBP's effective strategy, it leads to the inclusion of many redundant maneuvers in the obtained path. In [50], a modification of the termination condition is proposed in a way such that the SBP keeps running to interactively converge the path cost. The solution convergence remains an unanswered problem; until it is proven that given infinity run-time RRT will not achieve an optimal path [51]. Recently, a family of optimal SBP, RRT*, PRM* and RRG* is introduced to guarantee asymptotic optimality. Despite their effectiveness, they provide no theoretical guarantees for reaching an optimal solution [38].

Many efforts have been made to apply classical approaches onto real-time motion planning [52–56]. Incremental algorithms to update distance maps, Voronoi diagrams, and configuration-space collision maps are presented in [52]. The representations are initialized by using a given grid map or point cloud. For efficient on-line applications, only the cells that are affected by changes in the environment are updated. Therefore, these algorithms can be used in real-world scenarios with unexpected or moving obstacles. Another practical approach to solve the limitations of the road-map based mobile robot path planner in a home environment is introduced in [53]. The proposed approach incrementally constructs a hierarchical road-map which has a multi-layered structure using low cost sonar sensors. Motion planners based on RRT algorithms for multi-agent system, omni-directional robots and an autonomous vehicle in an urban environment are respectively proposed in [54–56]. However, classical approaches do not produce optimal paths and tend to be locked in some local minima. Moreover, some of them may not provide the suitable solution in the presence of multiple obstacles or in dynamic environments. In order to avoid the inefficiency of the classic methods, the heuristic approaches, mainly due to increased computing power for processing large data sets, are employed.

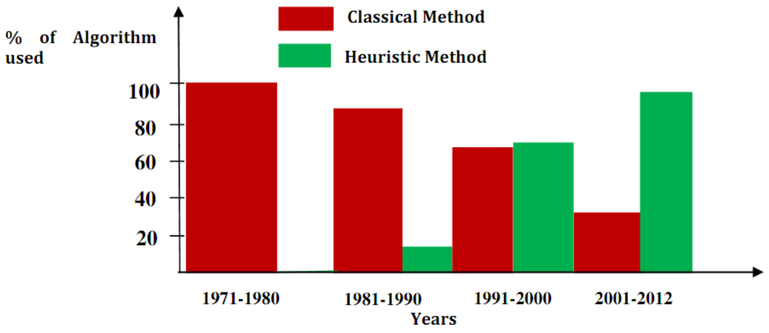


**Figure 2.2:** Application of the classical and heuristic algorithms [1].

The heuristic methods analyzed in this paper consist of neural networks (NN), fuzzy logic (FL), nature-inspired and hybrid algorithms. Neural networks have been successfully applied in many robot path planning applications due to their advantages, such as nonlinear mapping, learning ability, parallel processing [3, 4, 57–64]. In fuzzy logic algorithm, robot navigation is based on a set of IF-THEN rules. In [5, 65–74], different approaches based on fuzzy logic are introduced to solve robot path planning problems. Methods of integrating fuzzy inference systems and neural networks to achieve ability of human thinking (fuzzy logic) and the ability of learning (neural networks) are studied in [7, 75–77]. Other approaches are inspired by biological behaviors, which become more and more popular in robot path planning applications. A chronological order of the most popular nature-inspired methods is provided in Table 2.1 [12]. Some widely known algorithms inspired by biological behaviors have been successfully applied in robot path planning, for instance, Genetic Algorithm (GA) [8, 78–85], Particle Swarm Optimization (PSO) [9, 86–92], Ant Colony Optimization (ACO) [10, 93–98].

| **Nature-inspired methods in chronological order** | | | |
|---|---|---|---|
| **Year** | **Nature-inspired methods** | **Year** | **Nature-inspired methods** |
| 2012 | Keill Herd | 2002 | Estimation of Distribution Algorithm |
| 2010 | Bath Algorithm | 1995 | Particle Swarm Optimization |
| 2010 | Artificial Bee Algorithm | 1992 | Ant Colony Optimization |
| 2009 | Cuckoo Search Algorithm | 1989 | Tabu Search |
| 2009 | Gravitational Search Algorithm | 1983 | Simulated Annealing |
| 2007 | Firefly Algorithm | 1979 | Cultural Algorithms |
| 2007 | Intelligent water drops | 1975 | Genetic Algorithms |
| 2005 | Harmony Search Algorithm | 1966 | Evolutionary Programming |
| 2005 | Honey Bee Algorithm | 1965 | Evolution Strategies |
| 2002 | Bacterial Foraging Algorithm | | |

**Table 2.1:** Nature-inspired methods in chronological order [12].

To perform effectively the mentioned algorithms, it is important to take into account the hardware characteristics. The better the hardware is used, the more accurate data is obtained. There are many types of hardware which were used in the mentioned studies. In [3], the AURO robot was built up as a prism platform with three-wheeled configuration, which has a length of 850 mm, a width of 500 mm, and a height of 750 mm. It consists of a single steerable drive wheel at the front and two passive rear wheels. The drive wheel, as well as the passive wheels are equipped with shaft encoders used for odometry measurement. For sensing the environment, it uses an ultrasonic scanning range finder, rotating from $0^o$ to $360^o$ and three tactile sensors on the bumper. The robot obtains range images by an ultrasound scanning range finder. The ranges for desired angular sector are obtained in $N$ steps, covering up to $180^o$ arcs in front of the robot and are measured by scanning every 200 ms by time-of-flight principle. In [99], two small robots - Khepera are used to validate the performances of a kernel smoothing based reinforcement learning methodology. Each robot has 5 ultrasonic sensors which are placed around its body. These sensors are used to reconstruct the odometry and detect the natural features of the environment. A Turtlebot2 mobile robot, which was equipped with UTM-30LX LIDAR device, is investigated in [100]. Other UGV's hardware platforms can be found in [13].

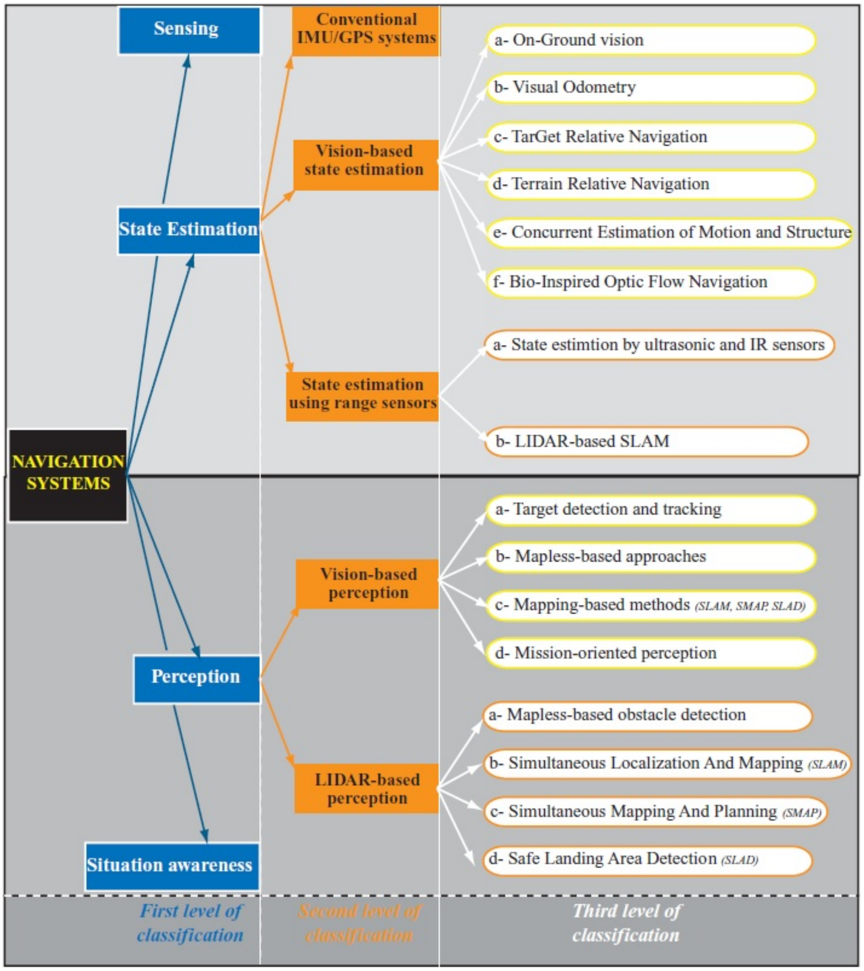

Sensing
Conventional IMU/GPS systems
a- On-Ground vision
b- Visual Odometry
c- TarGet Relative Navigation
d- Terrain Relative Navigation
e- Concurrent Estimation of Motion and Structure
f- Bio-Inspired Optic Flow Navigation
Vision-based state estimation
State Estimation
State estimation using range sensors
a- State estimtion by ultrasonic and IR sensors
b- LIDAR-based SLAM
NAVIGATION SYSTEMS
a- Target detection and tracking
b- Mapless-based approaches
c- Mapping-based methods (SLAM, SMAP, SLAD)
d- Mission-oriented perception
Vision-based perception
Perception
a- Mapless-based obstacle detection
b- Simultaneous Localization And Mapping (SLAM)
c- Simultaneous Mapping And Planning (SMAP)
d- Safe Landing Area Detection (SLAD)
LIDAR-based perception
Situation awareness
First level of classification
Second level of classification
Third level of classification

**Figure 2.3:** Classification of navigation systems developed for UAVs [2].

Regarding to UAVs family, conventional and basic navigation sensors include state estimation sensors for flight control, such as an IMU (three gyroscopes, three accelerometers, and three magnetometers) for attitude estimation, a global navigation satellite system (GNSS) for position and velocity estimation, and an altimeter (barometric, LIDAR, radar) for enhancing the height estimation. Indeed, lightweight integrated IMU/GPS/altimeter devices (e.g., Xsens MTi-G 68 g, sbg IG-500N 45 g, Microstrain 3DMGX3- 35 23 g) with acceptable accuracy and affordable prices are available today. For perception, ranging sensors and cameras are the most used sensors. Cameras or electro-optic sensors are a popular approach for environment sensing because they are light, passive, compact and provide rich information about the vehicle's self-motion and its surrounding environment. Computer vision is a popular solution for applications that require target detection and tracking, landmark recognition, and other tasks. There exist different types of imaging sensors, such as single cameras, stereo cameras, omnidirectional cameras, and optic flow sensors. The drawback of camera-based approaches is their sensitivity to the ambient light and scene texture. Furthermore, the complexity of image processing algorithms makes their real-time implementations on embedded microprocessors challenging.

LIDAR (light detection and ranging) is a suitable device for mapping and obstacle detection, because it directly measures the range by scanning a laser beam in the environment and measuring distance through time of flight or interference. Other common terms for LIDAR are LADAR (laser detection and ranging) and laser radar, which are often used in military contexts. LIDAR does not rely on ambient lighting and scene texture. However, LIDAR units are heavier than cameras and energy-consuming (active), and most available off-the-shelf systems use a single-line scan (2D LIDAR). For 3D navigation, these 2D LIDAR units have been mounted on sweeping or spinning mechanisms when used on UAVs and UGVs.

Radar is the sensor of choice for long-range collision and obstacle detection in larger UAS and manned aircraft. Radar provides nearly all-weather, high-resolution and broad-area imagery. The problem with radar is that it requires a lot of power and the localization of the beam requires a large antenna. Radar units are also heavier than previous sensors, which makes their integration into small UAVs difficult. Some lightweight radar systems are available today, but they are expensive, such as the miniature radar altimeter (MRA) Type 1 from Roke Manor Research Ltd., which weighs only 400 g and has a range of 700 m [2]. Figure 2.3 presents the existing navigation systems developed for UAVs and classify them into categories, groups and classes based on the autonomy level they provide (first level of classification), the sensing technology used (second level of classification) and the algorithmic approach or method used (third level of classification).

Based on the development of new hardware and computing power, the classical methods dominated the field of robot path planning before 2000, but they have much lower percentage of usage after that year. It is also found that heuristic approaches are more and more popular in robot navigation field compared to classical techniques, as shown in Figure 2.2. Therefore, this chapter concentrates on heuristic-based algorithms for the robot path planning problems(section 2.2). In addition, one of the most attractive classical methods, named potential field method, is also considered due to the good results (section 2.3). Their advantages and drawbacks are discussed, then the applications are analyzed. The discussion, conclusion and future outline of robot path planning are provided in section 2.4.

## 2.2 Heuristic-Based Algorithms

This section covers neural network (NN), fuzzy logic (FL), some of the most common nature-inspired algorithms which are GA, PSO, ACO, and their applications in the robot path planning domain. The positive and negative aspects of the algorithms are available in each subsection. In the beginning of subsections GA, PSO, ACO, a brief introduction of the basic algorithm is provided. This description is considered to be necessary in order to comprehend the use of the specific method for robot path planning performances. The section is organized in four parts. Neural network (NN) is presented in 2.2.1 and fuzzy logic (FL) is showed in 2.2.2. After that, the hybrid algorithm integrating neural network and fuzzy logic (NN-FL) technique is introduced in 2.2.3. Nature-inspired algorithms are lastly analyzed in 2.2.4. Other hybrid algorithms are integrated in each subsection.

### 2.2.1 Neural Network

Recently, the neural network has been developed in many robot path planning applications [3, 4, 57–64, 101]. NN is employed to model complex relationships between inputs and outputs in the robot navigation task.

The implementation of NN in robot navigation is categorized into three categories: *i*) *interpreting the sensory data*, *ii*) *obstacle avoidance* and *iii*) *path planning* [101]. S.X. Yang and M. Meng [57] have proposed robot path planning using a neural network in a non-static environment. Two testing scenarios are surveyed: a well-known beam robot competition micro mouse maze and an environment of both moving target and moving obstacles. The study shows that the robot travels on a continuous smooth route to catch the moving target while avoiding moving obstacles. The combination of the Lyapunov stability theory and qualitative analysis guarantee the system's stability. However, the sizes of mobile robot, moving obstacles and target are not considered in the simulations. Moreover, the robot and moving obstacles are assumed to have the same constant velocity, thus this methodology might confront some difficulties when executed in real systems. A hierarchical architecture which constructs internal models of a robot environment for goal-oriented navigation by an imitation learning process is proposed in [58]. This method is based on the Reservoir Computing paradigm for training Recurrent Neural Networks (RNN). It consists of two randomly generated RNNs, one for modeling the localization ability and one for learning the navigation skill. The experimental results on a simulated robot indicate that the trained system can localize the robot in large unknown environments and navigate successfully to the desired goals.

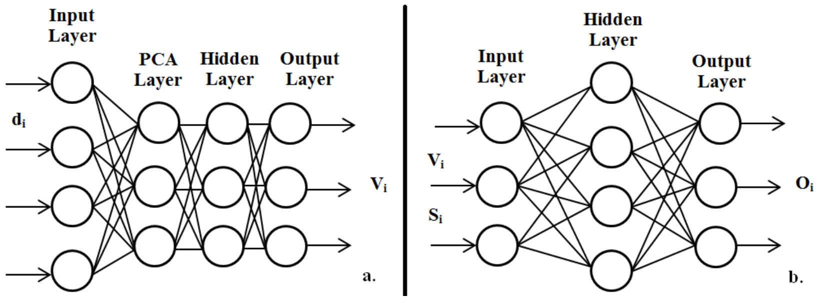


**Figure 2.4:** Topology of PCA (a) and MLP (b) network[3].

From another point of view, D. Janglova [3] constructed collision-free path based on two neural networks: principal component analysis (PCA) and a multiplayer perception (MLP), in Figure 2.4a, 2.4b respectively. PCA is an unsupervised linear procedure which finds uncorrelated features from the inputs. It is a data reduction method which converts the input data to a few principal components. The number of principal components is a compromise between training efficiency and the accuracy of the results. The PCA network learns by generalized Hebbian rule. In the beginning, PCA network is performed to decide the safe space (outputs—$V_i$) using data from ultrasound range finder (inputs—$d_i$). Hence, the robot has sufficient information about its distances to all the objects from the work space. The knowledge about free segments, $V_i$ are obtained as the outputs for the first neural network. Those values together with the goal segments $S_i$ are used as inputs for the second neural network. The outputs of the second neural network are suitable directions $O_i$ of the robot which will be sent to the controller. The network includes 18 input neurons, 20 hidden neurons and 9 output neurons $O_i$. The advantages of this approach are the ease of the implementation and the approximation of any inputs/outputs map. However, the disadvantages are represented by a low speed and a huge amount of samples for the training process.

S.H. Dezfoulian et al. [59] inherited the study done by [3] for determining a free space in a robot working environment. The algorithm is performed on three robots using different range sensors. Although there are differences in the journeys, all three mobile robots can successfully navigate from the starting location to the goal location. Hence, it promises to extend the algorithm to real system using low cost sensors. However, the main drawback of this method is the large amount of training samples (3000 training patterns) for the robots need to gather in different scenarios.

Further, M.K. Singh et al. [4, 60, 61] designed a neural network algorithm that enables the robot to move safely in an unknown environment with static/dynamic obstacles. The inputs of the proposed neural network consist of left, right, and front obstacle distance (LOD, FOD, ROD respectively); the angle between the robot, the target (TA) and the output represents the steering angle of the robot as depicted in Figure 2.5. The information is received from an array of sensors. A four-layers neural network is applied to solve the optimization problem of the path planning. Compared with [59], the neural network is trained by presenting only 200 patterns typical scenarios. Therefore, that algorithm dramatically reduces the gathering of training samples, as well as the time of computation.

M.A. Sagban et al. nominated a new neural network approach based on the

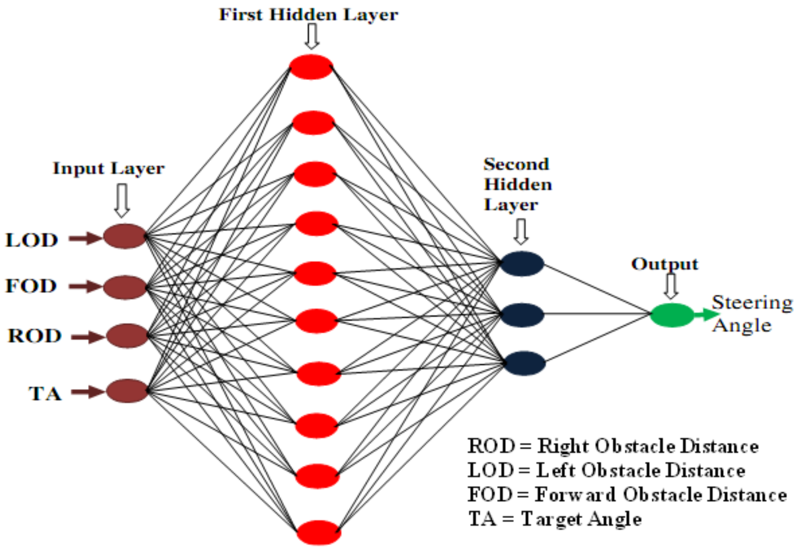


**Figure 2.5:** Neural Network structure for the robot path planning [4].

reactive navigation algorithm in unstructured indoor environments [62]. The method integrated off-line and on-line learning stages to reach optimum performance. In [63], a NN back-propagation algorithm is applied to train the robot on-line so that to make it capable of avoiding moving obstacles. Expanding the technique in [62] for formation navigation of multiple mobile robots, an adaptive NN control is proposed in [64]. The algorithm is based on Lyapunov function, graph theory and PSO algorithm. First, the method generates successive paths for the leader robot according to current obstacles perceived. Then, the followers use that information to successfully perform their navigation. This method conducts quite good results, both in simulations and in real experiments.

In [99], the classical multi-agent reinforcement learning algorithm is modified such that it does not need the unvisited state. The neural networks and kernel smoothing techniques are applied to approximate greedy actions by estimating the unknown environment. A method which combines Potential Field Method (PFM) and fuzzy-neural for the real-time globally optimized path planning in a dynamic environment is presented in [102]. This method used fuzzy control to establish the space model, then combined PFM and fuzzy-neural to search for the shortest path and smoothly avoid the obstacles in a dynamic environment. Table 2.2 condenses the above reviewed papers.

In a summary, NN has been widely used to implement motion planners for autonomous robots. However, NN has some drawbacks. For instance, a neural network is usually time consuming. Moreover, the learning algorithm may not be able to guarantee the convergence to an optimal solution. Therefore, the integrated methods are more appropriate for addressing the robot navigation problem in real-world applications.

| **Applications of NN for Robot Path Planning** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Approach | Authors | Kinematic model | Obstacle shape | Static Obstacle | Dynamic Obstacle | Static Target(s) | Dynamics Target(s) | Adapted velocities | Simulation | Real system | Year |
| NN | S. X. Yang [57] | N | Rectangle | Y | Y | N | Y | N | Y | N | 2000 |
| | E. Antonelo [58] | N | Arbitrary | Y | N | Y | N | N | Y | N | 2010 |
| | D. Janglova[3] | Y | Arbitrary | Y | N | Y | N | N | Y | Y | 2004 |
| | S. H. Dezfoulian [59] | N | Arbitrary | Y | N | Y | N | N | Y | N | 2013 |
| | M. K. Singh [4] [60] | Y | Arbitrary | Y | N | Y | N | N | Y | Y | 2009 |
| | M. K. Singh [61] | Y | Arbitrary | Y | N | Y | N | N | Y | Y | 2011 |
| | M. Sagban [62] | Y | Arbitrary | Y | N | Y | N | N | Y | Y | 2012 |
| | I. Engedy [63] | Y | With markers | Y | Y | Y | N | Y | Y | Y | 2009 |
| NN/PSO | X. Chen [64] | Y | Circle | Y | Y | Y | Y | Y | Y | Y | 2006 |
| NN | D. L. Cruz [99] | Y | Rectangle | Y | N | Y | N | N | Y | Y | 2017 |
| NN/FL/PFM | Q. Q. Wu [102] | Y | Arbitrary | Y | Y | Y | N | N | Y | N | 2017 |

**Table 2.2:** Applications of NN technique for the robot path planning.

## 2.2.2 Fuzzy Logic Technique

Besides the neural network, the fuzzy logic is considered for expressing the subjective uncertainties in human mind. The human has a noticed ability to perform navigation tasks without any exact measurements or computations. It is highly desirable to mimic this ability to develop autonomous robot navigation strategies [65]. In the fuzzy logic algorithm, the decision making of robot navigation can be phrased by a set of IF-THEN rules. To easily implement it, the navigation problem is broken into simpler tasks and behaviors.

H. Chang and T. Jin introduced a fuzzy inference model to solve the mobile robot path planning problem [5]. The locations of the goal/obstacles and current speed of the robot in unknown dynamic environments are perceived from the sensors. In that model, three major navigation goals: target orientation (Seek Goal), obstacle avoidance (Avoid Obstacle) and rotation movement (Maintain Heading) are included in a cost function to find the optimal steering angle $\theta$. The mobile robot intelligently navigates by varying the weights of the cost functions depending on the environment (Figure 2.6).

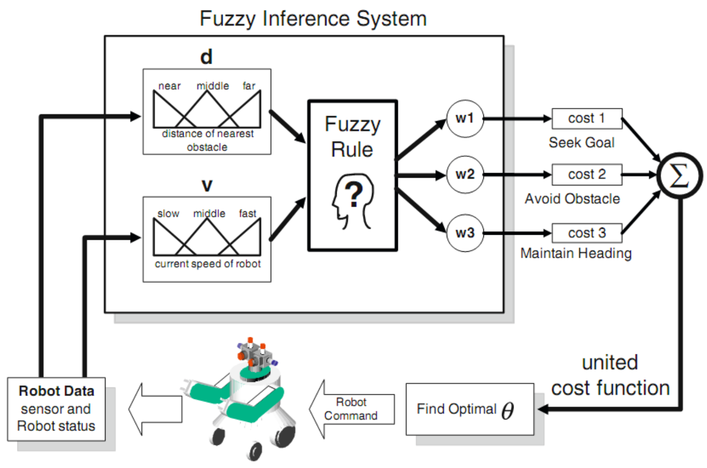


**Figure 2.6:** Structure of fuzzy inference system for the robot path planning [5].

In [66], V.M. Peri et al. have solved multi-robots navigation problem by using a fuzzy logic controller and a Petri net. The fuzzy rules navigate the robot in the working space according to the obstacles distribution or targets position. Each controller of the mobile robot enables it not only to avoid obstacles in a cluttered environment but also to avoid other mobile robots. Hence, the robot can perform its task and incorporate with other robots by implementing the collision prevention rules. In [67], J.H. Lilly suggested a different approach, utilizing both negative fuzzy rules and traditional positive rules. In this method, the negative fuzzy rules determine movements to be avoided rather than being performed. A positive rules base is stated to drive the robot to the goal in the absence of obstacles, while a negative rules base is activated in the presence of obstacles. As a result, fewer rules compared to the use of the solely positive rules are applied on the obstacle avoidance controller.

A fuzzy logic system with 48 fuzzy rules is designed in [68], which comprises of three behaviors: target seeking, obstacle avoidance and barrier following. A combination of multiple sensors is used to sense the obstacles near the robot, the target location and the current speed of the robot. Thus, the mobile robot is able to preferably "see" its environment and avoid static/dynamic obstacles autonomously. It can generate reasonable paths towards the target in various scenarios without suffering from the "symmetric indecision" and "dead cycle" problem. A novel terrain traversability analysis method for a mobile robot navigation process is presented in [69]. The robot builds a real-time map and analyses the terrain traversability of its working environment. Using the fuzzy approach, the robot can handle the uncertainties based on the data provided by the mounted laser range finder. It extracts the roughness and slope from an elevation map created from point cloud measurement data. Those values are inserted into the fuzzy system and the traversability is calculated based on the fuzzy rule and the defuzzifier. This traversability value is transformed into a risk value which is expressed in the form of a vector field histogram (VFH) to avoid the obstacles. The comparison of this method and the conventional method showed the reliability of the proposed method in spite of the limited sensor data acquisition area.

The robot navigation method using fuzzy logic technique and stereo vision based path planning in a complex unknown environment is introduced in [70][71][72]. With a purpose of increasing the autonomy in cluttered environments, an ordered hierarchical architecture based on fuzzy reasoning is suggested [70]. The robustness, accuracy, adaptability and efficiency of the proposed system are tested in various scenarios in which the obstacles are randomly distributed. The fuzzy logic rules are expected to robustly lead the robot towards the goal position without colliding with dynamic obstacles.

The navigation of an autonomous mobile robot in an unknown environment with obstacles is considered by G. Mester [73]. In this study, not only the distances between the robot, the obstacles and the target are considered but also the orientations between them are taken into account. A robot system integrating techniques, such as dead reckoning, self-localization and environment recognition has been proposed by T. Lee and C. Wu [74]. In their approach, membership functions and fuzzy rules are designed based on the genetic algorithm. Nevertheless, the method only solves the direction problem, without considering the velocity. In addition, the genetic algorithm may not be the best choice for generating the rule base, so the robot might fail to the find suitable path in some scenarios.

The optimization of the traveled distance using an appropriate FLC is proposed in [103] for an omnidirectional mobile robot. The implemented fuzzy controller has shown its adaptation to the dynamics of the real robots. In [104], a fuzzy path planner utilizes a multi-objective evolutionary algorithm. The performance of the evolutionary dual rule-based fuzzy path planner is evaluated through comparing the created paths, the estimated Pareto-optimal fronts from dual multi-objective particle swarm optimization and the dual multi-objective quantum-inspired evolutionary algorithm. In [105], the robot navigation is handled by two controllers. The first controller computes a direct path from the start position to the goal position, without considering the obstacles in the path. For obstacle avoidance in robot navigation, the fuzzy logic controller is taken. This fuzzy logic controller takes the inputs from the laser sensor of the robot and gives the change in the angular velocity as output to the robot in order to avoid the obstacle. The uses of fuzzy logic in robot path planning of the reviewed studies are summarized in Table 2.3.

As afore-mentioned, fuzzy logic has the power of simulating the human thinking represented by linguistic variables and knowledge base represented by IF-THEN rules. However, it has difficulties in selecting the most suitable rules and membership functions. On the other hand, neural networks are able to learn by updating their synaptic weights. However, this representation of knowledge as synaptic weights cannot be acquired by the human reasoning. These matters together suggest a method of integrating fuzzy inference system and neural network to achieve both advantages of similarity to human thinking (fuzzy logic) and the ability to learn (neural networks) in subsection 2.2.3. Furthermore, fuzzy logic can be applied to solve the problem of the parameter adaptation with some natural-inspired algorithms for optimal robot path planning which will be discussed in section 2.2.4.

| **Applications of FL for Robot Path Planning** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Approach | Authors | Kinematic model | Obstacle shape | Static Obstacle | Dynamic Obstacle | Static Target(s) | Dynamics Target(s) | Adapted velocities | Simulation | Real system | Year |
| FL | H. Chang [5] | Y | Corridor | Y | N | Y | N | N | Y | Y | 2013 |
| | V.M. Peri [66] | N | Arbitrary | Y | Y | Y | N | N | Y | Y | 2005 |
| | J.H. Lilly [67] | N | Circle | Y | N | Y | N | N | Y | N | 2007 |
| | A. Zhu [68] | N | Arbitrary | Y | Y | Y | N | N | Y | N | 2004 |
| | Y. Tanaka [69] | Y | Arbitrary | Y | N | Y | N | N | Y | Y | 2015 |
| | A. Foudil [70] | N | Arbitrary | Y | Y | Y | N | Y | Y | Y | 2014 |
| | L. Li [71] | N | Arbitrary | Y | N | Y | N | N | N | Y | 2009 |
| | C.H.Chao [72] | N | Arbitrary | Y | N | Y | N | N | N | Y | 2009 |
| | G. Mester [73] | N | Arbitrary | Y | N | Y | N | Y | Y | N | 2008 |
| FL−GA | T. Lee [74] | N | Arbitrary | Y | N | Y | N | Y | Y | Y | 2003 |
| FL−PSO | M. S. Masmoudi citeM.S.Masmoudi | Y | Arbitrary | Y | N | Y | N | Y | Y | Y | 2016 |
| FL−PSO | C. Hong citeC.Hong | Y | Circle | Y | N | Y | N | N | Y | N | 2016 |
| FL | N. Kumar [105] | Y | Arbitrary | Y | N | Y | N | Y | Y | N | 2017 |

**Table 2.3:** Applications of FL for the robot path planning.

### 2.2.3 Neuro-Fuzzy Technique

The robot path planning performances in [4] are improved in [6][106] where the NN-FC technique is used instead of the sole neural technique. With the same inputs as treated in [4], the technique performs a neural network as a preprocessor of a fuzzy logic controller. The compared performances between the two techniques demonstrate the prominence of neuro-fuzzy system. The algorithm is illustrated in Figure 2.7.

The integrated methods also have been proposed in [7, 75–77]. A method, named Prune-able fuzzy adaptive resonance theory neural architecture (PA-FARTNA) is adopted in [75]. The mobile robot is able to automatically navigate in dynamic environments with an optimal global path planning using its on-line learning ability. In [7], it is suggested an adaptive five-layer neuro-fuzzy network which consists of a fuzzy controller with 48 fuzzy rules and a learning adaptation model. Two learning algorithms are designed to adjust the parameters of the membership functions and automatically overcome a problem of redundant fuzzy rules. As a consequence, a reasonable trajectory

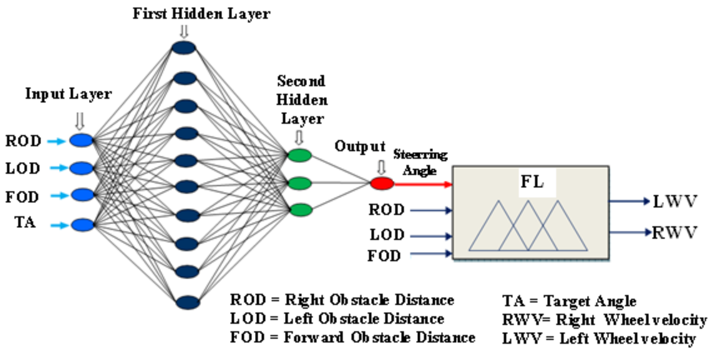

First Hidden Layer
Input Layer
Second Hidden Layer
Output
Steerring Angle
ROD
LOD
FOD
TA
FL
LWV
RWV
ROD = Right Obstacle Distance
LOD = Left Obstacle Distance
FOD = Forward Obstacle Distance
TA = Target Angle
RWV= Right Wheel velocity
LWV = Left Wheel velocity

**Figure 2.7:** Neuro-Fuzzy structure for the robot path planning [6].

is generated for a mobile robot in a dynamic environment with both moving target and moving obstacles. As it can be seen in Figure 2.8, the inputs are the distances between the robot and the obstacles in the left, the front and the right, $d_l$, $d_f$, $d_r$; the target direction $\theta_d$; the current speed of the robot $r_s$. Next, the input membership variables are denoted in the second layer. Then, the rule base is referred in the third layer. After that, the output membership variables are designed in the fourth layer. In the end, the accelerations of the left and right wheels, $a_l$ and $a_r$ are chosen as outputs. The inputs $d_l$, $d_f$, $d_r$ are expressed by only two linguistic variables: Near and Far. Similarly, $\theta_d$ is expressed by three linguistic variables: Left, Center and Right; the current speed $r_s$ is expressed by two linguistic variables: Fast and Slow. The good results prove the efficiency of the algorithm. However, the proposed approach requests much effort for tuning the parameters.

The performances of different integrated approaches (NN-tuned FL, GA-NN-FL, GA-tuned adaptive network-based fuzzy inference system <Anfis>) are compared to the other three methods (default behavior, manually-constructed FL, potential field method) through computer simulations of a car-like robot in a dynamic environment [77]. The approach based on default behavior is quite simple; instead of using a motion planner, the robot maintains the default rules with maximum acceleration and zero deviation at each time step. The collision between the robot and the most critical obstacle is avoided based on the predicted position technique. In the manually-constructed FL approach, Mamdani type fuzzy logic controller is applied while in NN-tuned FL, GA-NN-

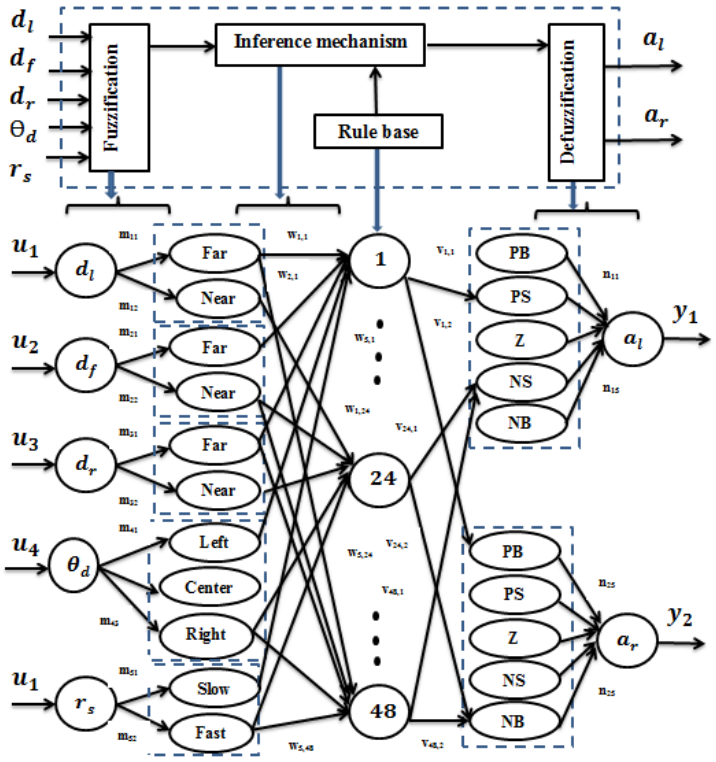


**Figure 2.8:** Neuron- Fuzzy structure for the robot path planning [7].

FL approaches, a back-propagation algorithm and GA are used to optimize FL, respectively. In GA-tuned Anfis approach, GA is also applied to find the optimal solution for Anfis. The performance of the approach based on default behavior gives the worst performance. Moreover, the integrated approaches show better performances than the others in most of the studied scenarios. The potential field method is the fastest of all methods however less adaptive than the integrated approaches. It also indicates that the performances of the neuro-fuzzy approaches depend on the training data. The proposed approaches face some difficulties when the testing scenarios are much different from the training scenarios.

In [107], a function of the obstacle avoidance is realized by searching the

| **Application of Neuron-Fuzzy technique for Robot Path Planning** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Approach | Authors | Kinematic model | Obstacle shape | Static Obstacle | Dynamic Obstacle | Static Target(s) | Dynamics Target(s) | Adapted velocities | Simulation | Real system | Year |
| NN−FL | D.R. Parhi [6] | N | Arbitrary | Y | N | Y | N | Y | Y | Y | 2008 |
| | S.K. Pradhan [106] | N | Arbitrary | Y | N | Y | N | Y | Y | Y | 2006 |
| PAFARTNA | R. Araujo [75] | Y | Arbitrary | Y | Y | Y | N | N | Y | Y | 2006 |
| NN−FL | A. Zhu [7] | N | Arbitrary | Y | N | Y | N | Y | Y | Y | 2009 |
| | M.M. Joshi [76] | N | Arbitrary | Y | N | Y | N | Y | Y | N | 2011 |
| NN−FL, GA−Anfis, PFM | N. B.Hui [77] | Y | Circle | N | Y | Y | N | Y | Y | N | 2006 |
| NN−FL, PSO | Y. Gou[107] | Y | Rectangle | Y | N | Y | N | N | Y | N | 2017 |
| NN−FL, A* | Z. Shi[108] | Y | Arbitrary | Y | N | Y | N | N | Y | N | 2016 |

**Table 2.4:** Applications of Neuron-Fuzzy technique for the robot path planning.

next feasible node by a fuzzy-neural network. Aiming at the parameter optimization problem, an improved particle swarm optimization algorithm is used to optimize the parameters of the fuzzy-neural network, which avoids the instability of the system caused by improper parameter selection.

In [108], Neuron-Fuzzy is improved by using the A* graph search algorithm to guarantee an optimal path, providing the rationality and the feasibility. The grid map is divided into two stages, including the A* algorithm in the first stage and the Neuron-Fuzzy in the second stage. In addition, a neural network based on adaptive control strategy is introduced to ensure the stability and to compensate the sensor failure, fact caused by the loss of data and information uncertainty.

Normally, the integrated approaches based on neural and fuzzy provide much better results than individual techniques. The comparisons in [6][106][76][78] show this advantage; however, the integrated methods usually have a time consuming problem. The overview of application on Neuro-Fuzzy technique of robot path planning in the subsection is presented in Table 2.4.

### 2.2.4 Nature-Inspired Algorithms

Recently, robot navigation techniques inspired by biological behaviors, known as bio-mimetic algorithms, have obtained much more attention. The following sub-sections introduce the applications of robot path planning using three members of them, GA, PSO and ACO. The choice of these three nature-inspired methods is made because there are many studies which demonstrated the good performances for robot path planning.

#### I. Genetic Algorithms (GA)

GA is an optimization tool based on the natural genetics that takes the advantages from procedures such as natural selection, crossover, and mutation. GA has great potentiality to solve the combination optimization problems. The pseudo code of GA path planning is provided [78].

Various studies have been executed based on GA in robot path planning domain. In [79], that issue is solved by applying a knowledge based genetic algorithm (problem-specific genetic algorithm) instead of the standard GA. The algorithm is designed with both domain knowledge and small-scale local search. The proposed method is suitable for both static and dynamic environments. This algorithm is extended in [80] for multiple mobile robots in dynamic environments. When the robot's working space is simple or does not change very fast, this approach provides a near optimal solution from the current location to the target. In a very complicated fast-dynamic environment, a feasible path is always produced. To generate a reasonable collision free path from the starting point to the goal point when the mobile robot is trapped in an acute "U" or "V"-shaped obstacle or the mobile robot encounters dynamic obstacles, a GA-based dynamic path planning algorithm (DPPA) is proposed in [81]. In addition, this algorithm is goal-oriented and thus reduces unnecessary search time. The performance in both simulation and real-time implementation are investigated and evaluated by checking the ability to generate a feasible path in order to avoid acute obstacles/dynamic obstacles and to find the shortest path.

In [8], B.K Oleiwi et al. presented a hybrid approach based on GA, a modified search A* algorithm and fuzzy logic. A* algorithm tends to minimize the route cost on the graph, as in the road map approach or on a regular grid. It evaluates the goodness of each node or cell by combining two metrics which are the distance from the robot to the start position and estimated distance from the robot to the goal position. As a result, a multi-objective optimization of free-collision path is generated. The approach obtains a smooth trajectory with an associated minimum energy cost by using cubic spline. First, the modified GA and a modified search A* algorithm are applied to find the optimal path. After that, the global optimal trajectory is treated as input for the fuzzy motion controller to regenerate a time-based trajectory. When unknown obstacles appear in the robot's path, the fuzzy controller decreases the speed of the robot. The performances are very impressive in an environment with dynamic obstacles. A description of the proposed approach is depicted in Figure 2.9.

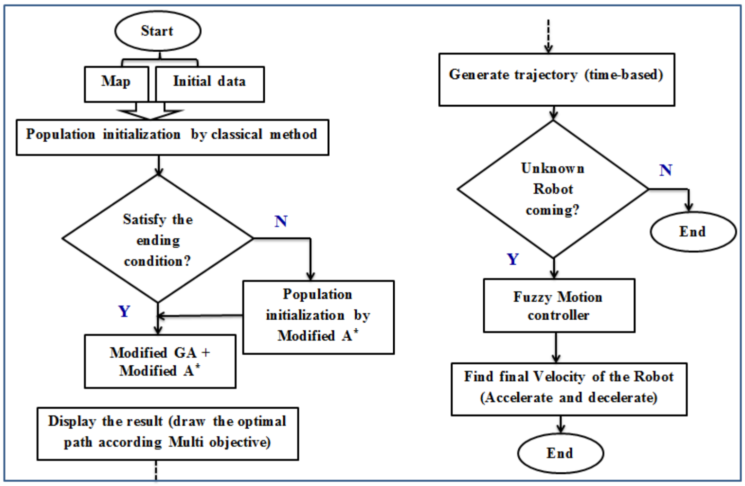

Start
Map
Initial data
Population initialization by classical method
Satisfy the ending condition?
N
Y
Population initialization by Modified A*
Modified GA + Modified A*
Display the result (draw the optimal path according Multi objective)
Generate trajectory (time-based)
Unknown Robot coming?
N
End
Y
Fuzzy Motion controller
Find final Velocity of the Robot (Accelerate and decelerate)
End

**Figure 2.9:** Robot path planning by GA-Fuzzy-A* approach [8].

In order to track the moving target, a method based on fuzzy logic and genetic algorithm is presented in [82]. An optimal path of an autonomous robot is delivered by applying the hybrid meta-heuristic GA-PSO algorithm [83]. The combination of GA and PSO proposes a new solution by executing crossover and mutation operators. As a result, the premature convergence in conventional GA and PSO algorithm is avoided.

The novel genetic-fuzzy algorithm has been applied to generate a robot path in an unknown environment [84]. In this approach, the genetic algorithm serves to find the optimal path in an alien dynamic environment. A full consideration of three elements: the collision avoidance path, the shortest distance and smoothness of the path are taken into account in the fitness function. Moreover, a dual-population concept on diversifying the population is also introduced to improve the performances. In order to overcome the local trap problem and to avoid premature convergence, A.H. Karami et al. designed an adaptive selection operator instead of the conventional GA based on feedback information of the search process [85]. In the beginning, an initial population is generated by a random-based method. Then, the created paths are improved in each generation of the genetic algorithm. Therefore, the robot path planning quality is significantly improved.

In a similar approach, a GA is implemented in [109] to generate a collision-free optimal path, linking the initial configuration of the mobile robot (Source) to a final configuration (Target). After that, Piecewise Cubic Hermite Interpolating Polynomial (PCHMIP) is used to smooth the generated optimal path. Finally, an adaptive fuzzy controller is designed to keep track of a mobile robot on the desired smoothed path (by transmitting the appropriate right and left velocities using wireless communication). In parallel, sensor fusion (odometry sensors and Kinect sensors) is used to estimate the current position and orientation of the robot using the Kalman filter. GA is applied for the task allocation and collision-free path planning of three robots working in the common work-space in [110].

Five path planners for mobile robot path planning namely: A*, a relaxed version of A*, the GA, the Tabu search (TS) algorithm and the NN algorithm are compared in [111]. The study shows that heuristic algorithms are in general not appropriate for solving the path planning problem in grid environments. They are not as effective as A* since the latter always exhibits the shortest execution times and the best solution qualities. GA was found to be less effective for large problems, i.e., it is able to find optimal solutions like A* in some cases, but it always exhibits a greater execution time. TS was also found to be the least effective as the exploration space is huge in large problems, and it only explores the neighborhood of the initial solution. On the contrary, NN provides better solutions than the two aforementioned techniques. This can be explained by two reasons. The first reason is that heuristic approaches need to generate one or several initial paths with the greedy approach (in the case of GA and TS), and this operation itself takes time which is comparable to the execution of the whole A* algorithm. The second reason is that the heuristic approaches need several iterations to converge, and this number of iterations depends on the complexity of the map and the positions of the start and goal cells.

The applications of the above-mentioned GA algorithm are shown in Table 2.5.

| **Application of GA technique for Robot Path Planning** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Approach | Authors | Kinematic model | Obstacle shape | Static Obstacle | Dynamic Obstacle | Static Target(s) | Dynamics Target(s) | Adapted velocities | Simulation | Real system | Year |
| GA | Y. Hu [79] | N | U shape, Rectangle | Y | Y | Y | N | N | Y | N | 2004 |
| | S.X. Yang [80] | N | Arbitrary | Y | Y | Y | N | N | Y | N | 2006 |
| GA based DPPA | S.C.Yun [81] | N | Paralle lepiped | Y | Y | Y | N | N | Y | Y | 2011 |
| GA—Fuzzy, A* | B.K. Oleiwi [8] | Y | Circle | Y | Y | Y | N | N | Y | N | 2014 |
| GA—Fuzzy | B. Karim [82] | N | Point | N | N | N | Y | Y | Y | N | 2013 |
| GA- PSO | H.C. Huang [83] | N | U shape, Rectangle | Y | N | Y | N | N | Y | N | 2011 |
| GA-FL | S.M.R. Farshchi [84] | Y | Cylinder | Y | Y | Y | N | N | Y | N | 2011 |
| Adaptive GA | A.K. Karami [85] | N | Trap shape | Y | N | Y | N | N | Y | N | 2015 |
| GA, Adaptive FL | A. Bakdi [109] | N | Trap shape | Y | N | Y | N | N | Y | N | 2017 |
| GA/ A* | K. Jose [110] | Y | Arbitrary | Y | N | Y | N | Y | Y | Y | 2016 |

**Table 2.5:** Applications of GA technique for the robot path planning.

In summary, the main imperfection of the GA approach in robot path planning field is that it is not feasible for dynamic environments. Since GA operates in a grid map and does not control the population diversity, a premature convergence might happen. Therefore, GA is integrated with other algorithms such as fuzzy or PSO to achieve better results in robot path planning applications.

### II. Particle Swarm Optimization (PSO)

Similar with GA, the PSO performs a random population and uses objective function to evaluate particles. However, the crossover and mutation operations as in genetic algorithm disappear in PSO. The particles update themselves with the internal velocity based on their self-experiences and their social experiences. As the PSO takes real numbers as particles, it is very convenient compared to the GA, which needs to be changed to binary encoding.

Initially, the PSO is inspired by the social behavior of bird flocking or fish schooling. This algorithm is a population based stochastic optimization technique. Basically, the PSO is initialized with a set of random solutions and then it updates each generation based on the optimal schema. Then, the global optimum is achieved by changing the collection of particles in a search space towards a promising area. In each iteration, the particles will update their locations until reaching the target location (global optimum) [9]. Their movement is affected by a fitness function that evaluates the quality of each solution. The Negative PSO algorithm (NPSO) is presented in [86] and then,

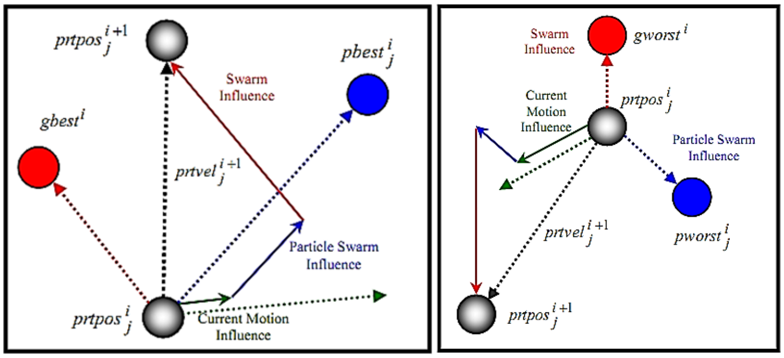


**Figure 2.10:** Depiction of a particle position update in PSO (left) and NPSO (right) [9].

it is employed for robot motion planning in [9]. In the opposite direction of the PSO, each particle in the NPSO updates its position according to its own previous worst solution and its group's previous worst solution to find the optimal value as in Figure 2.10-right hand side. The procedure is applied to avoid the worst positions based on similar formula of the regular PSO. In this research, the performances of the PSO and the NPSO are compared in the robot path planning.

In [87], Y. Zhang et al. presented a multi-objective robot path planning algorithm based on PSO in an uncertain environment. The objective function consists of the risk degree and the path length. Hence, the path planning problem is considered as a constrained bi-objective optimization problem with uncertain coefficients. A polar coordination PSO (PPSO) for the robot path planning in dynamic environments is presented in [88]. The algorithm decomposes the task into a global phase and a local planning phase. It can search for the global optimal path depending on the static obstacles information. Then, an on-line real-time path planning strategy is adopted to avoid moving obstacles by predicting their future positions. In [89], a path planning algorithm for a group of robots is formulated as an optimal problem with constraints on obstacle avoidance and V-shape formation in the dynamic environment. The fitness function is defined by minimizing the trajectory of the group while keeping the V-shape formation.

Many variations of PSO have been proposed in the literature. K.D. Sharma et al. proposed a novel idea for vision-based robot navigation using stable adaptive fuzzy tracking controllers and PSO-based hybrid methodologies [90]. GA and PSO approaches have been carried out for robot navigation in unknown environments [91]. For the same purpose, Darwinian particle swarm optimization algorithm is applied to escape from local optimal solutions [92]. This one is expanded from the original PSO by adding a natural selection operation. The experimental results demonstrate an effective convergence as compared to the traditional algorithms in real system executions and simulated trials. Also, the distribution of target locations, does not greatly affect the adaptive algorithm's performance.

A new methodology is proposed in [112] to determine the optimal trajectory for multi-robot in a clutter environment using the hybridization of improved particle swarm optimization (IPSO) with an improved gravitational search algorithm(IGSA). The proposed approach embedded the social essence of IPSO with motion mechanism of IGSA. The hybridization IPSO-IGSA maintains the efficient balance between exploration and exploitation because of adopting co-evolutionary techniques to update the IGSA acceleration and particle positions with IPSO velocity simultaneously. The objective of the algorithm is to minimize the path length of all the robots to their respective destination in the environment.

The problem of multi-robot target searching in unknown environments is presented in [113] in which the extension of PSO, named A-RPSO (Adaptive Robotic PSO), is used as a controlling mechanism for the robots. This method is similar with PSO, however with two modifications: firstly, it takes into account the obstacle avoidance; secondly, A-RPSO has a mechanism to escape from local optima. A new on-line path planning method is proposed in [114] which generates collision-free and COLREGs-compliant paths using a multi-objective optimization. A key feature of the proposed technique is the incorporation of both compliant and non-compliant target vessels without making any changes to the basic path planner.

The summary of PSO algorithm for robot path planning in this part is presented in the Table 2.6. Most of the proposed PSO approaches were not reliable in real-time robot applications, especially in dynamic environments. It might be said that these researches mainly solve problems by updating the velocity parameters, aiming to attain more effective and faster convergence.

| **Application of PSO technique for Robot Path Planning** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Approach | Authors | Kinematic model | Obstacle shape | Static Obstacle | Dynamic Obstacle | Static Target(s) | Dynamics Target(s) | Adapted velocities | Simulation | Real system | Year |
| PSO/ NPSO | E. Masehian [9] | N | Arbitrary | Y | N | Y | N | N | Y | Y | 2010 |
| PSO | Y. Zhang [87] | N | Arbitrary | Y | N | Y | N | N | Y | N | 2013 |
| | Y. Hao [88] | N | Arbitrary | Y | Y | Y | N | N | Y | N | 2007 |
| | Y. Wang [89] | Y | Circle | Y | Y | Y | N | N | Y | N | 2009 |
| PSO-Fuzzy Lyapunov | K.D. Sharma [90] | N | Rectangle | Y | N | Y | N | N | Y | N | 2006 |
| Stochastic PSO | C. Xin [91] | N | Various | Y | N | Y | N | N | Y | N | 2006 |
| Fuzzy, Darwinian PSO | M.S. Couceiro [92] | N | Arbitrary | Y | Y | Y | N | Y | Y | Y | 2012 |
| IPSO/IGSA | P.K.Das [112] | N | Arbitrary | Y | Y | Y | N | Y | Y | Y | 2016 |
| A-RPSO | M. Dadgar [113] | N | Arbitrary | Y | N | Y | N | N | Y | N | 2016 |
| PSO | L. Hu [114] | N | Arbitrary | Y | N | Y | N | N | Y | N | 2017 |

**Table 2.6:** Applications of PSO technique for the robot path planning.

## III. Ant Colony Optimization (ACO)

Like PSO algorithm, ACO is a data clustering algorithm which implements swarm behavior. ACO is originated from the natural ant colony behavior. At the essence of the behavior, the interactive communication between the ants enables them to find the shortest path between their nest and the food sources. The optimal route is obtained through evaluating the amount of pheromones deposited by ants on the paths. This characteristic of the real ant colonies is inherited by the ACO algorithm to solve the discrete optimization. Consequently, the ACO is more suitable for problems where the source and the destination are clearly predefined and specific.

An algorithm for solving multi-goal planning problems in the presence of obstacles is presented in [93]. In this study, the ant colony optimization is combined with a sampling-based point to point path planning algorithm. To evaluate the performance, a quantitative comparison with two existing sampling-based algorithms has been made. The ACO algorithm, offering a compromise between solution quality and speed, is the superior choice given the physical parameters for the robot planning. In [94], a method for solving the SLAM problem by applying an optimization technique based on the Ant Colony Optimization meta-heuristic is introduced. A tree-like data structure is generated, in which the path from the root to a leaf represents a metrical map estimation. An (almost) optimal path is found by employing an ACO.

In [95], X. Chen et al. proposed a two-stage ACO model which has the ability to overcome the main inconsistency problem between the premature convergence and the optimal path. In [10, 96], an optimal path planning method is introduced based on Ant Colony Optimization Meta-Heuristic (ACO-MH). The robot is considered as a point so that it occupies an exact cell in the discrete representation of working environment. The length of the path and the difficulty for the navigation are taken as the cost functions that are evaluated by a fuzzy logic system. The algorithm has the adaptive capability of changing in the environment so that it can perform a global robot path planning with dynamic obstacles. The mobile robot is navigated on-line as indicated by the flow chart in Figure 2.11.

In [97], the searching range and the speed of the algorithm are significantly improved by choosing the initial pheromone distribution. The combination of potential field method (PFM) and ACO is proposed for the problem of robot planning in [98]. In the hybrid algorithm, a deterministic planning is established by applying PFM, then ACO is used for searching an optimal route. The results indicate that the probability of finding the most advantageous solution is substantially increased.

A heading direction methodology is proposed in [115] in conjunction with an ACO during the motion planning in the vicinity of obstacles to plan safer trajectories for real-time navigation and map building of a mobile robot. A LIDAR-based local navigation algorithm is implemented to carry out obstacle avoidance missions. As the robot plans its trajectory towards the target, unreasonable path will be inevitably planned. A heading-enabled navigation paradigm is developed for locally guiding the robot to plan more reasonable and safer trajectories. A novel UAV's optimal path planning approach based on the combination of A* search algorithm and ACO algorithm, is presented in [116]. The proposed path planning solution aims to identify the optimal patrolling path in a complex 3D topography given a set of patrolling positions.

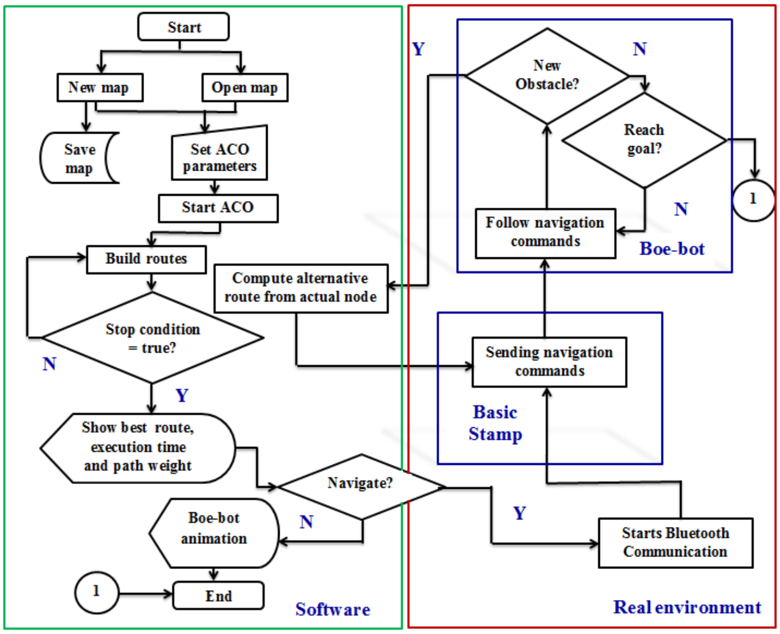


**Figure 2.11:** Robot navigation based on Fuzzy-ACO algorithm[10].

In [117], an ACO algorithm has been applied to solve the path planning problem of a mobile robot in complex environments. The algorithm parameters have been analyzed and tuned for different working areas with obstacles in different number, sizes and shapes. Also, the performances of the proposed algorithm were tested for different resolutions of working area representations.

In all cases, it was possible to find optimal or near-optimal minimum-length paths from the initial to the final positions, without colliding with obstacles or wall-borders. The summary of this subsection is presented in Table 2.7.

In conclusion, the traditional ACO involves some shortcomings, for example, requiring a long time to reach the optimal value if the size of the problem is large. The solutions are blindly created in the beginning of the evolution phase, since the pheromone concentrations of all members are equally initialized. As a consequence, it takes a long time to find a better path from a great number of smoother paths.

| **Application of ACO technique for Robot Path Planning** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Approach | Authors | Kinematic model | Obstacle shape | Static Obstacle | Dynamic Obstacle | Static Target(s) | Dynamics Target(s) | Adapted velocities | Simulation | Real system | Year |
| ACO | B. Englot [93] | N | Rectangle | Y | N | Y | N | N | Y | N | 2011 |
| | R. Iser [94] | N | Various | Y | N | Y | N | N | Y | N | 2010 |
| | X. Chen [95] | N | Various | Y | N | Y | N | N | Y | N | 2013 |
| ACO- Fuzzy | M.A.P. Garcia [96][10] | N | Rectangle | Y | N | Y | N | N | Y | Y | 2007 2009 |
| Improved ACO | J. Bai [97] | N | Circle | Y | N | Y | N | N | Y | N | 2011 |
| PFM- ACO | D. Zhao [98] | N | Rectangle | Y | N | Y | N | Y | Y | N | 2006 |
| ACO | C. Luo[115] | N | Rectangle | Y | N | Y | N | N | Y | N | 2016 |
| ACO/A* | H. Wang[116] | N | Various | Y | N | Y | N | N | Y | N | 2016 |
| ACO | R. Uriol[117] | N | Various | Y | N | Y | N | N | Y | N | 2017 |

**Table 2.7:** Applications of ACO technique for the robot path planning.

## 2.3 Classical Algorithms

As mentioned in section 2.2, the classical methods no longer provide suitable and effective results for mobile robot navigation processes in unknown and dynamic environments compared to the heuristic techniques. Hence, in this section, only the Potential Field Method (PFM) is presented due to its advantages. PFM is broadly applied for real time collision-free path planning in mobile robot domain. In this method, the robot is considered as a particle steering under the influence of a potential field, which is determined by the obstacles and the goal. The goal is supposed to have the attractive potential force, while the obstacles generate the repulsive potential forces. Therefore, the potential field will pull the robot towards the goal and push it far away from obstacles.

The PFM is attractive due to its mathematical simplicity and efficiency. However, the basic PFM algorithm is simply based on the distance between the robot and the obstacles/target in a static environment. A navigation system for a family of indoor monitor mobile robots is presented in [118]. The robot detects its working space using a 2D laser range finder. The hybrid localization method integrates an odometry localization method, a straight line matching method, and a corner matching method. Then, a weight average method is implemented to obtain the optimal pose according to the error model. PFM is used to plan a collision free path that satisfies certain optimization criteria. Recently, several proposed potential field methods applied in a dynamic environment with both moving targets and moving obstacles are studied. In order to avoid the moving obstacles, the relative positions and velocities of the robot and the obstacles are taken into account in the repulsive potential function [119–123]. Similarly, the attractive potential function is modified so that this value increases as the relative distance or the relative velocity between the robot and target becomes larger [119].

In [120, 121], a modified PFM is extended to track a moving target. Furthermore, in [122], the implementation elements, such as a maximum linear speed and an angular speed of the robot, are also contemplated. The approaches ensure that the robot follows the moving target while avoiding the moving obstacles. An integrated method, named path-guided PFM with Stochastic Reachable Sets including sampling-based technique and PFM is proposed in [124]. The method has a low computational cost and it flexibly works in very crowded dynamic environments. The sampling-based technique is used to identify a collision-free path with respect to the static obstacles. Then, MPF is applied to safely navigate through the moving obstacles, using the path as an attractive intermediate goal bias. In addition, the authors incorporate a repulsive potential field for each moving obstacle based on pre-computed stochastic reachable (SR) sets. The proposed method is tested in environments with 300 stochastic moving obstacles, that either are free of static obstacles or have static obstacles in the shape of a "bug trap" or narrow corridors. The obtained results show that the method avoids local minima and has a very high success rate (over 90%) in highly dynamic environments.

Comparative performances of various robot path planning schemes have been employed in [123]. Hybrid algorithms including genetic-fuzzy, genetic-neural systems and a conventional PFM have been examined. An attempt is made to identify the best method in terms of the traveling time, robustness, adaptability and reliability. The time-consumptions of those hybrid algorithms are quite small. Hence, they are qualified for on-line implementations. However, PFM is found to be less adaptive than genetic-fuzzy or genetic-neural approaches.

A new method to avoid a local minima in PFM is proposed in [100]. It is shown that the PFM has an attracting equilibrium at the target point, repelling equilibrium in the obstacle centers and saddle points on the borders. The unstable equilibrium is avoided by using the established Input-to-State Stability (ISS) property. An extended potential field controller (ePFC) is presented in [125], which enables an aerial robot to safely track a dynamic target location while simultaneously avoiding any obstacles in its path. In [126], Stochastic Reachable (SR) sets are implemented to generate accurate potential fields for the moving obstacles. A small number of SR sets are computed as a priority, then used to generate a potential field that represents the stochastic motion of obstacles for online path planning. A model predictive path-planning controller is introduced in [127] such that its objective includes potential functions along with the vehicle dynamics terms. The uses of PFM for robot path planning presented in this subsection are summarized in Table 2.8.

To conclude, although PFM has a simple structure and it is easily implementable. However, it might fail in some scenarios where obstacles are located close to each other. It is necessary to improve the PFM or to integrate it with other intelligent techniques.

## 2.4 Summary

The robot path planning problem is divided into classical methods and heuristics methods. The classical methods are easy to implement. Therefore, they are preferred in many real-time motion planning applications and may obtain good results, as shown by RRT, PFM, PRM. However, the classical approaches often require precise information about the robot's working environment, thus more accurate sensors should be used in real-time applications. Recently, a family of optimal SBP, such as RRT*, RRM*, RRG* is adopted to guarantee

| **Applications of PFM for Robot Path Planning** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Approach | Authors | Kinematic model | Obstacle shape | Static Obstacle | Dynamic Obstacle | Static Target(s) | Dynamics Target(s) | Adapted velocities | Simulation | Real system | Year |
| PFM | F. Tan [118] | N | Various | Y | N | Y | N | N | N | Y | 2010 |
| New PFM | S.S. Ge [119] | N | Point | N | Y | N | Y | Y | Y | N | 2002 |
| | L.Huang [120] | N | Circle | Y | N | N | Y | Y | Y | N | 2009 |
| | L.Huang [121] | N | Circle | Y | N | N | Y | Y | Y | N | 2012 |
| | L. Valbuena [122] | Y | Cube | Y | N | Y | N | N | Y | Y | 2012 |
| Path-Guided APF-SR | H.T. Chiang [124] | N | Square | Y | Y | Y | N | N | Y | N | 2015 |
| GA-Fuzzy, GA-NN, PFM | N. B. Hui [123] | Y | Circle | N | Y | N | Y | Y | Y | Y | 2009 |
| PFM/ ISS | M. Guerra[100] | Y | Various | Y | N | Y | N | N | Y | N | 2016 |
| ePFC | A. C. Woods[125] | Y | Various | Y | N | Y | N | N | Y | Y | 2017 |
| PFM | N. Malone [126] | N | Various | Y | Y | Y | N | N | Y | N | 2017 |
| Predict PFM | Y. Rasekhipour [127] | Y | Various | Y | N | Y | N | Y | Y | N | 2017 |

**Table 2.8:** Applications of PFM technique for the robot path planning.

the asymptotic optimality. Although they work very effectively, they provide no theoretical guarantees for reaching an optimal solution [38]. Compared to the classical methods, the heuristic methods are considered more "intelligent" and more "advanced" as they can adapt to both uncertain and incomplete information in constantly changing environments (i.e., neural network, fuzzy logic methods) and they achieve the optimal solutions (the nature-inspired methods). Thus, they are applied on autonomous navigation with complex tasks. However, their serious drawbacks are the necessary learning phase and the high computational time. In many applications, the computational capacity of the robot controller unit is limited, thus the approaches with lower computational cost such as RRT, PFM are better choices.

In the last decades, a significant progress towards the development of robots' path planning has been made. This chapter has reviewed the state-of-the-art heuristic approaches and potential field method in the robot path planning domain. The Heuristic methods analyzed in this chapter consist of neural network, fuzzy logic and nature-inspired methods (genetic algorithm, particle swarm optimization and ant colony optimization). The applications of the reviewed studies are summarized in Table B.1 in Appendix.

As mentioned before, each method has some advantages and disadvantages. The Neural network has been successfully applied in many robot path planning applications due to the nonlinear mapping, learning ability and parallel processing characteristics. However, it is usually time consuming since it often has a large number of parameters that have to be adjusted. It also requires a big variety of training databases with a possible large learning cost. Another drawback of NN is that it is impossible to interpret the functionality as it is a black-box. It is also difficult to determine the number of layers and the number of neurons; this is out of the scope of this thesis.

Fuzzy logic has a power of simulating the human thinking represented by linguistic variables and knowledge-based IF-THEN rules, but it has difficulties in selecting the most suitable rules and membership functions. One approach to design a fuzzy logic system is to try to imitate the actions of an experienced operator. It is claimed that this provides good results. The fuzzy law is expressed in terms of linguistic variables, which is simple and transparent. Fuzzy logic, as a mathematical tool to handle uncertainties, has been applied to systems that are difficult to be precisely defined. The integrated approaches based on neural and fuzzy techniques provide much better results than the individual ones, as they both achieve the advantages of similarity to the human thinking (fuzzy logic) and the ability to learn (neural networks). As a guideline for implementing high efficient neuro-fuzzy robot path planning systems, they should have characteristics such as fast learning, on-line adaptability and self-adjustment to changing environments. The data acquisition and pre-processing of input training data are important factors for neuro-fuzzy systems.

Nature-inspired algorithms have great potentialities to solve the optimization problems. Optimal criteria, such as the minimum traveled distance, minimum time and minimum control effort are considered for path planning problems. However, they are not reliable in real-time applications of robot path. Therefore, they are often integrated with other methods, such as classical algorithms, fuzzy and neural approaches, to achieve better results.

Clearly, there is no single optimal path planning algorithm suitable for all autonomous navigation applications. For each application, a suitable solution is selected based on the user's purposes, system characteristics, working spaces and experiences. In the case where the application highly desires to inherit the expertise of an expert operator, the fuzzy control is the best choice. However, in a cluttered environment, integrated methods which combine several algorithms, for example classical algorithms with nature-inspired method, obtain optimal performance and reduce complexity of operations.

To implement a real-time autonomous navigation process for a complex system that has a highly dynamic characteristic and small memory capacity, an effective algorithm with fast updated ability is an essential requirement. Then, PFM is an attractive solution as this method has a simple structure and a low computational complexity.

In this thesis, all the problems mentioned above are considered and investigated. Without loss of generality, Unmanned ground vehicles (UGVs) and Unmanned Aerial Vehicles (UAVs) are used as case studies to implement the proposed algorithms.

In Chapter 3, we firstly considered fuzzy-based autonomous navigation process of an UGV in static environment with an expectation to inherit expertise of an expert user.

In Chapter 4, we proposed a novel hierarchical approach to solve a global optimal path planning problem in cluttered environments. The aim of the approach is to generate optimal collision-free paths focused on minimizing the path length and maximizing the smoothness regarding the robot constraints.

Chapter 5 presented a real-time implementation of an UAV's autonomous navigation in unknown/dynamic environments. This chapter investigates the localization based on a sensor fusion method, a proposed PFM in unknown/-dynamic environments and optimal controllers.

Chapter 6 solved the UAV autonomous landing problem in unpredictable working space.

Finally, Chapter 7 provided the main contributions made by this thesis and outlined several directions for future research.

## Chapter 3

# Fuzzy-Based Autonomous Navigation of a Mobile Robot in Outdoor Environments

The fuzzy control popularity can almost solely be attributed to the fact that it is a means of expressing the subjective uncertainties in the human's mind. It is highly desirable to mimic the human's ability to develop autonomous robot navigation strategies [65].

This chapter considers a fuzzy-based autonomous navigation approach for an UGV in static environments. An investigated example is a mobile platform for archaeological prospection as this mobile platform is extremely expected to use the archaeologist's knowledge and experience. The fuzzy control provides a suitable tool towards accomplishing this goal.

The objectives of this study include the developments of:

1. An intelligent fuzzy mechanism by which the mobile platform travels in an efficient manner as humans do.

2. An adaptive system which effectively works with different terrains and presence of obstacles.

3. Adjusting the mobile platform's speed in a timely effective manner.

It is not easy to construct a fuzzy system which has a priority guarantee of infinite feasibility because of pragmatic decisions in practice. Instead, we might have a situation where we are given a fuzzy controller, and the goal is to divide an autonomous navigation task into implementable rules.

The results of this chapter are published as follows.

1. T. T. Mac, C. Copot, R. De Keyser, T. D. Tran, T. Vu, MIMO Fuzzy Control for Autonomous Mobile Robot, *Journal of Automation and Control Engineering*, vol. 4 (1), pp. 65-70, 2016.
2. T. T. Mac, C. Copot, L. Verdonck, R. De Keyser, Speed control strategy of geophysical measurement platform for archaeological prospection: a conceptual study, *IEEE International Conference on Systems, Man, and Cybernetics*, Budapest, Hungary, pp. 3748-3753, 2016. (P1-ISI)
3. T. T. Mac, C. Copot, R. De Keyser, T. D. Tran, T. Vu, MIMO fuzzy control for autonomous mobile robot, *7th International conference on Computer Research and Development*, Ho Chi Minh, Vietnam, pp. 277-282, 2015.

## 3.1 Introduction

Mobile robots are increasingly developed for outdoor missions that demand an extended degree of autonomy. Agricultural operations such as executing non-capacitate operations in fields inhabiting multiple obstacle areas [128], formation control of field covering planetary explorations [129], search and rescue missions in hazardous areas [130], autonomous measurement [131], represent the potential for autonomous vehicles in a natural environment. Unlike indoor operations, where only a flat terrain is considered, outdoor operations encompass different terrains and unknown obstacles, making the development of autonomous vehicles a challenging problem.

Because excavations are expensive and time-consuming, non-destructive methods for acquiring archaeological information have become important. One such method is the archaeological geophysics. In the last decade, the geophysical data acquisition has radically altered. Nowadays, measuring instruments are mounted on a cart towed by an autonomous vehicle (ll-terrain vehicle-ATV), which allows a rapid data collection. Even when motorized, the data-acquisition remains a time-consuming and a repetitive job for the human operator. For instance, following parallel lines over many hectares demands continuous concentration from the operator. Recently, there have been rapid developments in the automotive industry towards more autonomous and even self-driving vehicles. Meanwhile, the progress is being made towards self-driving cars, and many insights can be applied to the ATVs. Vehicle-towed platforms carrying magnetometers, ground-penetrating radars (GPR), electromagnetic inductors or earth resistance sensors, permit fast coverage of large areas at a high sample density, especially when several channels are used in parallel [132].

Also with regard to the ATVs, commonly used in archaeological geophysics as towing vehicles, the first autonomous prototypes have been developed [133–135]. In [134], a first step towards the design and development of the ATV's speed control system is described. The system seems appropriately for the geophysical measurements. In [135], several experiments have been conducted emphasizing the possibility to mimic the typical human driving behaviors. However, the terrain characteristics have not been taken into account even they are quite important factors for this robot type. In [136], an intuitive nonlinear lateral control strategy for trajectory tracking of a modified Ford E-350 van is proposed. In [137], prospection data (e.g., from aerial photography or geophysical prospection) is unable to answer all the questions asked in archaeological studies, since they cannot provide information with the same precision as in-situ excavations. Hithereto, these prototypes have not been used for archaeological prospection, despite their high potential to the field.

On one hand, it is necessary to develop an autonomous (robotic) platform to conduct the geophysical measurements which will be used to gain the better results in a timely-effective manner. On the other hand, the geophysical measurements have typical data collection requirements therefore it is highly desirable to capture the expertise of the archaeologist and use his knowledge to develop autonomous navigation strategies for the AGMP. A fuzzy logic control strategy provides a suitable means towards accomplishing this goal.

In this chapter, a fuzzy-based autonomous navigation strategy of the geophysical measurement platform is proposed for archaeological prospection. The aim is to develop the algorithm by which the AGMP can travel in an efficient manner as humans do. The AGMP can autonomously decide to accelerate in parts of exploitation area where no archaeological elements are presented, and to decelerate where archaeological elements need to be characterized with a high accuracy. In addition, it also adapts to different terrains (e.g., a high speed where its road is smooth; a low speed where its road is rocky) and obstacles (e.g., speed up in case no obstacle and slow down otherwise).

## 3.2 Fuzzy control

Fuzzy control provides a formal methodology for representing, manipulating, and implementing a human heuristic knowledge about how to control a system [11]. The fuzzy controller block diagram is given in Figure 3.1, where a fuzzy controller is embedded in a closed-loop control system. The plant outputs are denoted by $y(t)$, its inputs are denoted by $u(t)$, and the reference input to the fuzzy controller is denoted by $r(t)$. The fuzzy controller has four

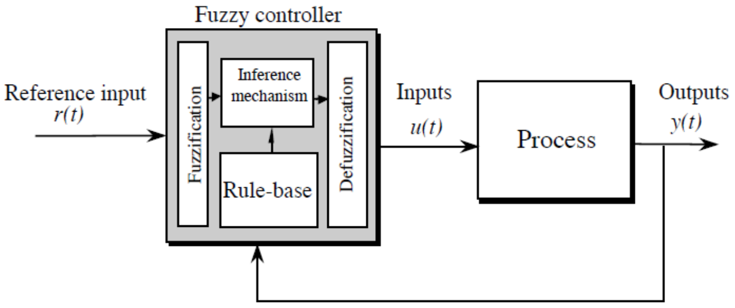


**Figure 3.1:** The closed loop with Fuzzy controller [11].

main components:

1. A rule-base (a set of IF-THEN rules), which contains a fuzzy logic quantification of the expert linguistic description of how to achieve a good control.
2. An inference mechanism, which emulates the expert decision making in interpreting and applying knowledge about how best to control the plant.
3. A fuzzification interface, which converts controller inputs into information that the inference mechanism can easily use to activate and apply rules.
4. A defuzzification interface, which converts the conclusions of the inference mechanism into actual inputs for the process.

### 3.2.1 Linguistic Variables, Values and Rules

A fuzzy system is a static nonlinear mapping between its inputs and outputs. It is assumed that the fuzzy system has inputs $u_i \in U_i$ where $i$ = 1, 2, . . ., $n$ and outputs $y_i \in Y_i$ where i = 1, 2, . . ., $m$, as shown in Figure 3.2. The inputs and outputs are crisp. They are real numbers, not fuzzy sets. The fuzzification block converts the crisp inputs to fuzzy sets, the inference mechanism uses the fuzzy rules in the rule-base to produce fuzzy conclusions (e.g., the implied fuzzy sets), and the defuzzification block converts these fuzzy conclusions into the crisp outputs.

#### Universes of Discourse

The ordinary (crisp) sets $U_i$ and $Y_i$ are called the universes of discourse for $u_i$ and $y_i$, respectively (in other words, they are their domains). In practical applications, most often the universes of discourse are simply the set of real numbers or some interval or subset of real numbers.

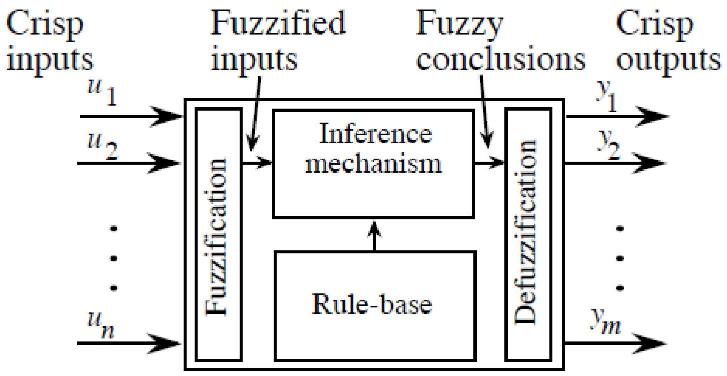


**Figure 3.2:** Fuzzy controller diagram [11].

#### Linguistic Variables

To specify rules for the rule-base, the expert will use a linguistic description; hence, linguistic expressions are needed for the inputs and outputs and the characteristics of the inputs and outputs. For the fuzzy system, linguistic variables denoted by $\tilde{u}_i$ are used to describe the inputs $u_i$. Similarly, linguistic variables denoted by $\tilde{y}_i$ are used to describe the outputs $y_i$.

### Linguistic Values

The $u_i$ and $y_i$ take on values over each universe of discourse $U_i$ and $Y_i$, respectively, linguistic variables $\tilde{u}_i$ and $\tilde{y}_i$ take on linguistic values that are used to describe characteristics of the variables. Let $\tilde{A}_i{}^j$ denote the $j^{th}$ linguistic value of the linguistic variable $\tilde{u}_i$ defined over the universe of discourse $U_i$. If we assume that there exist many linguistic values defined over $U_i$, then the linguistic variable $\tilde{u}_i$ takes on the elements from the set of linguistic values denoted by

$$\tilde{A}_i = \{\tilde{A}_i{}^j : j = 1, 2, ..., N_i\} \tag{3.1}$$

Similarly, let $\tilde{B}_i{}^j$ denote the $j^{th}$ linguistic value of the linguistic variable $\tilde{y}_i$ defined over the universe of discourse $Y_i$. The linguistic variable $\tilde{y}_i$ takes on elements from the set of linguistic values denoted by

$$\tilde{B}_i = \{\tilde{B}_i{}^p : p = 1, 2, ..., M_i\} \tag{3.2}$$

Linguistic values are generally descriptive terms such as "positive large", "zero", and "negative big" (i.e., adjectives). For example, if we assume that $\tilde{u}_1$ denotes the linguistic variable "speed", then we may assign $\tilde{A}_1{}^1$ = "slow", $\tilde{A}_1{}^2$ = "medium", and $\tilde{A}_1{}^3$ = "fast"; so that $\tilde{u}_1$ has a value from $\tilde{A}_1$ = $\{\tilde{A}_1{}^1, \tilde{A}_1{}^2, \tilde{A}_1{}^3\}$.

### Linguistic Rules

The mapping of the inputs to the outputs for a fuzzy system is in part characterized by a set of *conditions and action* rules, or (IF-THEN) form.

**IF** premise **THEN** consequence.

Usually, the inputs of the fuzzy system are associated with the premise, and the outputs are associated with the consequent. These IF-THEN rules can be represented in many forms. Two standard forms, multi-input multi-output

(MIMO) and multi-input single-output (MISO), are considered here. The MISO form of a linguistic rule is

**IF** $\tilde{u}_1$ is $\tilde{A}_1{}^j$ and $\tilde{u}_2$ is $\tilde{A}_2{}^k$,..., and $\tilde{u}_n$ is $\tilde{A}_n{}^l$ **THEN** $\tilde{y}_q$ is $\tilde{B}_q{}^p$.

It is an entire set of linguistic rules of this form that the expert specifies on how to control the system. Note that if $\tilde{u}_1$ = *velocity error* and $\tilde{A}_1{}^j$ = *positive large*, then $\tilde{u}_1$ is $\tilde{A}_1{}^j$, a single term in the premise of the rule, means velocity error is positive large. It can be easily shown that the MIMO form for a rule (i.e., one with consequents that have terms associated with each of the fuzzy controller outputs) can be decomposed into a number of MISO rules using simple rules from logic. For instance, the MIMO rule with $n$ inputs and $m$=2 outputs.

**IF** $\tilde{u}_1$ is $\tilde{A}_1{}^j$ and $\tilde{u}_2$ is $\tilde{A}_2{}^k$,..., and $\tilde{u}_n$ is $\tilde{A}_n{}^l$ **THEN** $\tilde{y}_1$ is $\tilde{B}_1{}^r$ and $\tilde{y}_2$ is $\tilde{B}_2{}^p$.

linguistically (logically) equivalent to the two rules.

**IF** $\tilde{u}_1$ is $\tilde{A}_1{}^j$ and $\tilde{u}_2$ is $\tilde{A}_2{}^k$,..., and $\tilde{u}_n$ is $\tilde{A}_n{}^l$ **THEN** $\tilde{y}_1$ is $\tilde{B}_1{}^r$

**IF** $\tilde{u}_1$ is $\tilde{A}_1{}^j$ and $\tilde{u}_2$ is $\tilde{A}_2{}^k$,..., and $\tilde{u}_n$ is $\tilde{A}_n{}^l$ **THEN** $\tilde{y}_2$ is $\tilde{B}_2{}^p$

### 3.2.2 Fuzzy Sets, Fuzzy Logic, Rule-Base

The fuzzy sets and fuzzy logic are used to heuristically quantify the meaning of linguistic variables, linguistic values, and linguistic rules that are specified by the expert. The concept of a fuzzy set is introduced by first defining a membership function.

#### Membership Functions

Let $U_i$ denotes a universe of discourse and $\tilde{A}_i{}^j \in \tilde{A}_i$ denotes a specific linguistic value for the linguistic variable $\tilde{u}_i$. The function $\mu(u_i)$ associated with $\tilde{A}_i{}^j$ that maps $U_i$ to [0, 1] is called a membership function. This membership function describes the certainty that an element of $U_i$, denoted $u_i$, with a linguistic description $\tilde{u}_i$, may be classified linguistically as $\tilde{A_i}^j$. Membership functions are subjectively specified in a heuristic manner from the human's experience or intuition.

**Fuzzy Sets**

Given a linguistic variable $\tilde{u}_i$ with a linguistic value $\tilde{A_i}^j$ defined on the universe of discourse $U_i$, and the membership function $\mu_{\tilde{A_i}^j}$ $(u_i)$ that maps $U_i$ to [0, 1], a fuzzy set denoted with $A_i^j$ is defined as follows:

$$A_i^j = \{(u_i, \mu_{\tilde{A_i}^j}(u_i)) : u_i \in U_i\} \tag{3.3}$$

**Fuzzy Logic**

Next, some sets of theoretical and logical operations on fuzzy sets are introduced.

*a. Fuzzy Intersection (AND):*

The intersection of fuzzy sets $A_i^1$ and $A_i^2$, which are defined on the universe of discourse $U_i$, is a fuzzy set denoted by $A_i^1 \cap A_i^2$ , with a membership function defined by either of the following two methods:

*Minimum:*

The minimum of the membership values is defined as:

$$\mu_{A_i^1 \cap A_i^2} = min\{(\mu_{A_i^1}(u_i), \mu_{A_i^2}(u_i)) : u_i \in U_i\} \tag{3.4}$$

*Algebraic Product:*

The algebraic product of the membership values is defined as

$$\mu_{A_i^1 \cap A_i^2} = \{(\mu_{A_i^1}(u_i) * \mu_{A_i^2}(u_i)) : u_i \in U_i\} \tag{3.5}$$

*b. Fuzzy Union (OR):*

The intersection of fuzzy sets $A_i^1$ and $A_i^2$, which are defined on the universe of discourse $U_i$, is a fuzzy set denoted by $A_i^1 \cup A_i^2$, with a membership function defined by either of the following two methods:

*Maximum:*

The minimum of the membership values is defined as:

$$\mu_{A_i^1 \cup A_i^2} = max\{(\mu_{A_i^1}(u_i), \mu_{A_i^2}(u_i)) : u_i \in \mathfrak{U}_i\} \tag{3.6}$$

*Algebraic Sum:*

The algebraic sum of the membership values is defined as:

$$\mu_{A_i^1 \cup A_i^2}(u_1, u_2, ..., u_n) = \{(\mu_{A_i^1}(u_i) + \mu_{A_i^2}(u_i) - \mu_{A_i^1}(u_i) * \mu_{A_i^2}(u_i)) : u_i \in U_i\} \tag{3.7}$$

The notation $\oplus$ denotes for the union of the two fuzzy sets. In fuzzy logic, union is used to present the "or" operation. For example, use maximum to present the "or" operation, then the "black" membership function in Figure. 3.3, is $\mu_{A_i^1 \cup A_i^2}$, which is formed from the two others ($\mu_{A_i^1}(u_i)$ and $\mu_{A_i^2}(u_i)$).

$$\mu_{A_i^1 \oplus A_i^2} = \{(\mu_{A_i^1}(u_i) + \mu_{A_i^2}(u_i) - \mu_{A_i^1}(u_i) * \mu_{A_i^2}(u_i)) : u_i \in U_i\} \tag{3.8}$$

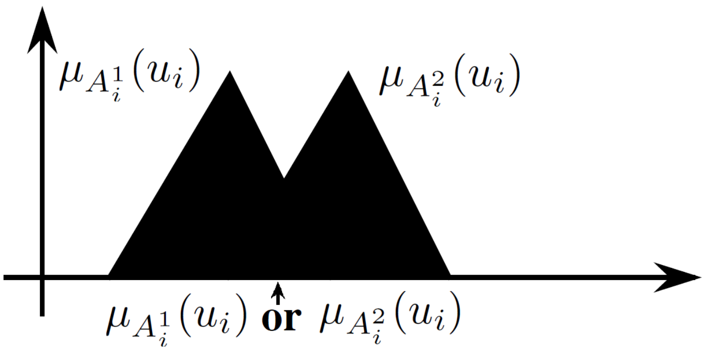


**Figure 3.3:** A membership function for the “or” of two membership functions.

*c. Fuzzy Cartesian Product:*

The above intersection and the union operations are both defined for fuzzy sets that lie on the same universe of discourse. The fuzzy Cartesian product is used to quantify operations on many universes of discourse. If $A_1^j$, $A_2^k$, ..., $A_n^l$

are fuzzy sets defined on the universes of discourse $U_1$, $U_2$, ..., $U_n$ respectively, their Cartesian product is a fuzzy set, denoted by $A_1^j$ x $A_2^k$ x ... x $A_n^l$ with a membership function defined by

$$\mu_{A_1^j A_2^k ... A_n^l}(u_i, u_2, ..., u_n) = \mu_{A_1^j}(u_1) * \mu_{A_2^k}(u_2) * ... * \mu_{A_n^l}(u_n) \tag{3.9}$$

in which "*" is "and" operation.

**Fuzzy Quantification of Rules: Fuzzy Implications**

This part shows how to quantify the linguistic elements in the premise and consequent of the linguistic IF-THEN rule with fuzzy sets. For example, suppose we are given the IF-THEN rule in MISO form as mentioned in Linguistic Rules. We can define the fuzzy sets as follows:

$$\begin{aligned}
A_1^j &= \{(u_1, \mu_{A_1^j}(u_1)) : u_1 \in U_1\} \\
A_2^k &= \{(u_2, \mu_{A_2^k}(u_2) : u_2 \in U_2\} \\
&\vdots \\
A_n^l &= \{(u_n, \mu_{A_n^l}(u_n)) : u_n \in U_n\} \\
B_q^p &= \{(y_q, \mu_{B_q^p}(u_q)) : B_q \in Y_q\}
\end{aligned} \tag{3.10}$$

These fuzzy sets quantify the terms in the premise and the consequent of the given IF-THEN rule, to make a fuzzy implication.

**IF** ${A_1}^j$ and ${A_2}^k$, ... , and ${A_n}^l$ **THEN** is ${B_q}^p$.

### 3.2.3 Fuzzification

Fuzzy sets are used to quantify the information in the rule-base, and the inference mechanism operates on fuzzy sets to produce fuzzy sets; hence, we must specify how the fuzzy system will convert its numeric inputs $u_i \in U_i$ into fuzzy sets (a process called fuzzification) so that they can be used by the fuzzy system.

Let $U_i^*$ denote the set of all possible fuzzy sets that can be defined on $U_i$. Given $u_i \in U_i$, fuzzification transforms $u_i$ to a fuzzy set denoted by $\hat{A}_i^{fuz}$

defined on the universe of discourse $U_i$. This transformation is produced by the fuzzification operator $\mathcal{F}$ defined by:

$$\mathcal{F} : U_i \rightarrow U_i^* \tag{3.11}$$

where

$$\mathcal{F}(u_i) = \hat{A}_i^{fuz} \tag{3.12}$$

### 3.2.4 The Inference Mechanism

The inference mechanism has two basic tasks: (1) determining the extent to which each rule is relevant to the current situation as characterized by the inputs, $u_i$, $i$ = 1, 2, . . ., $n$ (*matching*); and (2) drawing conclusions using the current inputs $u_i$ and the information in the rule-base (we call this task an inference step). For matching, note that $A_1^j$ x $A_2^k$ x ... x $A_n^l$ is the fuzzy set representing the premise of the $i^{th}$ rule $(j,k,...,l;p,q)_i$ (there may be more than one such rule with this premise).

#### Matching

Suppose that at some point we get the inputs $u_i$, $i$ = 1, 2, . . ., $n$ and the fuzzification produces

$\hat{A}_1^{fuz}, \hat{A}_2^{fuz}, ...., \hat{A}_n^{fuz}$

the fuzzy sets representing the inputs. There are then two basic steps to matching.

*Step 1: Combine Inputs with Rule Premises:*
The first step in matching involves finding fuzzy sets $\hat{A}_1^j$, $\hat{A}_2^k$, ... ,$\hat{A}_n^l$ with membership functions

$$\begin{aligned}
&\mu_{\hat{A}_1^j}(u_1) = \mu_{A_1^j}(u_1) * \mu_{\hat{A}_1^{fuz}}(u_1) \\
&\mu_{\hat{A}_2^k}(u_2) = \mu_{A_2^k}(u_2) * \mu_{\hat{A}_2^{fuz}}(u_2) \\
&\vdots \\
&\mu_{\hat{A}_n^l}(u_n) = \mu_{A_n^l}(u_n) * \mu_{\hat{A}_n^{fuz}}(u_n)
\end{aligned} \tag{3.13}$$

that combine the fuzzy sets from fuzzification with the fuzzy sets used in each of the terms in the premises of the rules.

*Step 2: Determine Which Rules Are On:*
In the second step, we form membership values $\mu_i(u_1, u_2, \ldots, u_n)$ for the $i^{th}$ rule premise that represent the certainty that each rule premise holds for the given inputs. Define

$$\mu_i(u_1, u_2, ..., u_n) = \mu_{\hat{A}_1^j}(u_1) * \mu_{\hat{A}_2^k}(u_2) * ... * \mu_{\hat{A}_n^l}(u_n) \tag{3.14}$$

It represents the certainty of a premise of a rule and thereby represents the degree to which a particular rule holds for a given set of inputs.

**Inference Step**
There are two standard alternatives to performing the inference step, one that involves the use of implied fuzzy sets and the other that uses the overall implied fuzzy set.

Next, the inference step is taken by computing, for the $i^{th}$ rule $(j, k, \ldots, l; p, q)$ the implied fuzzy set $\hat{B}_q^i$ with the membership function

$$\mu_{\hat{B}_q^i}(y_q) = \mu_i(u_1, u_2, ....., u_n) * \mu_{B_q^p}(y_q) \tag{3.15}$$

The implied fuzzy set $\hat{B}_q^i$ specifies the certainty level that the output should be a specific crisp output $y_q$ within the universe of discourse $Y_q$ taking into consideration only rule $i$. Note that since $\mu_i(u_1, u_2, ..... ,u_n)$ will vary with time, so will the shape of the membership functions $\mu_{\hat{B}_q^i}(y_q)$ for each rule.

Alternatively, the inference mechanism could, in addition, compute the overall implied fuzzy set $\hat{B}_q$ with membership function

$$\mu_{\hat{B}_q}(y_q) = \mu_{\hat{B}_q^1}(y_q) \oplus \mu_{\hat{B}_q^2}(y_q) + \ldots + \oplus \mu_{\hat{B}_q^R}(y_q) \tag{3.16}$$

that represents the conclusion reached considering all the rules in the rule-base at the same time.

Up to this point, we have used fuzzy logic to quantify the rules in the rule-base, fuzzification to produce fuzzy sets characterizing the inputs, and the inference mechanism to produce fuzzy sets representing the conclusions that it reaches after considering the current inputs and the information in the rule-base. Next, we look at how to convert this fuzzy set quantification of the conclusions to a numeric value that can be an input to the plant.

### 3.2.5 Defuzzification

A number of defuzzification strategies exist, and it is not difficult to invent more. Each provides a means to choose a single output (which we denote with $y_q^{crisp}$) based on either the implied fuzzy sets or the overall implied fuzzy set.
As they are more common, we first specify typical defuzzification techniques for the implied fuzzy sets $\hat{B}_q^i$

*Center of gravity (COG):*
A crisp output $y_q^{crisp}$ is chosen using the center of area and area of each implied fuzzy set, and is given by:

$$y_q^{crisp} = \frac{\sum_{i=1}^{R} b_i^q \int_{Y_q} \mu_{\hat{B}_q^i}(y_q) dy_q}{\sum_{i=1}^{R} \int_{Y_q} \mu_{\hat{B}_q^i}(y_q) dy_q} \tag{3.17}$$

where $R$ is the number of rules, $b_i^q$ is the center of area of the membership function of $B_q^p$ associated with the implied fuzzy set $\hat{B}_q^i$ for the $i^{th}$ rule $(j,k,...,l;p,q)_i$
and

$$\int_{Y_q} \mu_{\hat{B}_q^i}(y_q) dy_q \tag{3.18}$$

denotes the area under $\mu_{\hat{B}^i_q}(y_q)$.

*Center-average:*
A crisp output $y_q^{crisp}$ is chosen using the centers of each of the output membership functions and the maximum certainty of each of the conclusions represented with the implied fuzzy sets, and is given by

$$y_q^{crisp} = \frac{\sum\limits_{i=1}^{R} b_i^q sup_{y_q}\{\mu_{\hat{B}^i_q}(y_q)\}}{\sum\limits_{i=1}^{R} sup_{y_q}\{\mu_{\hat{B}^i_q}(y_q)\}} \tag{3.19}$$

where *sup* denotes the *supremum* (i.e., the least upper bound which can often be thought of as the maximum value). Also, $b_i^q$ is the center of area of the membership function of $B_q^p$ associated with the implied fuzzy set $\hat{B}^i_q$ for the $i^{th}$ rule $(j,k,...,l;p,q)_i$.

## 3.3 Case Study: Fuzzy-Based Autonomous Navigation Strategy of a Geophysical Measurement Platform for Archaeological Prospection

In this section, the challenges of a geophysical measurement platform are firstly stated. Secondly, a fuzzy based traverse-terrain behavior approach is developed. Thirdly, a multiple behaviors integration is used. Lastly, the experimental results are presented.

### 3.3.1 The Challenges of a Geophysical Measurement Platform for Archaeological Prospection

The development of the autonomous geophysical measurement platform (AGMP) for archaeological prospection has several challenges, such as an effective interaction with environments and an intelligent autonomous navigation.

To have a successful interaction with the working environments, the AGMP should be able to determine the route that should be followed, i.e., route planning. In a first instance, the AGMP will be conceived and tailored to be able to follow the predetermined routes. Once the information of the field and the obstacles have been determined using the digital maps or GPS measurements, an efficient path resulting in parallel tracks can be defined. To determine the possible obstacles, the field is split into simple shapes (e.g., parallel tracks), which are merged into blocks as depicted in Figure 3.4.

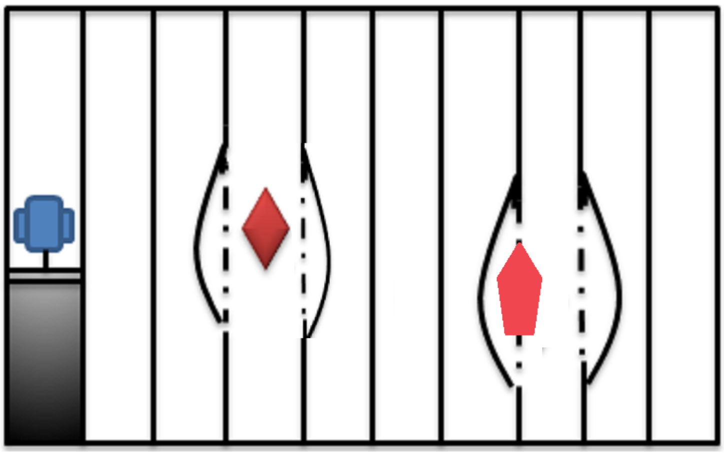


**Figure 3.4:** The path planning strategy for the AGMP (blue) in a field with obstacles (red). The black lines denote the pattern of intersection. The gray zone: the effected area. The white zone: the area to be effected.

To make an autonomous AGMP, one solution is to equip the platform with mechanisms to actuate its steering, throttle, brakes and gearbox. It is very dangerous for the AGMP to avoid some possible pitfalls at a high speed. For instance, the archaeological equipment in contact with rough ground may get damaged when it abruptly moves. Therefore, the balance between the performance (time efficiency) and the robustness (avoiding instrumental damage) is a delicate task.

The last challenge is an adaptive speed control to accommodate an intelligent system which may autonomously decide to accelerate in parts of the exploration area where no archaeological elements have been detected during the previous prospection conducted, and decelerate in those parts where the archaeological elements need to be characterized with greater accuracy by the AGMP. This can be achieved by taking into account the past positions and the related archaeological data in a model to predict the future area where the archaeological features are more likely to exist. Figure 3.5 depicts the concepts of such an intelligent AGMP. The data from the previously inspected area is fed to the robot, along with a prediction of the distance to the next point of interest. The yellow shape denotes the area of an archaeological area of interest. Within this region, the robot decreases its speed to be able to capture all the underground details. Outside this area, it accelerates as to gain time in the operation.

Obviously, the AGMP should work in outdoor environments with different terrains, such as smooth, rough or rocky ones. Therefore, the AGMP should also be able to adjust its speed with changes in terrain. To accommodate challenging terrains, a perception-based linguistics framework is proposed. The premise of the approach is to embed the heuristic knowledge into the AGMP's navigation strategy. The method is highly robust in coping with the uncertainty and imprecision that are inherent in perception of the natural environments. The strategy in this paper is comprised of two independent behaviors: a regional traverse-terrain and an obstacles avoidance, that decide the motion commands for the AGMP. However, as the AGMP is used for archaeological prospection, it will switch to a special mode as soon as it

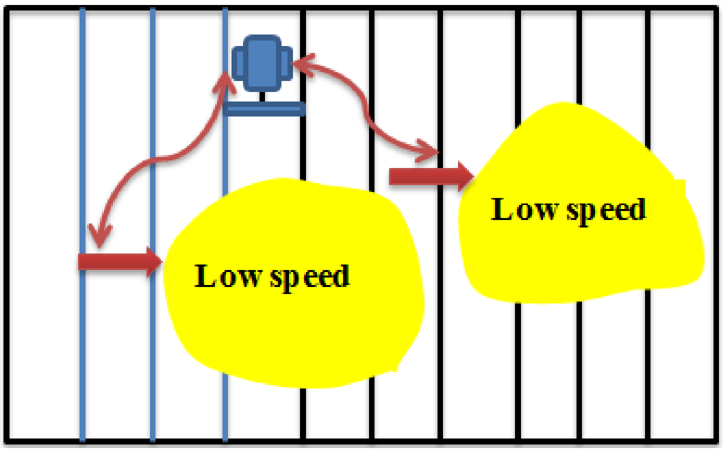


**Figure 3.5:** Representation of the concept for a speed control strategy in an intelligent AGMP for the archaeological data collection areas.

detects the areas where the archaeological elements need to be characterized with the greater accuracy.

In summary, for the AGMP, the first problem is addressed as path following. In [138], we have proposed a fuzzy-based approach for tracking different desired trajectories. The second problem deals with an autonomous navigation algorithm in an outdoor environment. The last challenge is a special control mode in areas where the archaeological elements must be carefully considered. These last two problems are solved by the proposed method as presented in the following subsections.

### 3.3.2 Fuzzy-Based Traverse Terrain Behavior

In this section, the addressed problem is to navigate the AGMP in suitable ways in different terrains. This section is divided into two parts. In the first part, a strategy of terrain assessment is proposed followed by a strategy for terrain based navigation. The terrain assessment and the AGMP navigation rules represent the driving actions of a skillful archaeologist.

- *Terrain roughness assessment*

  The most important attributes that characterize this difficulty are roughness and slope. The roughness characteristic relates to the coarseness and irregularity of the surface to be traversed. In this conceptual study, several terrains are considered as shown in Figure 3.6. The roughness in this study is presented by {Smooth, Rough, Rocky}, defined by the membership functions shown in Figure 3.7.

- *Terrain slope assessment*

  The slope is defined as the inclination/declination of the ground plane to be traversed. The terrain slope value is converted into three linguistic fuzzy sets {*Flat, Slope, Steep*}, defined by the membership functions shown in Figure 3.8. The human's knowledge is embedded in the definition of the membership functions. The chosen membership functions depend on the wheel design and mobile platform mechanism which determine the AGMP's capabilities of hill climbing and rock climbing. If the mobile platform cannot climb a rock then the rock is considered as an obstacle and not taken into account for the roughness of the road.

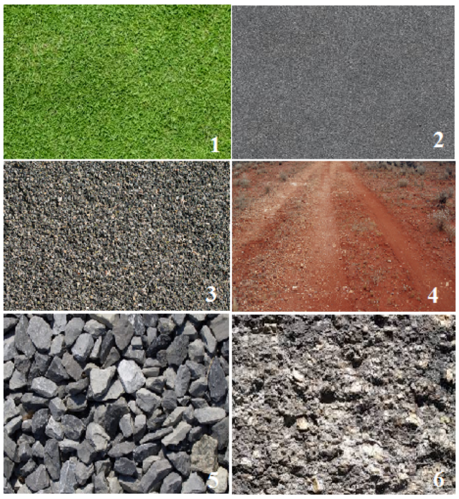


**Figure 3.6:** The terrain roughness assessment of the geophysical measurement platform for the archaeological prospection. 1,2. Smooth; 3,4. Rough; 5,6. Rocky.

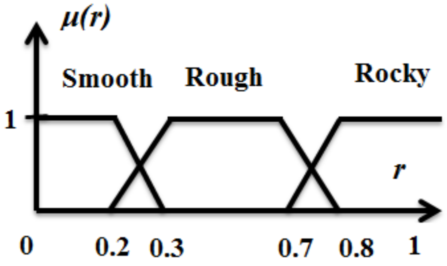


**Figure 3.7:** Membership functions for terrain roughness.

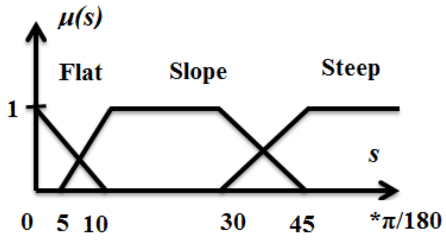


**Figure 3.8:** Membership functions for terrain slope.

- *Fuzzy rule based terrain traversability $\tau$*

  The fuzzy rule based terrain traversability involves the roughness and the slope of the traversed terrain. The rule base is developed based on the embedded human knowledge of terrain traversabilty. The linguistic fuzzy sets {*Low, Medium, High*} represent the traversability value with the membership functions shown in Figure 3.9. The rules correspond to the terrains that are dangerous, reasonable and convenient traversing, respectively. The FAM (Fuzzy Associative Memory) table for the rule bases are constructed as shown in TABLE 3.1. The rule based surface which represents the relationship among terrain roughness, terrain slope and terrain traversability is presented in Figure 3.10.

**Table 3.1:** The fuzzy logic rule base for terrain traversability.

| | Terrain Slope | | |
|---|---|---|---|
| Terrain Roughness | Flat | Slope | Steep |
| Smooth | High | High | Low |
| Rough | High | Medium | Low |
| Rocky | Medium | Low | Low |

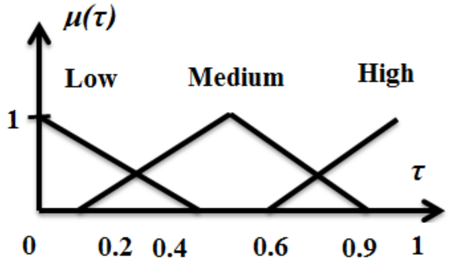


**Figure 3.9:** Membership functions for terrain traversability.

- *Terrain fuzzy-based navigation*

  Then, the rules for motion control variables including the translational speed $v$ are proposed. These rules mimic the driving decisions of an archaeologist navigating the mobile platform on different terrains. The translation speed $v$ is denoted by three linguistic fuzzy sets {*Slow, Medium, Fast*} with the membership functions shown in Figure 3.11.

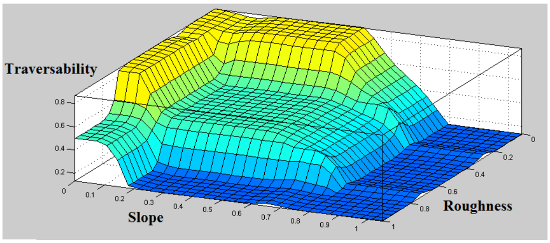


**Figure 3.10:** The surface rules for terrain traversability.

Notice that the angular speed is not controlled here since without obstacles, the mobile platform is expected to move straight. The speed rules are as follows:
1. If $\tau$ is High Then $v$ is Fast.
2. If $\tau$ is Medium Then $v$ is Medium.
3. If $\tau$ is Low Then $v$ Slow.

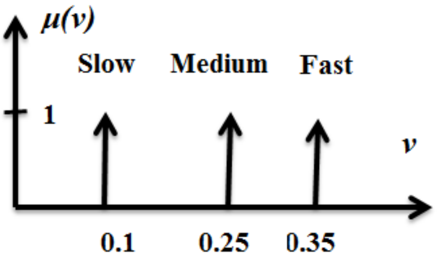


**Figure 3.11:** Membership functions for translational speed (m/s).

### 3.3.3 Local Obstacle Avoidance Fuzzy-Based Behavior

In a similar manner as the terrain navigation behavior, the angular speed $\omega$ is presented by three linguistic fuzzy sets {*Zero, Medium, Big*} with the membership functions shown in Figure 3.12. The turn rates are used to change the AGMP's direction. The rotation speed depends on the distance $d$ of the AGMP with respect to an obstacle which is denoted by three linguistic fuzzy sets {*Very near, Near, Far*} as shown in Figure 3.13. The rules of collision avoidance navigation using turn rules and motion rules are proposed as below.

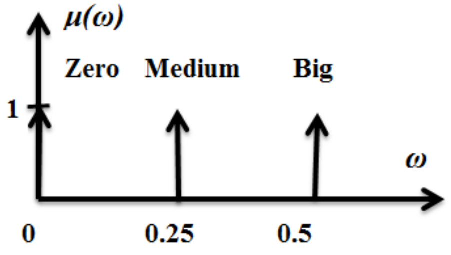


**Figure 3.12:** Membership functions for angular speed (rad/s).

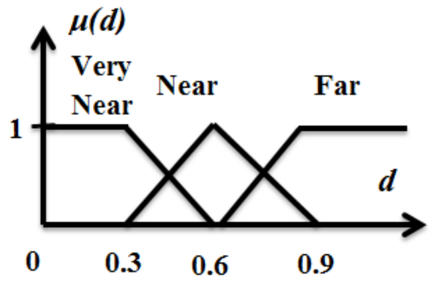


**Figure 3.13:** Membership functions for obstacle distance (m).

### I. Turn Rules

Clearly, to avoid obstacles, the AGMP needs to change its direction. The changing angle is decided based on factors such as the obstacles dimensions, the distance between the AGMP and the obstacles, the current AGMP's speed. However, on the test fields, the obstacles are rocks, trees or people. Those obstacles have that smaller dimensions compared to the geophysical measurement platform. Normally, the field is covered with a sequence of straight parallel tracks. The distance between two adjacent tracks should be equal to the effective operating width of the geophysical measurement tool. When the field has obstacles, AGMP should be able to avoid them then to return on the same track. The fuzzy-based turn rules are as follows:

1. If the $d$ is Far Then $\omega$ is Zero.
2. If the $d$ is Near Then $\omega$ is Medium.
3. If the $d$ is Very near Then $\omega$ is Big.

#### II. Motion Rules

The AGMP's speed is based on the distance between the platform and the obstacles in front of it. When the obstacle is far away, then the mobile platform can freely move with its high speed. However, when the AGMP is close to an obstacle it should decelerate. The rules proposed for the translational speed are quite simple, however they simulate human driver behaviors very well.
1. If the $d$ is Far Then $v$ is Fast.
2. If the $d$ is Near Then $v$ is Medium.
3. If the $d$ is Very near Then $v$ is Slow.

### 3.3.4 Multiple Behaviors Integration

As mentioned before, the data collection is the most important behavior for the AGMP, therefore this behavior has the highest priory. This means each time the AGMP detects archaeological elements, it immediately decreases its speed. If in the next small area, it stills detect some valuable data for archaeological purpose, it keeps its slow speed to be able to capture all the underground details. Otherwise, it changes to normal speed to gain time in operation. The translational velocity is selected for the data collection mode (e.g. 0.1 (m/s)).

In an area without any archaeological elements, the following behaviors integration method is applied. In the preceding two sections, fuzzy rule sets are proposed for two independent behaviors of the terrain traversing and the local obstacle avoidance. The fuzzy rule sets are purely concerned as a particular objective for each behavior, disregarding the others. However, in the the AGMP's working field, the two behaviors should be reconciled and fused to obtain an autonomous navigation strategy.

The behavior fusion approach employed in this section uses appropriate weighting factors. The weight factors determine the degree of influence of each behavior on the final mobile platform motion command. The two

behaviors integration scheme is summarized as presented in Figure 3.14. In this study, $\alpha$ and $\beta$ are assigned as weight factors of the traverse-terrain and the local avoid-obstacle behaviors. The weights are represented by three linguistic fuzzy sets {*Small, Medium, High*} as shown in Figure 3.15. The terrain fuzzy based weight rules are as follows:
1. If $\tau$ is Low Then $\alpha$ is High.
2. If $\tau$ is Medium Then $\alpha$ is Medium.
3. If $\tau$ is High Then $\alpha$ is Small.
The first rule implies that when the terrain is difficult to travel by the mobile

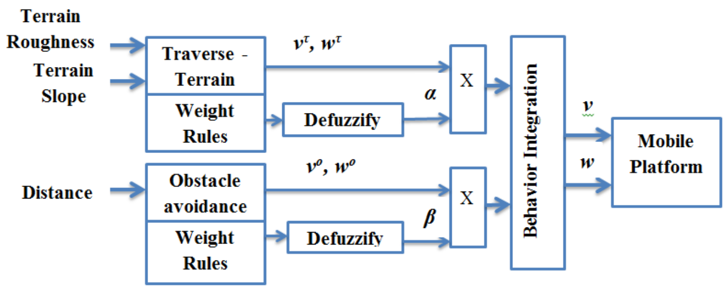


**Figure 3.14:** Two behaviors integration scheme of the AGMP's autonomous navigation strategy.

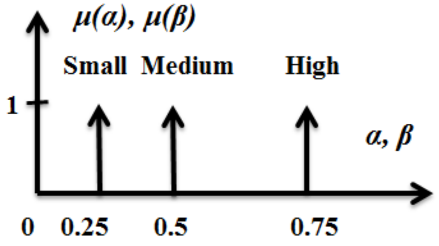


**Figure 3.15:** Membership functions for weight factors in the behavior fusion approach.

platform (for example, the road is rocky and steep), then the fuzzy rule set for traverse-terrain assigns a high weighting value $\alpha$. Conversely, the last rule infers that when the mobile platform moves in very convenient terrain (for example, the road is very smooth and flat) then a small value is assigned. Similarly, the obstacle avoidance fuzzy-based weight rules are as follows:

1. If $d$ is Very near Then $\beta$ is High.
2. If $d$ is Near Then $\beta$ is Medium.
3. If $d$ is Far Then $\beta$ is Small.

The first rule suggests that when the mobile platform is in a dangerous situation (e.g, the obstacle location is very near) then the "avoid obstacle behavior" assigns a high value. Observe that the weight rules for the "terrain traversing" and the "avoid obstacle" behaviors are independent. Therefore, it is necessary to combine both of them to find the final motion commands.

For each navigation behavior, the fuzzy base rule set generates an independent motion for the mobile platform ($v$, $\omega$). The final commands are computed as follows:

$$\begin{aligned} v &= \frac{\alpha v_p^{\tau} + \beta v_p^{o}}{\alpha + \beta} \\ w &= \frac{\alpha w_p^{\tau} + \beta w_p^{o}}{\alpha + \beta} \end{aligned} \tag{3.20}$$

where $v_p^{\tau}$, $v_p^{o}$ are the velocities for the terrain navigation behavior and the local obstacle avoidance. Similarly, $w_p^{\tau}$, $w_p^{o}$ are the corresponding angular velocity for the two mentioned behaviors.

The state of the mobile platform coordinates is given by (3.21):

$$q = \begin{bmatrix} x_c & y_c & \theta \end{bmatrix}^T \tag{3.21}$$

where the variables $x_c$, $y_c$ and $\theta$ respectively stand for the coordinate of robot center of gravity in the $x$-axis and the $y$-axis and the heading angle. Assuming that there is no-slip condition at wheels, the velocity of the wheel centers are parallel to the heading direction. The formula of $\dot{q}$ can be expressed as:

$$\dot{q} = \begin{bmatrix} cos\theta & 0 \\ sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} \tag{3.22}$$

where $v$ and $w$ are the translational velocity and the angular velocity of the mobile platform. For the platform tread width of $B$ and the wheel radius of $r$, the right and left tread velocities $\dot{\theta}_r$ and $\dot{\theta}_l$ can be found as:

$$\begin{aligned}\dot{\theta}_r &= \frac{v+Bw}{r}\\ \dot{\theta}_l &= \frac{v-Bw}{r}\end{aligned} \tag{3.23}$$

## 3.3.5 Hardware Specifications

In this research, the experiments have been conducted by using the Lego EV3 type treaded robot. For detecting the obstacles, an ultrasonic sensor is chosen. The program is developed using leJOS EV3 Eclipse in Java. The brain and embedded computer of the Lego EV3 Mindstorm robot is the EV3 Brick which is able to directly connect with its sensors and motors. It contains the main computational units and the communication interfaces. The robot is driven by two independent EV3 Large Servo Motors (see Figure. 3.16). The details of each component are presented as follows.

**Programmable Brick**

The EV3 Brick serves as a control center and a power station for the robot. It

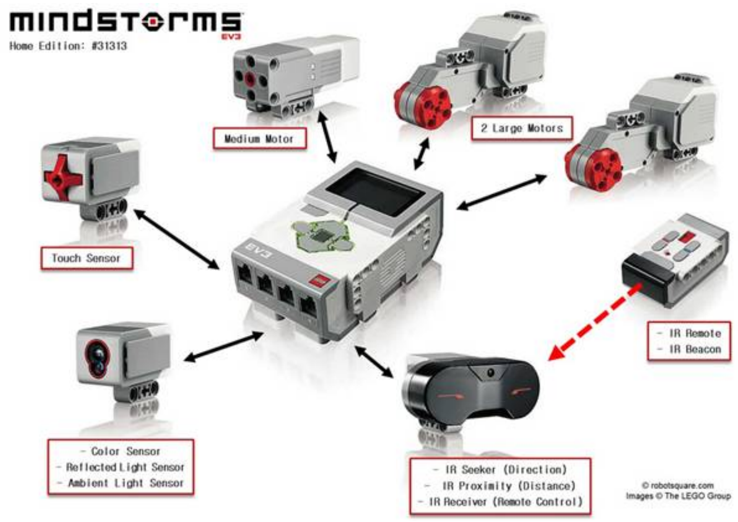


**Figure 3.16:** EV3 Lego robot and its components.

includes:

- 4 Input ports (to connect sensors to the EV3 Brick);
- 1 Mini USB PC port (to connect the EV3 brick to a computer);
- USB host port (to add Wi-Fi dongle);
- Micro SD Card port (to increase available memory for the EV3 Brick);
- Built-in speaker.

**EV3 Ultrasonic Sensor**
The digital EV3 Ultrasonic Sensor generates sound waves and reads their echoes to detect and measure the distance from objects. It can also send single sound waves to work as sonar or listen for a sound wave that triggers the start of a program. The main characteristics of the EV3 ultrasonic sensor are presented as follows.

- Measures distances between 1 cm and 250 cm.
- Accurate to $\pm$ 1 cm.
- Front illumination is constant while emitting and it blinks while listening.
- Returns true if other ultrasonic sound is recognized.

**EV3 Large Servo Motors**
The EV3 Large Servo Motor is a powerful motor that uses tacho feedback for a precise control process within one degree of accuracy. By using the built-in rotation sensor, the intelligent motor can be made to align with other motors on the robot so that it can drive in a straight line at the same speed. It can also be used to give an accurate reading for experiments. The main characteristics of the EV3 large servo motors are presented as follows.

- Tacho feedback to one degree of accuracy.
- 160-170 RPM.
- Running torque of 20 N.cm.
- Stall torque of 40 N.cm.

### 3.3.6 Sensor Accuracy

There are several options for an obstacle detection task. To decide which sensor to use in this study, an experiment has been performed. Both the ultrasonic sensor and the infrared sensor were tested. A comparison between the real and the measured values is shown in Table 3.2. It is clear that the ultrasonic sensor is the most accurate one. The measured distances of the infrared sensor seems to be different depending on the material of the measured object. Therefore, the ultrasonic sensor is used for obstacle detection in this work.

**Table 3.2:** The distance measurement of different sensors.

| Real Distance | Infrared sensor black | Infrared sensor color | Ultrasonic Sensor |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 15 | 18 | 24 | 15.9 |
| 30 | 34 | 40 | 30.2 |
| 45 | 44 | 48 | 45.6 |
| 60 | 51 | ∞ | 60.2 |
| 75 | ∞ | ∞ | 75.1 |
| 90 | ∞ | ∞ | 91.1 |
| 105 | ∞ | ∞ | 105.3 |
| 120 | ∞ | ∞ | 120.1 |

### 3.3.7 Experimental Results

The experiment is implemented in two test cases with different scenarios. The first test demonstrates the performance of the mobile platform in the smooth terrain with different terrain slope, no obstacles, one archaeological area as shown in Figure 3.17.

The slope in the tested region increases gradually. In case no archaeological elements are detected, only terrain behaviors are considered for the speed control. Applying the rule for the fuzzy logic rule base, the terrain traversability $\tau$ has the linguistic values {*High, High and Slow*} respectively. Then, the rules for motion control variables are applied according to the terrain fuzzy based

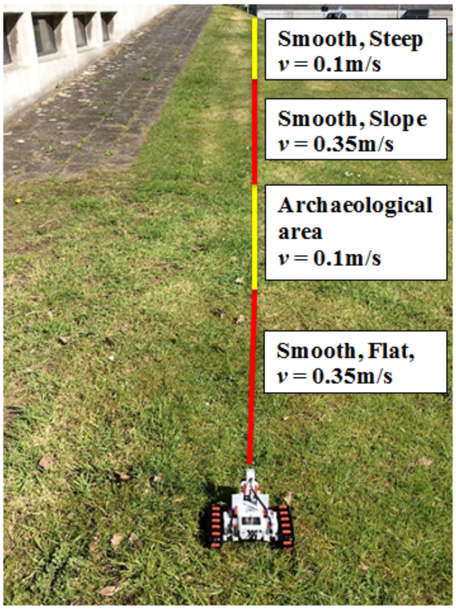


**Figure 3.17:** Speed control strategy for test case 1.

navigation. Notice that the angular speed is set to be zero since the mobile platform is designed to move straightly without obstacles. As a consequence, the translation speed of the robot has the linguistic values {*Fast, Fast, Slow*}, respectively. According to the membership functions, the translational speed is performed suitable for each terrain e.g. drives 0.35 m/s (at maximal translation speed) for the High value of terrain traverability and 0.1m/s for the Low value terrain traversability. On first 2 meters, no archaeological elements are detected, robot's speed is 0.35 m/s. However, as soon as the archaeological elements are detected, the robot immediately changes its translational speed to 0.1m/s for next 1 meter. Then it changes its speed to 0.35 m/s for next 1 meter since it is out of the archaeological area. Finally, robot's speed is 0.1m/s for the Low value terrain traversability.

In the second test, the influence of the two simultaneous navigation behaviors, the traverse-terrain and the obstacle avoidance are demonstrated as shown in Figure 3.18.

The robot is navigated through regions having a high traversability (for example, rough and flat terrain) or a medium traversability (for example,

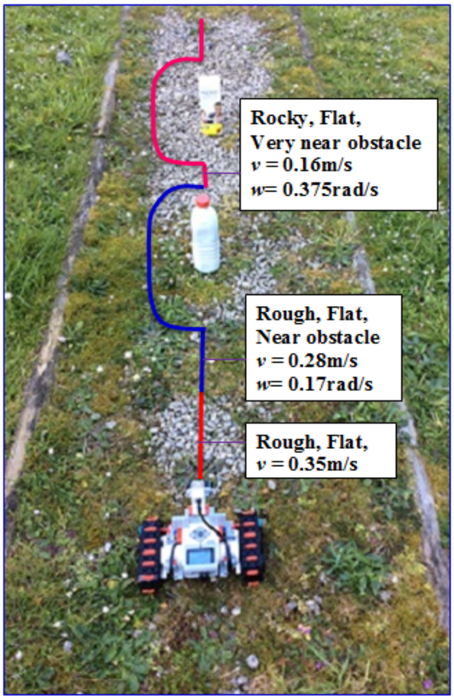


**Figure 3.18:** Speed control strategy for test case 2.

rocky and flat). Since the terrain has the high traversability behavior and the obstacle is located near the robot, then the weights on the traverse terrain have a small value and the obstacle avoidance becomes more important. At this point, the robot slowly moves and turns to avoid the obstacle. The translational speed and the angular speed of the robot are 0.28 m/s and 0.17 rad/s, respectively. After that, the robot will return to the straight line (as the robot needs to follow the parallel tracks), however, it detects the obstacle in the very near distance. Therefore, it will dramatically reduce its translational speed (0.16 m/s) and fast turn (0.375 rad/s) again for safety purposes.

## 3.4 Summary

First, an intelligent fuzzy based autonomous navigation strategy is introduced in this study for representing, manipulating, and implementing a human heuristic knowledge about how to control a system. Second, the control challenges such as interaction with environments, adaptive speed control to accommodate an intelligent system have been highlighted through the geophysical measurement platform for archaeological prospection. For instance, the archaeological equipment in contact with rougher ground may get damaged when moved abruptly. Therefore, the balance between the performance (time efficiency) and the robustness (avoiding instrumental damage) is a delicate task.

In the initial stage, the main AGMP's challenges are clearly addressed which relate to the terrain traversal behaviors, the local obstacle avoidance and the geophysical investigation behavior. The recommendations of these three behaviors are integrated to generate the final motion commands for the AGMP. As a consequence, the system is intelligently navigated in outdoor environments. The data collection is the most important behavior for the AGMP, therefore this behavior has the highest priority. The designed fuzzy system effectively works for the AGMP, which poses all the mentioned challenges.

The fuzzy-based autonomous navigation strategy has major advantages such as: *i*) the fuzzy logic rules that govern the AGMP's motions are simpler and more understandable since they can emulate the archaeologist driver perceptions, knowledge, and experiences and *ii*) the behavior-based strategy has a modular structure that can be extended to incorporate new behaviors.

As mentioned before, the fuzzy control effectively works for autonomous navigation when it inherits the knowledge and experiences of the experts. Moverover, there are many applications in which the mobile robot is requested to travel in an environment without any knowledge of operators. In those cases, it has difficulties in selecting the most suitable rules and membership functions. In addition, one may want to figure out an optimal collision-free path from a starting position to a goal position under certain constraints. Therefore, it is necessary to consider that the robot path planning is a constrained multi-objective optimization problem. This problem and the proposed solution are introduced in Chapter 4.

# Chapter 4

# Hierarchical Global Path Planning Approach on Multi Objective Particle Swarm Optimization

The quality of a robot path is one of the most important aspects of autonomous robot requirements which can be measured in terms of, for example, length, clearance, smoothness, and energy, or some combination of the above. Unfortunately, planners such as fuzzy control presented in Chapter 3 or neural networks produce solutions that may be far from optimal. The nature-inspired algorithms have advanced potentialities to solve the multi-objective robot path planning optimization problems. However, they are time consuming and unable to create a feasible path in some scenarios.

Therefore, a hierarchical approach, which combines several algorithms, is much preferred in the area of robot application [139]. Their advantages are computational efficiency and a straightforward implementation.

In this chapter, a multi-objective path planning is proposed in a cluttered environment. This method has the ability of finding an optimal global robot path method in terms of the path length and the path smoothness, considering the constraints of mobile robots with computational efficiency.

The results of this chapter are published as follows.

1. T. T. Mac, C. Copot, T. D. Tran, R. De Keyser, A Hierarchical Global Path Planning for Mobile Robot based on Multi-Objective Particle Swarm Optimization, *Applied Soft Computing*, vol. 59, pp. 68-76, 2017. (A1, rank: Q1, impact factor: 3.541).
2. T. T. Mac, C. Copot, T. D. Tran, R. De Keyser, A Hierarchical Global Path Planning based on Multi-Objective Particle Swarm Optimization, *IEEE 21st International Conference on Methods and Models in Automation and Robotics*, Miedzyzdroje, Poland, 29th August-1st September, pp. 930-935, 2016. (P1-ISI).

## 4.1 Introduction

As mentioned in Chapter 2, the existing path planning methods are mainly divided into two categories: *i*) classical algorithms and *ii*) heuristic-based algorithms [138]. The prominent classical methods consist of cell decomposition method (CD) [140], potential field method (PFM) [141], road-map method (RM) [142] and subgoal method (SG) [143]. Heuristic methods include neural networks (NN) [144, 145], fuzzy logic (FL) [146] and the nature-inspired methods from which the most famous ones are the genetic algorithm (GA) [147] and the particle swarm optimization (PSO) [148].

Recently, researchers have been patiently seeking for more powerful integrated methods for this problem. The aim is to figure out an optimal collision-free path from a starting position to a goal position under certain constraints. However, it is nearly impossible to introduce an exact definition for the term "optimal path", as there are many types of robots which have different characteristics and constraints. Therefore, it can be focused on several aspects, such as safety, smoothness, distance and energy with respect to specific constraints of mobile robots. In other words, the robot path planning can be considered as a constrained multi-objective optimization problem [149]. Some of the suggested integrated methods are good for dealing with simple environments. However, they work inefficiently and are time consuming for a cluttered environments.

A trend to apply path planning approaches with hierarchical structures has been proposed in [150–152] to solve the optimal path. In those studies, the lower level principally focuses on obtaining a geometric collision-free

path. To find such a path, graph search methods can be applied, such as A* [150, 151]. Then, the higher level is used to provide a series of subgoals to generate an optimal path. In [152], X. Yang et al. argue that due to the hierarchical structure design and the interpolative reasoning mechanism, the proposed path planning is very simple and concise. That brings several benefits, such as the reduction of computation time, re-usability of modules, and easy extensibility. In [153], a modified $A^*$ algorithm, called Multi-Neuron Heuristic Search (MNHS) is implemented in a hierarchical manner where each generation of the algorithm provides a more detailed path, with a higher reaching probability. The algorithm is able to give an optimal path in numerous situations, with varying degrees of complexities where the standard $A^*$ algorithm fails. Such a hierarchical approach is also very successful in structural engineering as presented in [154]. This algorithm uses graph theory, Matroids and the greedy algorithm for optimal cycle basis selection. In [155], the size/topology optimization of trusses is proposed using GA, the force method and some concepts of graph theory. The approach is improved by using a suitable penalty function to reduce the number of numerical operations and to increase the speed of the optimization towards a global optimum.
Consequently, this chapter proposes a novel hierarchical approach which combines: 1) a triangular decomposition; 2) a Dijkstra algorithm; and 3) a proposed constrained multi-objective particle swarm optimization (CMOPSO) with an accelerated update methodology based on Pareto dominance principle. The aim is to generate optimal collision-free paths focusing on minimizing the path length and maximizing the smoothness regarding the physical limitation of robots. The proposed approach has a three-level structure. In the first level, the triangular decomposition is used to divide the robot's working environment into obstacle configuration space and free configuration space. Next, in the second level, the Dijkstra's algorithm is applied to find the collision-free path from the starting point to the goal point. Finally, CMOPSO with an accelerated update methodology based on Pareto dominance principle is proposed to find an optimal path. Briefly, the main contributions are the development of: (1) a novel hierarchical global path planning method for mobile robots in cluttered environments; (2) a proposed particle swarm optimization algorithm with an accelerated update methodology based on Pareto dominance principle and (3) an increased computational efficiency.

## 4.2 PSO Algorithm

Considering the search space $\mathcal{D}$ that has dimension $N$ ($\mathcal{D} \subset \mathcal{R}^N$), the position and the velocity of the $i^{th}$ particle in the swarm are $X_i$ = ($X_{i1}, X_{i2}$, ..., $X_{iN}$) $\in \mathcal{D}$ and $V_i$ = ($V_{i1}, V_{i2}$, ..., $V_{iN}$) $\in \mathcal{D}$. The particles will update their locations in the swarm towards the global optimum (or target position) based on two factors: 1) the personal best position (*Pb*) and 2) the global best position (*Gb*). The first term is the best position found by the $i^{th}$ particle itself over iterations 1 ... $t$ which is termed local leader and represented as $Pb_i(t)$= ($Pb_{i1}(t)$, $Pb_{i2}(t)$, ..., $Pb_{iN}(t)$). The second term is the best position of all particles in the swarm over iterations 1 ... $t$, which is termed global leader and it is represented as $Gb$. At the iteration $t+1$ of the search process, the velocity and the position will be updated according to the following equations:

$$\begin{aligned} V_i(t+1) = wV_i(t) + c_1 r_1 (Pb_i(t) - X_i(t)) \\ + c_2 r_2 (Gb(t) - X_i(t)) \end{aligned} \tag{4.1}$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{4.2}$$

where:
$w$ is the inertia weight;
$c_1$ and $c_2$ are two non-negative constants, referred to as cognitive and social factors, respectively;
$r_1$ and $r_2$ are uniform random numbers in [0, 1] that brings the stochastic state to the algorithm.
The pseudo code of this algorithm for minimizing a cost function $J$ is provided in Algorithm 1.

The original PSO algorithm is designed to solve a single-objective optimization for a continuous solution space. Therefore, there must be proposed the particle representation, the particle velocity and the particle movement so that they work properly with multi-objective optimization for the robot path planning problem.

## 4.3 Multi-Objective Optimization

The multi-objective optimization problem with $l$ parameters (decision variables) and $n$ objectives is formulated as follows:

**Algorithm 1 PSO pseudo-code**

```
Initialize population, parameters
While Termination criterion is unsatisfied
    For $i$=1 to Population Size
     Calculate particle velocity according to (4.1)
     Update particle position according to (4.2)
      If $J(X_i) < J(Pb_i)$
        $Pb_i = X_i$
        If $J(Pb_i) < J(Gb)$
          $Gb = Pb_i$
        End
      End
   End
End
```

**Minimum** $J(X) = [J_1(X), J_2(X), ....., J_n(X)]$
**subject to:**

$g_j(X) \leq 0$, $j$ = 1, 2, .., $m$
$h_k(X) = 0$, $k$ = 1, 2, .., $p$

where $X= [X_1, X_2, ... , X_l]$ is the vector of decision variables, $J_i$: $\mathcal{R}^n \rightarrow \mathcal{R}$ ($i$=1, 2, , ... , $n$) are the objective functions and $g_j$ ($j$=1, 2, , ... , $m$), $h_k$ ($k$ = 1, 2, .., $p$) are the inequal and equal constraint functions of the problem.

In order to solve a multi-objective optimization problem, there are several options such as weighted sum method, $\varepsilon$-constraint method and multi-objective Pareto optimal solutions. Those methods and their advantages/ disadvantages are introduced in the following subsections.

### 4.3.1 Weighted Sum Method

In this method, a set of objectives is scalarized into a single objective by adding each objective pre-multiplied by a user supplied weight. Then the optimization problem is formulated as:

**Minimize** $J(X) = \sum_{i=1}^{n} \omega_i \, J_i(X)$,
**subject to**

$g_j(X) \leq 0$, $j$ = 1, 2, .., $m$
$h_k(X) = 0$, $k$ = 1, 2, .., $p$

For a multi-objective optimization problem (MOP), the most common approach is to combine the optimization criteria into a single objective function often with weights for linear combinations of attribute values (called weighted sum method). This method is simple; however, it is problematic, as the final solution depends on weighting factors and it cannot find the solution in case of a non-convex objective space.

### 4.3.2 $\varepsilon$-Constraint Method

The objective of $\varepsilon$-constraint method is considering and restricting the rest of the objectives within user-specific values, and the optimization problem is formulated as:

**Minimize** $J_\alpha(X)$
**subject to**
$J_i(X) \leq \varepsilon_q$, i = 1, 2, ... , $n$ and $q \neq \alpha$
$g_j(X) \leq 0$, $j$ = 1, 2, .., $m$
$h_k(X) = 0$, $k$ = 1, 2, .., $p$
This method is applicable to either convex or non-convex problems. However, the disadvantage is that the $\varepsilon$ vector has to be chosen carefully between the minimum and maximum values of the individual objective function.

### 4.3.3 Multi-Objective Pareto Optimal Solutions

In contrast, the best trade-off solutions, inspired by biology behaviors (GA, PSO, ACO), called the Pareto optimal solutions (Pareto set), are the most powerful approach to solve the MOP. It is worth noting that the multi-objective optimization provides more than one solution. This set includes the solutions that no solution is better than the others with respect to all objective functions. There are some necessary definitions related to the Pareto optimal concept as follows.

Assume that $X^{(1)}$ and $X^{(2)}$ can have one of two possibilities: 1) one dominates the other 2) none of them dominates the other. The concept of dominate is introduced as following.

**Definition 6.1** $X^{(1)}$ is said to dominate the solution $X^{(2)}$, denoted as $X^{(1)} \prec X^{(2)}$, if and only if both conditions (1) and (2) are true:
(1) $J_i(X^{(1)}) \leq J_i(X^{(2)}) \, \forall$ i $\in \{1, 2\}$
(2) $J_i(X^{(1)}) < J_i(X^{(2)}) \, \forall$ i $\in \{1, 2\}$
It is intuitive that if the solution $X^{(1)}$ dominates another solution $X^{(2)}$, then the solution $X^{(1)}$ is better than $X^{(2)}$ in terms of multi-objective optimization. If any of the above conditions is violated, then the solution $X^{(1)}$ does not dominate the solution $X^{(2)}$. If $X^{(1)}$ dominates the solution $X^{(2)}$, then it is called the non-dominated solution within the set $\{X^{(1)}$ and $X^{(2)}\}$.

**Definition 6.2 (Pareto set)** feasible solution $X^* \in R_n$ of MOP is called a Pareto optimal solution, if and only if $X \nexists \in R_n$ such that $J_i$(X)$\prec J_i(X^*)$. The set of all the Pareto optimal solutions is called the Pareto set (PS), denoted as:

$PS = \{X^*) \in R_n, \nexists$ X $\in R_n, J_i$(X)$\prec J_i(X^*)\}$

**Definition 6.3 (Pareto front)** The image of the *PS* in the objective space is defined as *Pareto front (PF)*
$PF = \{J(X) \mid \in PS\}$
For instance, the Bi-Objective Pareto optimal solutions are presented in Figure 4.1.

## 4.4 The Hierarchical Robot Path Planning Approach

In this section, the problem statement and important concepts are briefly introduced in the first subsection. Then, the framework of the proposed hierarchical path planning approach is presented in the next one. Finally, the triangular decomposition method and Dijkstra's algorithm are also shortly described.

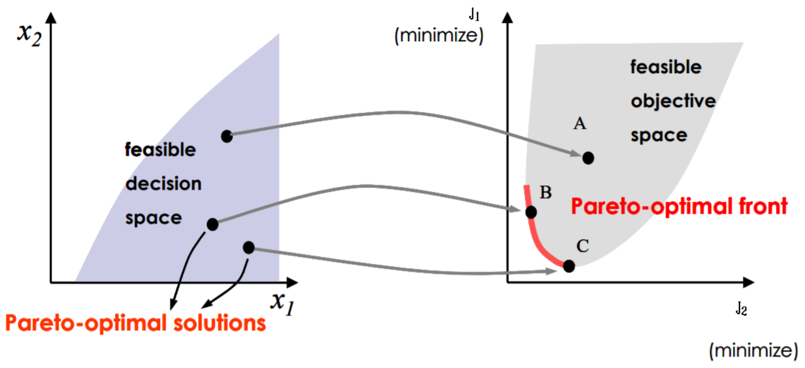


**Figure 4.1:** The graphical depiction of Pareto optimal solution. **Left:** Pareto optimal set of the entire feasible decision space; **Right**: Pareto optimal front defined by the set of all points mapped from the Pareto optimal set.

### 4.4.1 Problem Statement and Definitions

First, the problem of path planning investigated in this chapter can be stated as follows: considering a cluttered environment, a start position and a goal position; path planning is to find a collision-free optimal path with a sequence of points that is safe and feasible for the mobile robot to follow. Without loss of generality, the obstacles are assumed to have convex polygonal shapes. From the representation point of view, this assumption is not restrictive since any convex non-polygonal shape can be bounded by a convex polygonal region and a non-convex object can be divided into several convex objects.

**Definition 4.1 (Working space)**: a working space $\mathcal{C}$ is a physical space that is a sub-set of $\mathcal{R}^2$ for planar $2\mathcal{D}$ or $\mathcal{R}^3$ for $3\mathcal{D}$ spaces.

**Definition 4.2 (Obstacles configuration space)**: The obstacles are portions of $\mathcal{C}$ which are *occupied*, denoted by $O_1$, $O_2$, ...., $O_n$. The obstacles configuration space $\mathcal{CO}$ is the mapping of the obstacles in the working space with respect to the configuration space.

**Definition 4.3 (Free configuration space)**: The free configuration space $\mathcal{C}_{free}$ is a set of configurations in which the robot is free from collision with obstacles:

$$\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{CO} \tag{4.3}$$

### 4.4.2 The Framework of The Proposed Approach

In this subsection, the developed approach for the robot path planning problem is introduced. The key point is to develop multiple objective optimization based on the PSO algorithm in a hierarchical manner. There are three different levels in the frame work of the proposed approach presented in Figure 4.2. In the first level, the boundaries of every obstacles (in orange areas) are firstly expanded by an amount that is equal to the size of the robot plus a safety distance (in red areas) as shown in Figure 4.3. Therefore, the mobile robot can be considered as a point which freely moves out side those boundaries. The triangular decomposition method is applied to quickly find the free space. In the second level, Dijkstra's algorithm is used to generate the collision-free path from the start location to the goal location (dashed red path). In the last level, a CMOPSO with an accelerated update methodology based on the Pareto dominance principle is applied to obtain the optimal robot path planning in terms of solution quality and actual execution time (solid blue path). The following subsections present more details about each algorithms.

#### I. Triangular Decomposition

Cell decomposition is a partition of the free space into polygonal regions of the same type of geometry. The typical geometry are trapezoidal cells, triangular cells, polygonal cells, rectangular cells. In [140], the complexity and the quality of the path planning approach with respect to the chosen cell decomposition type are investigated. The authors suggest that the triangular decomposition algorithm is the most advantageous choice. This method has a low number of cells, robustness to noise, a low computation time and a large percentage of finding feasible paths. Thus, the triangular decomposition is very promising and it is applied in this work.

Basically, the triangular decomposition method has inputs $\mathcal{C}$ and $\mathcal{CO}$, while the outputs consist of a set of triangular cells $C = \{c_1, c_2, ...., c_m\}$ and the edges which correspond to adjacency among cells. Please refer to [50] for the definitions of the configuration space $\mathcal{C}$, the obstacles configuration space $\mathcal{CO}$, the free configuration space $\mathcal{C}_{free}$. The middle points of these adjacent edges are later used for Dijkstra's algorithm. Figure 4.3 is an example of the robot working space $\mathcal{C}$, 100 by 100 square meters which includes 4 obstacles. The START and STOP points respectively, denote the starting and the goal

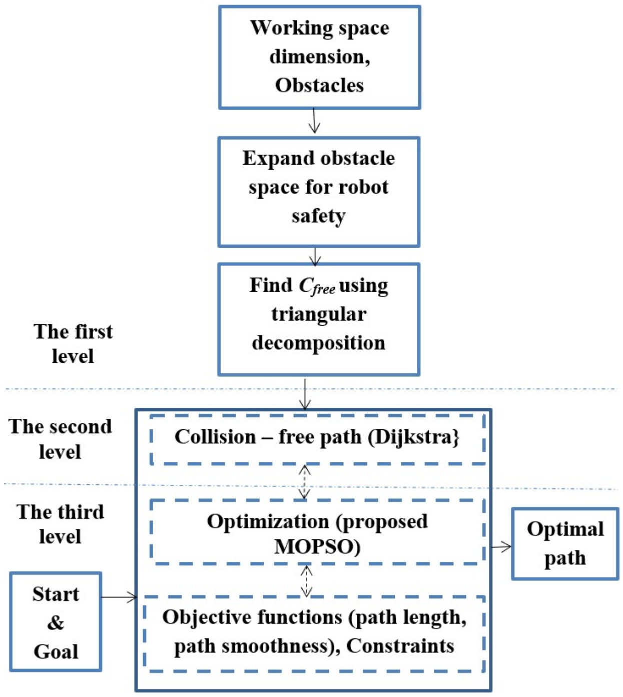

Working space dimension, Obstacles
Expand obstacle space for robot safety
Find $C_{free}$ using triangular decomposition
The first level
The second level
Collision – free path (Dijkstra}
The third level
Optimization (proposed MOPSO)
Optimal path
Start & Goal
Objective functions (path length, path smoothness), Constraints

**Figure 4.2:** The structure of the hierarchical robot path planning approach.

positions. The white areas are free configuration space $\mathcal{C}_{free}$ and the red ones are $\mathcal{CO}$. Obviously, the configuration free space includes a set of triangular cells:

$$\mathcal{C}_{free} = \{c_1, c_2, ...., c_m\} \tag{4.4}$$

## II. Dijkstra's Algorithm

Dijkstra's algorithm is an efficient algorithm used to search the shortest path in a graph. In order to solve the robot path planning problem, a graph is constructed from the triangular decomposition where each node corresponds to a cell. For more details about that algorithm, please refer to [156].

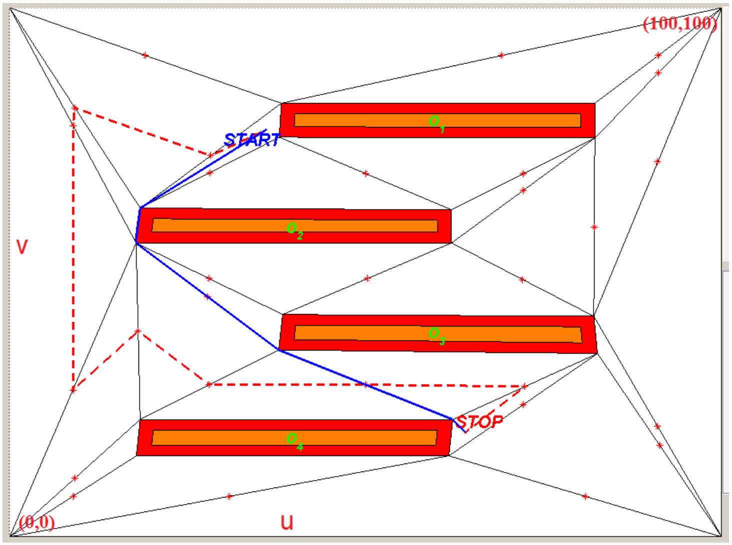


**Figure 4.3:** An example of the robot working space with four obstacles. The dashed-line denoted the pathway obtained by the triangular decomposition. The blue-line denotes the pathway obtained by our approach.

### III. Constrained Multi-Objective Particle Swam Optimization

The Evolutionary algorithms, in general, and the PSO, in particular, may find an optimal path by themselves; however, they have expensive computation. The aim of this step is to generate an optimal path based on the adjustment of the collision-free path points with the defined constraints by the proposed CMOPSO. More details on how the optimal path is generated will be given in the following sections.

## 4.5 Robot Path Planning Formulation

In this section, the specific knowledge of the robot path planning problem is used to create proper initial particles, constraints and objective functions. To achieve this purpose, appropriate initial particles are created based on the results obtained from the combination between the triangular decomposition algorithm and Dijkstra's algorithm. Furthermore, the modification of the original PSO is proposed to increase the speed of convergence, to evolve the robot constraints and to solve a multi-objective optimization problem. This section is divided into four subsections. The particles representation is firstly introduced. Then, the multi-objective path planning problem is defined, followed by the constraints problem. In the end, the problem formulation of the robot path planning is described.

### 4.5.1 The Particles Representation

Normally, the initial particles can be randomly created. This means that a number of points are arbitrarily selected from the robot working space. However, this strategy has little chance to obtain a reasonable path. When the environment becomes more complex, it will be more difficult to obtain a feasible one. Considering this problem, the combination of triangular decomposition and Dijkstra's algorithm is applied to ensure that all initial paths are free-collision ones.

As mentioned before, the triangular decomposition method is employed to archive $\mathcal{C}_{free}$ of the robot, then Dijkstra's algorithm is implemented to find a collision-free path. In the example shown in Figure 4.4, using Dijkstra's algorithm, the obtained collision-free path is $START \rightarrow P_1 \rightarrow P_2 \rightarrow ... \rightarrow P_8 \rightarrow STOP$, the robot moves in the environment with three obstacles. Since the robot moves in the $2\mathcal{D}$ environment, each point is represented in $u$ (horizontal axis) and $v$ (vertical axis) coordinates. For the path consisting of eight intermediate points, the corresponding particles representation is shown in Figure 4.4.

In general, the particle $X$ is defined as the coordinates of the intermediate point given by $u_i$ and $v_i$ $(i = 1...n)$, where the $n$ intermediate points are obtained by applying Dijkstra's algorithm as follows:

$$\begin{aligned} X_{2i-1} &= P_i(u) \\ X_{2i} &= P_i(v) \end{aligned} \tag{4.5}$$

Obviously, each intermediate point $P_i$ is the middle point of a line segment shared by adjacent cells. Let us assume that the relevant line segment has two

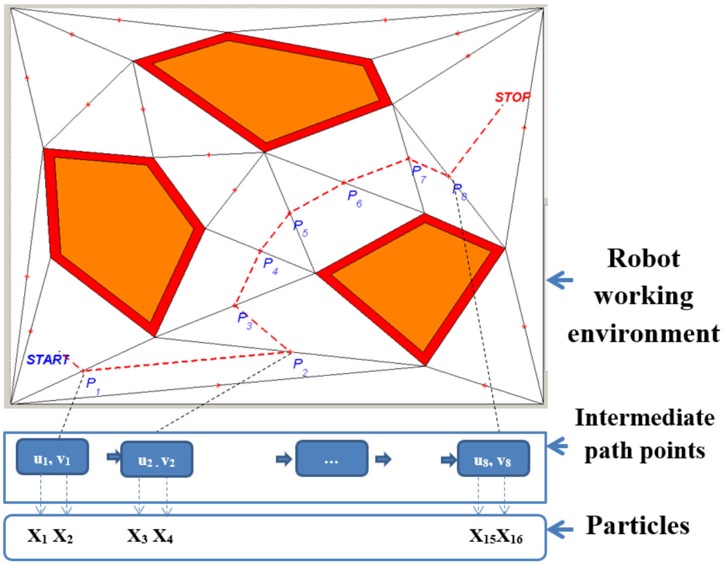


**Figure 4.4:** The solution representation of the robot path planning.

vertices termed $P_{i1}$ and $P_{i2}$ (for example $P_2$ is the middle point of the line with two vertexes termed $P_{21}$ and $P_{22}$ as shown in Figure 4.4). Thus, in general, for a path consisting of $d$ intermediate points, the robot path planning problem is transformed into the following optimization problem.

Find a set of points $S = \{P_1, P_2, ..., P_d\}$ together with the start point and the goal point to create an optimal path where $P_i$ ($i = 1 ... d$) satisfies the following coordinate constraints: $\min\{P_{i1}(u), P_{i2}(u)\} \leq P_i(u) \leq \max\{P_{i1}(u), P_{i2}(u)\}$ ($i = 1 ... d$) and $\min\{P_{i1}(v), P_{i2}(v)\} \leq P_i(v) \leq \max\{P_{i1}(v), P_{i2}(v)\}$ ($i = 1 ... d$). These constraints ensure that the robot is able to safely move in $\mathcal{C}_{free}$.

### 4.5.2 Multi-Objective Problem

The robot path planning is formulated as a multi-objective optimization problem with the aim of optimizing two objective functions while satisfying several inequality constraints. First, the objectives are formulated as follows.

***Minimization of the path length:***

Every path planning application must provide some degree of the shortest path. For the first performance criterion, the objective function is the total length of path, determined by:

$$\begin{aligned} J_1 = L(P) &= \sum_{i=0}^{d} L(P_i, P_{i+1}) \\ &= \sum_{i=0}^{d} \sqrt{(u_{i+1} - u_i)^2 + (v_{i+1} - v_i)^2} \end{aligned} \tag{4.6}$$

where $L(P_i, P_{i+1})$ represents the distance between two nodes $P_i$ and $P_{i+1}$; $u_i$, $v_i$ are the variable coordinates of node $P_i$ ($i$ = 0 ... $d$+1) in $\mathcal{C}_{free}$. In equation 4.6, $P_0$ ($START$) and $P_{d+1}$ ($STOP$) represent the start point and the goal point with chosen coordinates.
In the particles form, equation 4.6 can be written as:

$$\begin{aligned} J_1 = L(X) = &\sqrt{(X_1 - u_0)^2 + (X_2 - v_0)^2} \\ &+ \sum_{k=2i-1, i=1}^{i=d-1} \sqrt{(X_{k+2} - X_k)^2 + (X_{k+3} - X_{k+1})^2} \\ &+ \sqrt{(u_{d+1} - X_{2d-1})^2 + (v_{d+1} - X_{2d})^2} \end{aligned} \tag{4.7}$$

where:
$u_0$, $v_0$ are the coordinates of the start point.
$u_{d+1}$, $v_{d+1}$ are the coordinates of the target point. These values are known in advance.
$d$: the number of the intermediate points.

***Maximization of the path smoothness:***

The path smoothness is evaluated by summing the robot's turning angle in the desired path. The smoothness of a trajectory is a very important property in robot path planning, since the robot should not significantly change its direction. The path smoothness leads to less energy and time consumption. Henceforward, in this study, the smoothness is considered as the second objective function. The strategy which follows is not a general smoothing strategy; it is rather specific, assuming the following conditions:

- Assumption 1: the direction to leave from the starting point can be chosen freely.
- Assumption 2: the direction to arrive at the goal point can be chosen freely.
- Assumption 3: between the intermediate points, the mobile agent has to move in a straight line.

Taking into account these restrictions, it is obvious that maximizing the path smoothness is equal to minimizing the total of the robot's turning angle. The cost function of the robot's turning angle is defined as follows:

$$J_2 = \Theta(P) = \sum_{i=1}^{d} |\theta_i| \tag{4.8}$$

$$\theta_i = arctan\left(\frac{v_{i+1} - v_i}{u_{i+1} - u_i}\right) - arctan\left(\frac{v_i - v_{i-1}}{u_i - u_{i-1}}\right) \tag{4.9}$$

In a similar way, equation 4.9 can be written in the particles form as follows:

$$\begin{aligned} J_2 = \Theta(X) &= arctan\left(\frac{X_4 - X_2}{X_3 - X_1}\right) - arctan\left(\frac{X_2 - v_0}{X_1 - u_0}\right) \\ &+ \sum_{k=2i-1, i=1}^{i=d-2} arctan\left(\frac{X_{k+5} - X_{k+3}}{X_{k+4} - X_{k+2}}\right) - arctan\left(\frac{X_{k+3} - X_{k+1}}{X_{k+2} - X_k}\right) \\ &+ arctan\left(\frac{v_{d+1} - X_{2d}}{u_{d+1} - X_{2d-1}}\right) - arctan\left(\frac{X_{2d} - X_{2d-2}}{X_{2d-1} - X_{2d-3}}\right) \end{aligned} \tag{4.10}$$

### 4.5.3 Constraints Problem

This subsection is devoted to define the constraints problem for the robot path planning. There are two considered factors: the position constraints and the heading angle constraints of the robot.

#### ***The position constraints:***

The generated points $P_i$ with $i \in \{1, 2, ..., d\}$ are restricted by the coordinates of two points $P_{i1}$ and $P_{i2}$. Thus, the lower bounds and upper bounds are:

$$min\{P_{i1}(u), P_{i2}(u)\} \leq P_i(u) \leq max\{P_{i1}(u), P_{i2}(u)\} \tag{4.11}$$

$$min\{P_{i1}(v), P_{i2}(v)\} \leq P_i(v) \leq max\{P_{i1}(v), P_{i2}(v)\} \tag{4.12}$$

Based on the definition of the particles presented in previous subsection, the generated points $P_i$ can be expressed as:

$P_i(u) = X_{2i-1}$ and $P_i(v) = X_{2i}$.

Let's define:

$$\begin{aligned} X_{2i-1}^{LB} &= min\{P_{i1}(u), P_{i2}(u)\}; \quad X_{2i-1}^{UB} = max\{P_{i1}(u), P_{i2}(u)\} \\ X_{2i}^{LB} &= min\{P_{i1}(v), P_{i2}(v)\}; \quad X_{2i}^{UB} = max\{P_{i1}(v), P_{i2}(v)\} \end{aligned} \tag{4.13}$$

$LB_k$ and $UB_k$ are defined as lower and upper constraints. The general form of the position constraints problem is:

$$\begin{aligned} &LB_k \leq X_k \leq UB_k \\ &LB_k = (X_k^{LB}); \quad UB_k = (X_k^{UB}) \qquad \forall k \in \{1, 2, ..., N\} \end{aligned} \tag{4.14}$$

***The heading angle constraints:***

In the robot performance, it is difficult to rotate a large angle at once because of the physical system limitations. It is considered that the changing direction has a range of $[-90^\circ \quad 90^\circ]$. This can be stated as an inequality constraint:

$$\psi_i(X) = |\theta_i(X)| - 90^\circ \leq 0 \quad \forall i \in \{1, 2, ..., d\} \tag{4.15}$$

### 4.5.4 Problem Formulation

Aggregating the proposed objectives and constraints, the global robot path planning can be mathematically formulated as a constrained multi-objective optimization problem as follows:

$$Minimize \quad [J_1(X), J_2(X)] \tag{4.16}$$

Subject to:

$$\psi_i(X) \leq 0 \qquad \forall i \in \{1, 2, ..., d\} \tag{4.17}$$

$$LB \leq X \leq UB \tag{4.18}$$

In this problem, there are no equality constraints.

## 4.6 Constrained Multi-Objective Optimal Particle Swarm Optimization Algorithm

In order to solve the robot path planning problem described in section 4.5, a proposed CMOPSO algorithm is introduced in this section. First, the multi-objective optimization approach is presented. Second, the constrained method based on Pareto dominance principle is described, followed by the update of the archive, the global best and the particle position subsections. Please refer to the programming code in [157].

### 4.6.1 Multi-Objective Optimization Problem

For a multi-objective optimization problem (MOP), the most common approach is to combine the optimization criteria into a single objective function by linear combinations of attribute values (called weighted sum method). This method is simple; however, it is problematic, as the final solution depends on weighting factors.

Instead, the best trade-off solutions, called the Pareto optimal solutions or Pareto set, are the most powerful approach to solve the MOP. In the following subsection the constrained dominance to deal with the multi-constraints in MOP is introduced. This relationship is used to update the feasible archive, the global best and the particle position.

### 4.6.2 Constrained Pareto Dominance Principle

For the constrained multi-objective robot path planning optimization problem, the dominance relationship which includes not only the objective function values but also its violation degree constraints is taken into account. It is reasonable to use its constraints-violated function to evaluate the violation degree. It means that each time the defined constraints are not satisfied, the violation degree ($vd$) increases according to the following formulas:

$$vd = \frac{1}{d}\sum_{i=0}^{d-1} vd_j(X) \tag{4.19}$$

where:

$$vd_j(X) = \begin{cases} 0 & \text{if } \psi_i(X) \leq 0 \\ 1 & \text{if } \psi_i(X) > 0 \end{cases}$$

To handle efficiently the constraints, an effective scheme is applied in the proposed algorithm. The constrained dominance principle is defined as:
(1) The solution with a smaller constrained violation degree is chosen to dominate the other.
(2) The two solutions have the same values of constrained violation degree, the non-dominance principle is is applied based on the cost functions to choose the better solution.

### 4.6.3 Update the Archives and the Global Best Position

For the robot path planning problem, a particle represents a potential path which connects the start point to the goal point and avoids all the obstacles. Using the constrained Pareto dominance principle is a straightforward way to extend the basic PSO in order to handle multi-objective optimization problems. In this study, the update of the archives (with size $N_a$) is based on this principle. At each iteration, the best solutions found by the swarm (with size $N$) are compared on a peer-to-peer basis with elements from the archive, which is set to empty at the beginning of the search. If the archive is "empty", the current solution is directly stored. If the solution that the algorithm suggests to enter is dominated by another element in the archive, then it is automatically discarded; otherwise, such a solution is stored in the archive. In addition, if any of the elements in the archive are dominated by the new element, then such old elements are removed from the archive [158].

The global best position $(Gb)$ is the best solution obtained by all particles so far. In the proposed algorithm, the archive set is firstly found along the search process according to the above principle. Then, the global best position is chosen from the archive set by applying the roulette wheel selection.

### 4.6.4 Update the Particle Position

It is important to have a flexible search strategy based on the exploration ability and the exploitation ability. One approach is studied in [159], in which the parameters ($w$, $c_1$, $c_2$) change adaptively. In the search process of PSO, the search space will gradually decrease as the generation increases. In addition, a chaotic operator is applied to generate parameters ($r_1$, $r_2$). It is obvious that at the beginning of the search, the exploration ability is necessary to ensure that the algorithm can search in a large space. Thereafter, the exploitation ability is preferred to guarantee that the algorithm can search promise areas carefully and converge to the optimal solution. In the conventional PSO, each particle updates its position based on both the current global best position-*Gb* and the personal best position-*Pb* (or local best). The purpose of using the local best position is primarily to expand the diversity of the quality solutions; however, the diversity can be simply simulated by randomness at the beginning (for the exploration ability purpose), then reducing the randomness at each iteration (for the exploitation ability purpose). Therefore, in order to accelerate the convergence of the algorithm, it is possible to use the global best position only [160].

The update strategy of the particle position is based on the accelerated particle updates method. The purpose is an increasing convergence speed toward global non-dominated solutions. Based on that statement, the velocity and position vectors are formulated as:

$$V_i(t+1) = V_i(t) + c_1 + c_2(Gb(t) - X_i(t)) \tag{4.20}$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{4.21}$$

where:
$c_1$ is a random value in ($UB$-$LB$) that brings the stochastic state to the algorithm; UB and LB are defined as lower and upper constraint values;
$c_2 \in$ [0.1 0.7]; $Gb(t)$ is the global best in iteration $t$.
$V_i(t), V_i(t+1)$ are velocities of particle $i$ in iterations $t, t+1$.
$X_i(t), X_i(t+1)$ are positions of particle $i$ in iterations $t, t+1$.
To reduce the randomness as iterations are updated, the value of $c_1$ can be designed as:

$$c_1 = c_0 \gamma^t rand(UB - LB) \tag{4.22}$$

where $c_0 \in$ [0.1 0.5] is the initial value of the randomness parameter while $t$ is the number of the iterations and $\gamma \in$ (0 1) is a design parameter.

The purpose is to generate the solution by randomness around the middle of [LB UB] at the beginning (for the exploration ability purpose). As the number of iterations increase, the solution is near the optimal value, therefore, the effect of locality in the local best solution reduces. In other words, to reduce the randomness at each iteration for the exploitation ability purpose the term $\gamma^t$ is added in the formula of $c_1$ (t $\to \infty$, $c_1 \to 0$). The parameter $c_2$ effects to the speed of convergence this value is chosen.

## 4.6.5 Complexities of the Algorithms

The computational complexity of the proposed algorithm is an important aspect that should be carefully considered. The measure of an algorithm's complexity usually is the execution time, which can be estimated or predicted. Using quantity called "steps" it is able to make the time measurement independent to a specific computer. The total number of steps is normally expressed as a function of the input size, called complexity function $T(n)$, where $n$ is the input size. The bounds of running time of Dijkstra's algorithm can be characterized as a function of the number of the edges, denoted $|E|$ and the number of vertices, denoted $|V|$. The algorithm's running time is given by $O(|E|log(|V|))$ [161].

The number of computations required for a complete run of the PSO algorithm is the sum of the computations required to calculate the cost of a candidate solution (based on the current position of the particles) and the computations required to update each particle position and velocity. Both of these are directly proportional to the number of iterations. In the robot path planning, the problem size is the number of intermediate path points. When the pure PSO algorithm is applied, the problem size is a designed parameter, which depends on each scenario of the working environment. To generate a feasible path in cluttered environment, this parameter is needed to set big enough (i.e. $K$=2* $the\ number\ of\ obstacles$, as each intermediate point has two coordinates $u$ and $v$). The use of Pareto-based pure PSO has led to program run times in $O(KT_{max}MS^2)$, where $K$ is the number of intermediate points, $T_{max}$ is the number of iterations, $M$ is the number of objectives, and $S$ is the population size.

The proposed CMOPSO uses archives to store non-dominate solutions based on the Pareto dominance principle. For the robot path planning problem, this requires time $O(NMSN_a)$ to test a candidate solution, while insertions and deletions can be done in constant time [162], where $N$=2*$d$ and $d$ is the number of intermediate points, $N_a$ is the size of the archives. Obviously, the required time of the proposed method is much lower than pure PSO. Compared to the pure graph algorithm, it provides an optimal path in the robot path planning application.

## 4.7 Implementation of the Proposed Algorithm

In order to clarify the connection between the CMOPSO algorithm and the optimization problem, the proposed CMOPSO for tackling with the robot path planning is presented according to the following steps.

**Step 1 (Path planning optimization problem)**

*Step 1.1 (The particles representation)* Build the mathematical model of the robot path planning optimization (particle representation). Consider that the path has $d$ intermediate nodes, each node has two coordinates $u$ and $v$ and the archived particles representation is $X = \{X_1, X_2, ..., X_N\}$ with $N = 2d$.
*Step 1.2 (Multi-objective problem)* Establish the objective functions.
*Step 1.3 (Constraints problem)* Establish the constraint functions.
*Step 1.4 (Path planning optimization formulation)* Establish path planning optimization formulation.

**Step 2 (CMOPSO algorithm)** Implement the proposed algorithm to find the mathematical model of robot path planning optimization problem.

*Step 2.1 (Initialization)* Set the necessary parameters (size of swarm-$S$, the size of the archives-$N_a$, the maximal sampling time $T_{max}$). The time (iteration) counter is set to 0 and the initial values are $c_0 \in [0.1\ 0.5]$, $c_2 \in [0.1\ 0.7]$. To increase the optimal convergence, an initial swarm $X$ is randomly generated in $[LB\ UB]$. The pseudo code for this step is:

- For $i$ = 1 to $N$
- Initialize $V_i$ = 0
- Initialize $X_i$ in $[LB_i\ UB_i]$

*Step 2.2 (Evaluation)* Calculate the objective functions and the constrained violated degree of each particle.
*Step 2.3 (Update the archives)* Search for the non-dominated solutions and put them in the archives.
*Step 2.4 (Update the global best position)* The global best position is selected from the archives.
*Step 2.5 (Main loop)* **While** ($t<T_{max}$),
(where $T_{max}$: predefined maximum iterations)

1. Update designed parameter $c_1 = c_0{}^* \gamma^t$ where $\gamma$ is a designed parameter, chosen in (0 1).
2. Update the velocity and the position.
3. Calculate the objective functions and the constrained violated degree of each particle.
4. Update the archives and the global best position.
5. Update the iteration $t = t+1$.

**Step 3 (Display results)** Display the optimal robot path.

## 4.8 Test Case: Simulations and Analysis

In this section, the proposed algorithm is validated through several test cases, assuming that a robot is performing the respective mission in a 100x100 square meters working space. In each test, the obstacles are generated with different sizes and shapes and the robot has different start and destination points. The simulations have been implemented in Matlab R2015a and tested in Windows environment using an Intel core i3 CPU 2.27 GHz processor. In the following simulations, the proposed algorithm used a set of parameters as: the swarm size $S = 60$, the size of archive $N_a$ = 20; the maximum number of iterations $T_{max} = 60$, $c_0$ = 0.2; $c_2$ = 0.7; $\gamma$ = 0.97. These values are chosen based on experience.

### (1) Test case 1

This test case includes two obstacles where their positions are not in the connection between the start and goal points. The positions of the start and the goal are (9.1, 83.6) and (80.7, 13.8), respectively. Ideally, the proposed algorithm should be able to find the straight line between those points.

For this test case, Figure 4.5 presents two obtained paths by Dijkstra's algorithm (dashed red line), and CMOPSO (solid blue line). Table 4.1 shows the coordinate values of those corresponding paths. For this problem, there are four intermediate points which form particles denoted by $X_i$ ($i$ = 1 ...8). In other words, the dimension of decision variable is 8. The position constraints which ensure that the robot is able to move freely in the working space are found as:

$LB$ = [21.6, 37.8, 34.6, 31.7, 34.6, 31.7, 34.5, 18.1]

$UB$ =[48.8, 53.8, 48.8, 53.8, 72.3, 39.9, 72.3, 39.9]

The heading angle constraints are also formulated according to the previous subsection. Dijkstra's algorithm is used to find the best direction from the start position to the goal position, then the proposed CMOPSO is applied to generate the optimal path focused on minimizing the path length and the path smoothness. It can be seen that the obtained path is the optimal solution for test case 1.

**Table 4.1:** The obtained paths of the test case 1.

| Paths | The coordinates of the intermediate points $(u, v)$ |
|---|---|
| Dijkstra's | (35.2, 45.8)→(41.6, 42.8)→(53.4, 35.8)→(53.4, 28.9) |
| CMOPSO | (43.5, 50.0)→(48.0, 45.6)→(61.9, 32.1)→(64.0, 30.1) |

### (2) Test case 2

Figure 4.6 displays a cluttered space with 10 obstacles and shows two paths obtained by Dijkstra's algorithm (dashed red line) and CMOPSO (solid blue line) in the same approach as mentioned in test case 1. Dijkstra's algorithm finds the collision-free path from the start point to the target point via 8 intermediate points, thus the dimension of the decision variable is 16. The

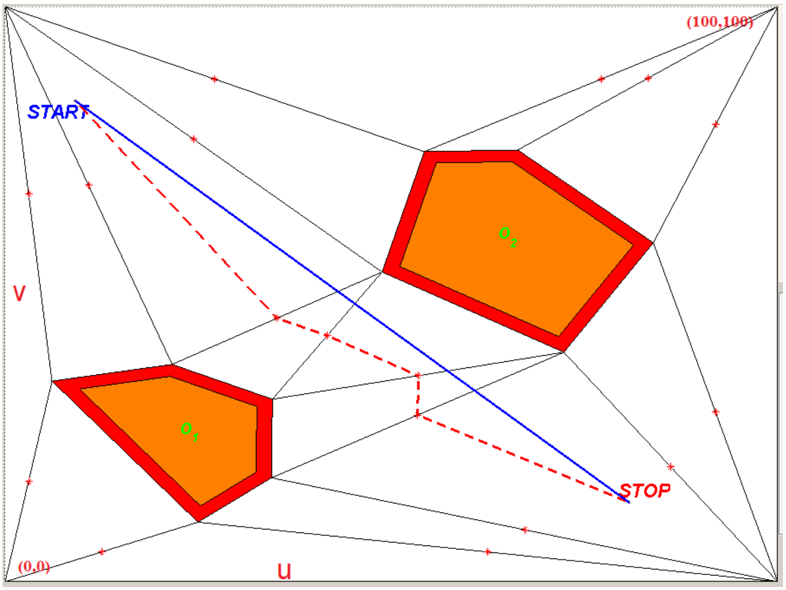


**Figure 4.5:** The obtained paths by Dijkstra's algorithm (dashed-red line) and CMOPSO (solid-blue line) in the test case 1.

start and end points are (4.5, 45.9) and (93.4, 43.3), respectively. The results are summarized in the Table 4.2 for the position constraints and in the Table 4.3 for the trajectories generated by Dijkstra's algorithm and CMOPSO.

**Table 4.2:** The robot position constraints of the test case 2

| Contraints | |
|---|---|
| *LB* | [ 0, 0, 0, 0, 0, 0, 0, 0, 74.4, 0 , 77.3, 0, 88.2, 0, 86.5, 0, 90.2, 0] |
| *UB* | [10.3, 64.3, 5.7, 11.5, 16.1, 7.5, 74.4, 5.6, 100, 5.6, 100, 5.4, 100, 8.4, 100, 35. 8, 100, 74.4] |

**(3) Test case 3** The test case includes 20 obstacles with different sizes and shapes. The start position of the robot is (7.2, 92.6). Simulations in this scenario are performed for three different locations of the targets $STOP1$= (50.4, 22.6); $STOP2$ = (85.9, 72.2); $STOP3$ = (9.5, 4.7). The dimensions of the decision variables are 30, 24 and 14 respectively. The results are summarized in the

**Table 4.3:** The obtained paths of the test case 2

| Paths | The coordinate of intermediate points $(u, v)$ |
|---:|:---:|
| Dijkstra's | (5.2, 32.1)→(2.9, 5.7)→(8.0, 3.8)→(37.2, 2.8)→(87.2, 2.9)→(88,7, 2.7)→ |
| (94.1, 4.2)→ | (93.2, 17.9)→(95.1, 23.7) |
| CMOPSO | (5.4, 13.8)→(5.7, 11.5)→(16.0, 7.3)→(52.5, 3.5)→(74.4, 4.3)→ |
| (79.7, 5.4)→ | (88.2, 8.4)→(89.0, 12.5)→(92.3, 17.8) |

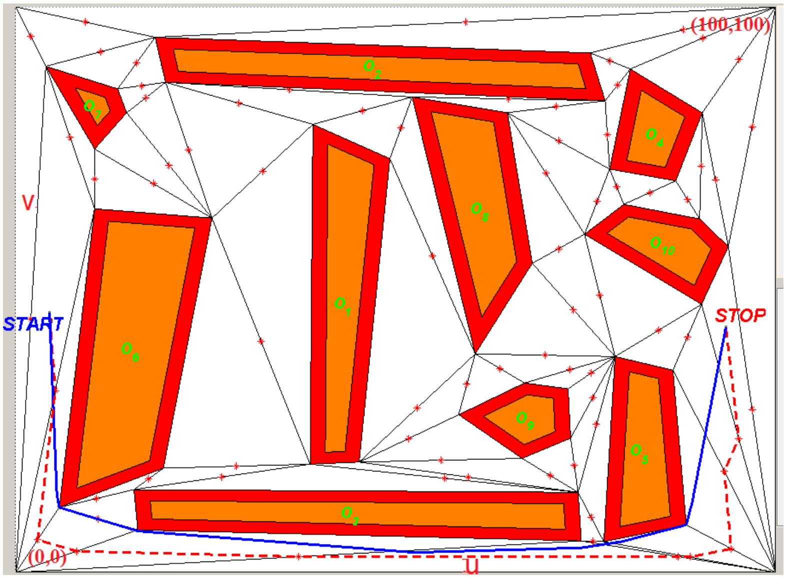


**Figure 4.6:** The multi-objective optimal paths obtained by Dijkstra's algorithm (dashed-red line), CMOPSO (solid-blue line) in the test case 2.

Table 4.5 for the position constraints and in the Table 4.4 for the trajectories generated by Dijkstra's algorithm and CMOPSO. The simulations are shown in Figure 4.7.

### (4) Test case 4

In the similar approach, the algorithm is extended in 3D space. The test case includes 12 obstacles. The two paths obtained by Dijkstra's algorithm (dashed

**Table 4.4:** The obtained paths of the test case 3

| Targets | Paths | The coordinate of intermediate points $(u, v)$ |
|---|---|---|
| $STOP1$ | Dijkstra's | (13.9, 85.1)→(14.3, 85.1)→(14.3, 80.6)→(17.3, 73.4)→ (19.5, 69.6)→(18.7, 66.8)→(18.7, 63.2) →(24.1, 60.8)→ (32.1, 55.9)→(36.6, 49.2 )→(38.9,46.5) →(41.7, 36.9) →(42.6, 35.3) →(45.6, 30.8) →(47.8, 26.5) |
| | CMOPSO | (11.2, 85.2)→(13.9, 79.9)→ (16.1, 75.6)→(19.5, 68.6)→(21.1, 65.3)→(23.1, 61.2))→ (24.3,59.2 →(28.1,53.8 )→(33.8, 45.8)→(36.3, 42.3)→ (38.4, 39.4) → (40.5,36.4)→(43.0, 32.8) → (44.6, 30.6) → (47.8, 26.2) |
| $STOP2$ | Dijkstra's | (9.4, 92,5)→(12.1, 94.1)→(21.5, 94.2)→(25.3, 94.9)→ (58.2, 92.5)→(60.3, 90.9)→(59.9, 86.8) →(63.1, 81.1)→ (69.3, 84.8)→(75.6, 84.7)→(79.2, 79.8)→(83.5, 71.7) |
| | CMOPSO | (14.3,93.2)→(20.2, 93.5)→(29.1, 94.1)→(37.9, 93.2)→ (50.6, 90.5)→(57.9, 88.1)→(61.6, 86.9) →(62.3, 86.7)→ (68.5, 84.6)→(74.8, 82.5)→(80.0, 77.7)→(83.3, 74.7) |
| $STOP3$ | Dijkstra's | (4.6, 92.6)→(2.9, 83.2)→(1.3, 71.1)→(1.3, 21.1)→ (1.9, 15.5)→(4.7, 12.3)→(8.1, 6.7) |
| | CMOPSO | (6.5, 85.8)→(5.0, 71.1)→(2.1, 43.5)→(2.0, 39.4)→ (3.0, 30.9)→(4.6, 24.7)→(7.4, 12.9) |

**Table 4.5:** The robot position constraints of the test case 3

| Targets | Types | The position constraints |
|---|---|---|
| $STOP1$ | $LB$ | [9.1, 84.9, 9.1, 76.1, 15.1, 70.7, 15.1, 68.5, 13.5, 65.2, 13.5, 57.9, 23.9, 53.1, 24.8, 53.1, 33.3, 39.7, 33.3, 39.7, 33.3, 38.2, 37.2, 35.8, 38.7, 32.3, 38.7, 29.1, 43.2, 23.8] |
| | $UB$ | [18.8, 85.2, 19.4, 85.1, 19.4 76.1, 23.9, 70.7, 23.9, 68.5, 23.9, 68.5, 24.4, 68.5, 39.9, 58.7, 39.8, 58.6, 44.5, 53.3, 46.3, 39.7, 46.4, 38.2, 46.4, 38.2, 52.4, 32.3, 52.4, 29.2] |
| $STOP2$ | $LB$ | [0, 84.9, 0, 88.2, 0, 88.3, 0, 89.8, 50.6, 89.7, 54.7, 86.6, 54.7, 86.6, 61.1, 75.2, 65.1, 82.5, 73.5, 82.5, 73.5, 81.9, 66.4] |
| | $UB$ | [18.7, 100, 24.2, 100, 43.1, 100, 50.7, 100, 65.9, 95.2, 65.8, 95.2, 65.2, 86.9, 65.1, 86.9, 73.5, 86.9, 77.6, 86.9, 84.9, 82.5, 84.9, 77.1] |
| $STOP3$ | $LB$ | [0, 85.2, 0, 66.4, 0, 42.1, 0, 0, 0, 0, 0, 0, 0, 0] |
| | $UB$ | [9.1, 100, 5.9, 100, 2.54, 100, 2.5, 42.2, 3.8, 30.9, 9.4, 24.8, 16.2, 13.3] |

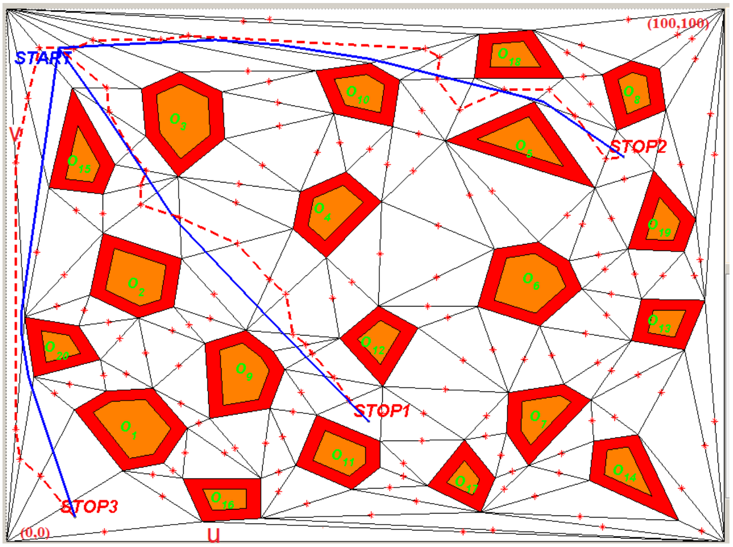


**Figure 4.7:** The multi-objective optimal paths obtained by Dijkstra's algorithm (dashed-red line), CMOPSO (solid-blue line) with the same start position and three different targetsin the test case 3.

red line) and CMOPSO (solid blue line) are shown in Figure 4.8. The start and end points are (10, 4.5, 3) and (0, 8, 2), respectively.

Figure 4.9 presents the approximated time consumption with different obstacles using CMOPSO. As it can be seen from this figure, the proposed approach generates optimal solutions in an extremely short period of time for environments with low, medium and high density (in terms of obstacles). The consumed computational time is less than 1 second for the environment with 20 obstacles.

Figures 4.5, 4.6, 4.7, 4.8 present the path planning results obtained by Dijkstra's algorithm and the proposed CMOPSO. The actual execution time reveals that CMOPSO is significantly fast in optimization, leading to a decrease in the computational burden. It is noticeable that CMOPSO converges quickly to the optimal value. Therefore, we can state that the proposed algorithm is reliable and satisfactory. The optimal path planning approach can be applied in many real-life robotic applications, such as autonomous car, surveillance operations, agricultural robots, planetary and space exploration missions.

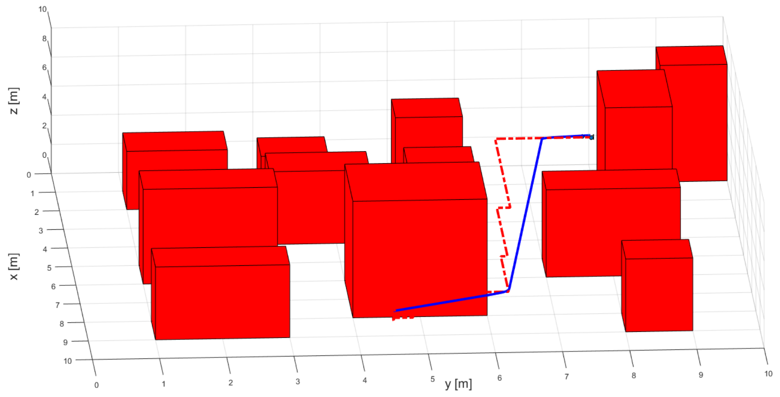


**Figure 4.8:** The multi-objective optimal paths obtained by Dijkstra's algorithm (dashed red line), CMOPSO (solid blue line) in 3D space.

## 4.9 Summary

In this chapter, a novel hierarchical approach for robot path planning in the presence of cluttered obstacles is proposed. It is a combination of the triangular decomposition method, Dijkstra's algorithm and a proposed constrained multi-objective particle swarm optimization (CMOPSO). The optimal algorithm includes an accelerate update methodology based on the constrained Pareto dominance principle.

The main achievements of this research are:

- The development of a novel hierarchical global path planning approach for mobile robots in cluttered environments;
- A proposed particle swarm optimization with an accelerated update methodology based on Pareto dominance principle for a constrained multi-objective path planning problem in term of minimizing the path length and maximizing the path smoothness considering the robot constraints;
- A proposed methodology with computational efficiency.

This proposed approach; however, is more suitable for environments where all the information about the obstacles and the targets is provided in advance. In

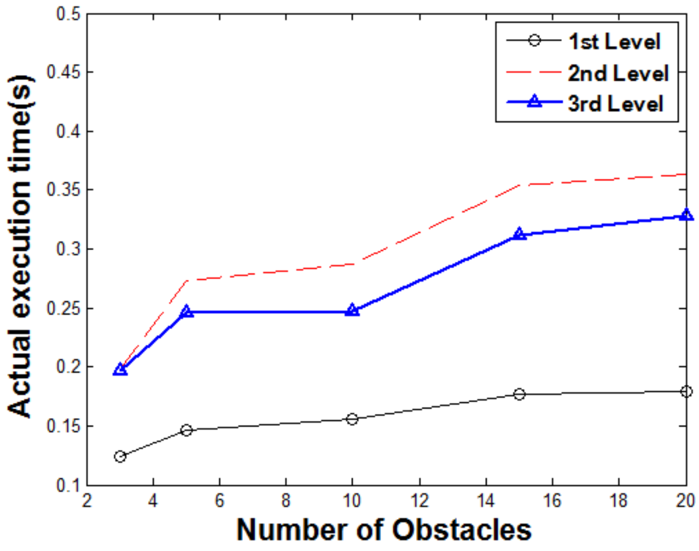


**Figure 4.9:** The actual execution time of CMOPSO's parts with different number of obstacles of thee levels (Level 1: triangular decomposition method; Level 2: Dijkstra's algorithm; Level 3: the proposed CMOPSO.)

other words, the main disavantage of the proposed path planning approach is the difficulty in adaptability with unpredictability. Hence, a development of a real-time robot path planning based on an improve potential field method in unknown/dynamic environments is presented in Chapter 5.

# Chapter 5

# Potential Field Method of an UAV's Autonomous Navigation

Fuzzy logic and Neural networks are intelligent approaches for mobile platforms; however, they have drawbacks, such as a time-consuming task (neural networks) and a difficulty in selecting the most suitable rules and membership functions (fuzzy logic). The nature-inspired algorithms can solve multi-objective robot path planning optimization problems. However, they are also time-consuming and not able to obtain a suitable path in some scenarios. The hierarchical approach in Chapter 4 is preferred for the environments where all information of the obstacles and the targets is provided in advance. For all these reasons, the robot path planning algorithms have been developed for unknown/dynamic environments and the potential field method (PFM) approach has been recognized as one of the most suitable candidates as it has a simple structure, a low computational complexity and an easy implementation.

In this chapter, a fully autonomous navigation solution of an Unmanned Aerial Vehicle (UAV) is presented. This solution includes the identification, localization, environment detection, online path planning and optimal control algorithms of the flying robot system. Novelty lies in: (*i*) the development of a position-estimation method using a sensor fusion technique in a structured environment. This localization method presents how to get the UAV's states (position and orientation), through a sensor fusion scheme, dealing with data provided by an optical sensor and an inertial measurement unit (IMU). Such a data fusion scheme also considers the time delay presented in the camera's signals due to the communication protocols; (*ii*) The improved potential field method which is capable of performing obstacle avoidance in an unknown environment and solving the non-reachable goal problem; and (*iii*) the design and implementation of an optimal Proportional-Integral-Derivative (PID) controller based on a novel multi-objective particle swarm optimization with an accelerated update methodology tracking such reference trajectories, thus characterizing a cascade controller. Experimental results validate the effectiveness of the proposed approach.

The results presented in this chapter are published in:

1. T. T. Mac, C. Copot, R. De Keyser, C. M. Ionescu, The Development of an Autonomous Navigation System with Optimal Control of an UAV in Partly Unknown Indoor Environment, *Mechatronics*, vol. 49, pp. 187-196, 2018. (A1, rank: Q1, impact factor: 2.496).
2. T. T. Mac, C. Copot, T. D. Tran, R. De Keyser, Ar.Drone UAV control parameters tuning based on particle swarm optimization algorithm, *IEEE international Conference on Automation, Quality and Testing, Robotics*, Cluj-Napoca, Romania, May 19-21, 2016. (P1-ISI).
3. T. T. Mac, C. Copot, A. Hernandez, R. De Keyser, Improved Potential Field Method for Unknown Obstacle Avoidance Using UAV in Indoor Environment, *IEEE 14th International Symposium on Applied Machine Intelligence and Informatics*, January 21-23, Herlany, Slovakia, pp. 345-350, 2016. (P1-ISI).
4. C. Copot, A. Hernandez, T. T. Mac, R. De Keyser, Collision-Free Path Planning in Indoor Environment Using a Quadrotor, *IEEE 21st International Conference on Methods and Models in Automation and Robotics*, Miedzyzdroje, Poland, 29th August-1st September, 2016. (P1-ISI).

## 5.1 Introduction

Over the last few years, Unmanned Aerial Vehicles (UAVs) have stirred up both scholar and commercial interest within the robotics community as the real and potential applications are numerous. To undertake the challenging task of autonomous navigation and maneuvering, a versatile flight control design is required.

A large number of studies have emerged in the literature on UAVs. Some examples of UAVs' applications can be found in precision agriculture [163], formation control of UGVs using an UAV [164], habitat mapping [165]. Modeling, identification and control of an UAV using on-board sensing are presented in [166]. The catch of a falling object using a single UAV has been accomplished in [167] and for a group of UAVs in cooperative formation in [168], where high-speed external cameras were applied to estimate the position of both the objects and the UAVs. The simultaneous localization and mapping (SLAM) was implemented to navigate an UAV in a working space [169]. The current implementations of UAVs still require collision avoidance, adaptive path-planing and optimal control algorithms. There exists a need to design methodologies to cope with these requirements to increase the degree of intelligence and, therefore, the autonomy of the UAVs.

Similar to a mobile robot, an autonomous UAV also consists of four essential requirements: i) *perception*, the UAV uses its sensors to extract meaningful information; ii) *localization*, the UAV determines its pose in the working space; iii) *cognition and path planning*, the UAV decides how to steer to achieve its target; iv) *motion control*, the UAV regulates its motion to accomplish the desired trajectory [170].

In recent literature, there has been a significant amount of work based on the PFM applied to ground agents' path plannings [171–174]. An interesting work on implementing and flight testing of this approach on an UAV is studied in [175]. To operate a real-time process, a layered approach is developed in an uncharted terrain. The global planner is based on the implementation of a Laplace equation that generates a potential function with an unique minimum at the target. The local planner uses a modification of the conventional potential field method in which not only the position of the UAV (as in the traditional PFM) but also the relative angles between the goal and the obstacles are taken into account. More than 700 successful obstacle avoidance experiments were performed. However, this approach sometimes encounters problems when the repulsion from the obstacles exceeded the UAV's physical constraints. It is pointed out that the potential field method has several inherent limitations [176] in which the non-reachable target problem is the most challenging and it is necessary investigating since it causes an incomplete path in the navigation task.

As the UAV is a complex system in which the electromechanical dynamics is involved [177], the robust controller is an essential requirement. In [178], the dynamical characteristics of a quadrotor are analyzed to design a controller which aims to regulate the posture (position and orientation) of the quadrotor. An autonomous control problem of an UAV in GPS-denied unknown environments is studied [179, 180]. In order to obtain reasonable dynamical performances, guarantee security and a sustainable utilization of equipment and plants, the controller has to be constantly optimal.

In this chapter, a real-time autonomous navigation of an Ar.Drone 2.0 in an indoor environment using only on-board visual and inertial sensors is proposed. The main contributions are the development of: (i) a position-estimation method based on a sensor fusion technique using only the on-board visual and inertial sensors considering the time delay of the camera's signals; (ii) a solution to solve the non-reachable target problem in conventional PFM; (iii) a multi-objective optimization for a PID controller based on a proposed multi-objective particle swarm optimization (MOPSO) with an accelerated update methodology to execute the navigation task. The motivation behind this research is to illustrate that a real-time autonomous navigation in unknown environments is feasible on low-cost UAV devices.

## 5.2 Path Planning Based on an Improved Potential Field Method

In this section, a conventional PFM is firstly presented, then the non-reachable goal problem of the PFM is described. Next, a proposed method, termed modified potential field method (MPFM), is introduced to overcome that problem. Lastly, the simulation results are described. This method ensures that the target is the unique global minimum of the total potential.

### 5.2.1 Conventional Potential Field Method and the Limitations.

In a conventional PFM, the agent is considered as a point mass in a working space. The obstacles exert repulsive forces while the target applies an attractive force on the agent. Conventionally, the attractive potential is defined as a function of the relative position between the agent and the target while the relative positions between the agent and the obstacles are used to calculate the repulsive potential. The sum of all the forces determines the subsequent direction and the travel speed. The attractive and the repulsive potential functions are:

$$\begin{aligned}
&U = U_{att} + U_{rep}\\
&U_{att} = \frac{1}{2}k_1 d^2(q, q_{goal})\\
&U_{rep} = \begin{cases} \frac{1}{2}k_2(\frac{1}{d(q,q_{obs})} - \frac{1}{d_0})^2 & \text{if } d(q,q_{obs}) \leq d_0 \\ 0 & \text{if } d(q,q_{obs}) > d_0 \end{cases}
\end{aligned} \tag{5.1}$$

where:
$k_1$ and $k_2$ are positive coefficients;
$q$, $q_{goal}$, $q_{obs}$ denote the positions of the robot, the target and the obstacle;
$d_0$ is the affected distance of obstacle;
$d(q, q_{goal})$ is the distance between the agent and the target;
$d(q, q_{obs})$ is the minimum distance between the agent and the obstacles.

The corresponding attraction force and repulsion force are negative gradients of respective attraction $U_{(att)}(q)$ and repulsion $U_{rep}(q)$ whose formulas are:

$$\begin{aligned}
&\overrightarrow{F} = \overrightarrow{F}_{att} + \overrightarrow{F}_{rep}\\
&\overrightarrow{F}_{att} = -k_1 d(q, q_{goal})\overrightarrow{n}_{RT}\\
&\overrightarrow{F}_{rep} = \begin{cases} -k_2(\frac{1}{d(q,q_{obs})} - \frac{1}{d_0})(\frac{1}{d(q,q_{obs})})^2\overrightarrow{n}_{RO} & \text{if } d(q,q_{obs}) \leq d_0 \\ 0 & \text{if } d(q,q_{obs}) > d_0 \end{cases}
\end{aligned} \tag{5.2}$$

where:
$\overrightarrow{n}_{RT}$ being the unit vector pointing from the robot to the target.
$\overrightarrow{n}_{RO}$ is the unit vector pointing from the robot to the obstacle.

Although the conventional PFM generates an effective path, it encounters several problems such as [176]:

- Non-reachable target.
- Oscillations in the presence of obstacles.

- Oscillations in narrow passages.
- No passage between closely spaced obstacles.

Among the above mentioned drawbacks, the non-reachable target problem is the most serious one since it causes an incomplete path. This occurs when the target is near the obstacles. In that case, when the agent approaches the target, it approaches the obstacles as well. As a consequence, the attractive force reduces while the repulsive force increases. Therefore, the agent is trapped in local minima and oscillations might occur.

Figure 5.1 (**Left**) presents a case where there are several obstacles located near the target. The repulsive force is considerably larger than the attractive force, therefore the agent is pushed away rather than reaching the target.

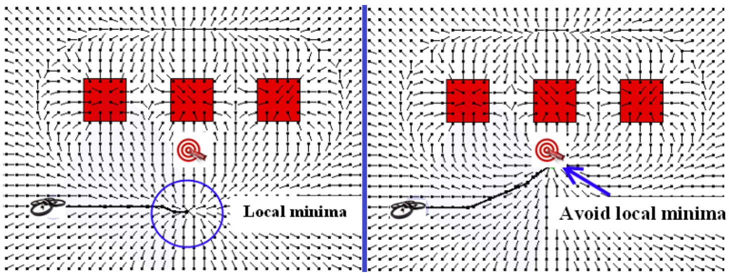


**Figure 5.1:** The problem and the solution when the position of the target is very close to the obstacles. **Left:** the target non-reach problem of the conventional PFM, **Right:** Avoid local minima solving the target non-reachable problem of the MPFM.

Figure 5.2 (**Left**) illustrates another case, in which the attractive field and the repulsive field are co-linear, in opposite directions and the total force approximates to zero thus the agent is trapped in a local minimum.

### 5.2.2 Proposed Attractive and Repulsive Potential

The crucial cause of the non-reachable target problem is that the goal position is not a global minimum of the total potential $U$ in (5.3). When the agent reaches the target, the attractive potential $U_{att}$ is equal to zero; however, the repulsive potential $U_{rep}$ is none-zero if there is at least one obstacle

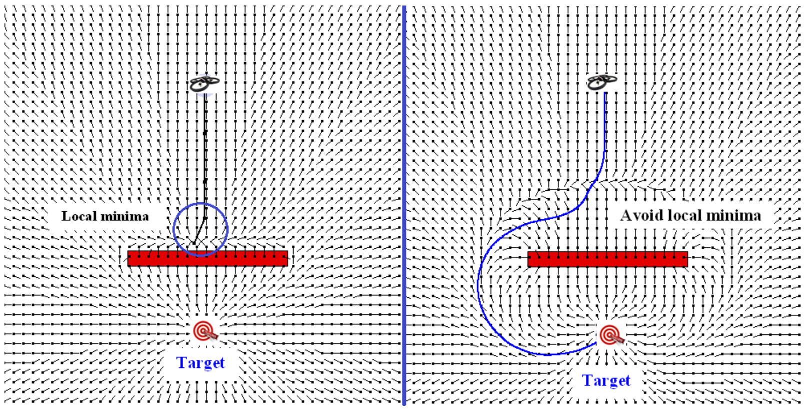


**Figure 5.2:** The problem and the solution when the agent, the obstacle and the target are aligned, the obstacle is in the middle and the attractive force approximates the repulsive force. **Left:** the target non-reach problem of the conventional PFM, **Right:** Avoid the local minima solving the target non-reachable problem of the MPFM.

which satisfies the condition $d(q,q_{obs}) < d_0$. To overcome that drawback, the proposed attractive and repulsive potentials are introduced to ensure that the total potential field force has the unique global minimum at the target position.

The MPFM are formulated as follows:

$$\begin{aligned}
&U = U_{att} + U_{rep} \\
&U_{att}(q,\dot{q}) = \rho_p d^2(q,q_{goal}) + \rho_v \|\dot{q}\|^2 \\
&U_{rep} = \begin{cases} \frac{1}{2}\alpha(\frac{1}{d(q,q_{obs})} - \frac{1}{d_0})^2 d^{\beta}(q,q_{goal}) & \text{if } d(q,q_{obs}) \leq d_0 \\ 0 & \text{if } d(q,q_{obs}) > d_0 \end{cases}
\end{aligned} \tag{5.3}$$

where $\rho_p$, $\rho_v$, $\alpha$, $\beta$ are positive coefficients; $d_0$ is the affected distance of obstacle; $d(q,q_{goal})$ is the distance between the agent and the target; $d(q,q_{obs})$ is the minimum distance between the agent and the obstacles.

The attractive force and repulsive force are the negative gradients of the attractive potential and repulsive potential as follows:

$$
\begin{aligned}
&\overrightarrow{F} = \overrightarrow{F}_{att} + \overrightarrow{F}_{rep} \\
&\overrightarrow{F}_{att} = -2\rho_p d(q, q_{goal})\overrightarrow{n}_{RT} - 2\rho_v \|\dot{q}\| \overrightarrow{v}_R \\
&\overrightarrow{F}_{rep} = \begin{cases} \overrightarrow{F}_{rep1} + \overrightarrow{F}_{rep2} & \text{if } d(q, q_{obs}) \leq d_0 \\ 0 & \text{if } d(q, q_{obs}) > d_0 \end{cases} \\
&\overrightarrow{F}_{rep1} = -\alpha\left(\frac{1}{d(q, q_{obs})} - \frac{1}{d_0}\right)\frac{d^{\beta}(q, q_{goal})}{d^2(q, q_{obs})}\overrightarrow{n}_{RO} \\
&\overrightarrow{F}_{rep2} = -\frac{\alpha\beta}{2}\left(\frac{1}{d(q, q_{obs})} - \frac{1}{d_0}\right)^2 d^{\beta-1}(q, q_{goal})\overrightarrow{n}_{RO}
\end{aligned}
\tag{5.4}
$$

where:
$\overrightarrow{n}_{RT}$ is the unit vector pointing from the robot to the target.
$\overrightarrow{n}_{RO}$ is the unit vector pointing from the robot to the obstacle.
$\overrightarrow{v}_R$ is the unit vector of the robot velocity.
$\|\dot{q}\|$ is the magnitude of the robot velocity. Applying the conventional PFM, the agent is not successful in autonomous navigation tasks when the obstacles are located near the target as mentioned before. However, the proposed method can properly handle such problems because it reduces the repulsive force when the agent moves towards the target. Thus, the agent is able to successfully reach the target, as shown in Figure 5.1 (**Right**) and Figure 5.2 (**Right**). It indicates that the proposed method effectively solves the non-reachable target problem when the obstacles are located near the target.

The proposed approach is developed to appropriately work in known and unknown complex environments. First, the global agent path planning is generated based on the proposed MPFM, then this path is re-planned when the agent senses a new obstacle until reaching the target (local path). The algorithm is presented in Figure 5.3.

### 5.2.3 Simulation Results in Complex Environment

To validate the proposed algorithm, simulations are executed under various complex environment conditions with known and unknown obstacles. There are large obstacles with different dimensions and shapes located in the way of the agent. Figure 5.4 illustrates two different cases of a complex indoor

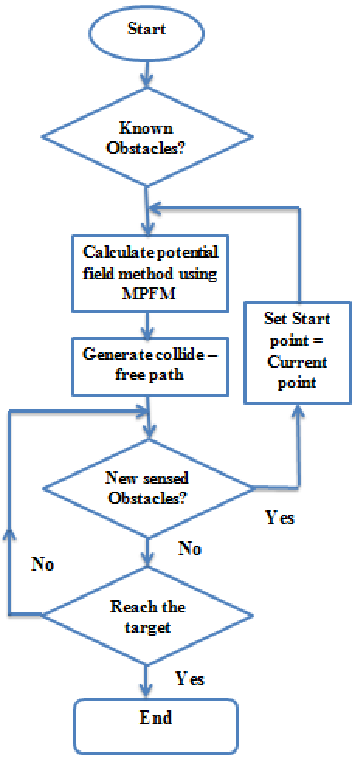

Start
Known Obstacles?
Calculate potential field method using MPFM
Generate collide – free path
Set Start point = Current point
New sensed Obstacles?
Yes
No
No
Reach the target
Yes
End

**Figure 5.3:** The flowchart of the agent's path planning in known and unknown environments based on the proposed MPFM.

environment with completely known obstacles in which the agent use the MPFM to arrive the target without colliding with the obstacles.

Figure 5.5 shows the result of the algorithm in complex scenarios with unknown obstacle. The known obstacle is presented in red, while the unknown obstacle is presented in black. Since the agent has no information about the black obstacle in advance, the trajectory is generated to avoid only the red obstacles, as displayed in Figure 5.5 (**Left**). However, it updates the path immediately (local path) as soon as it detects a new obstacle (the obstacle changes its color to orange) as shown in Figure 5.5 (**Right**).

Figure 5.6 presents the results of the algorithm in a dynamic environment. The moving obstacles have random movements. As soon as the obstacle is

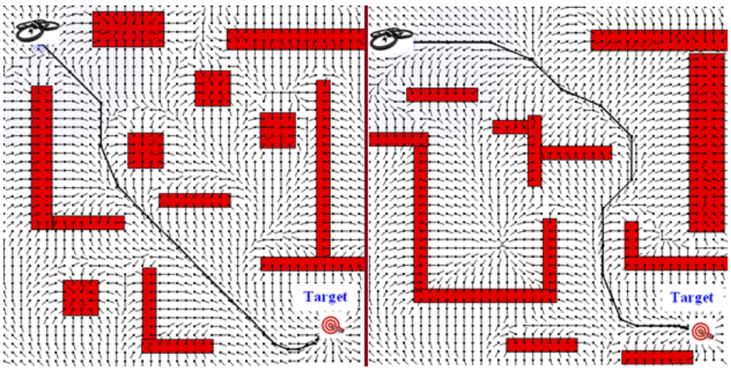


**Figure 5.4:** The proposed MPFM in the complex environments.

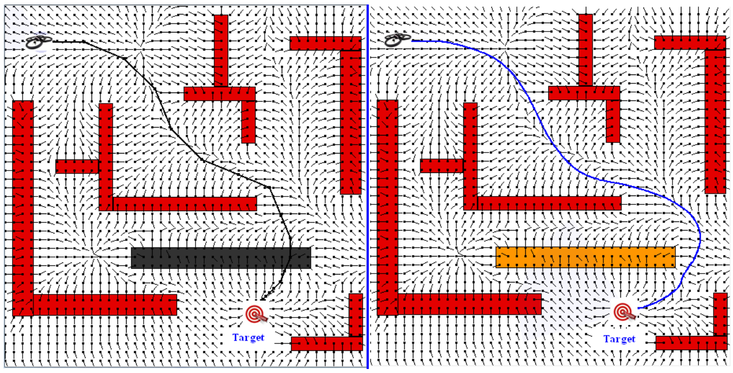


**Figure 5.5:** The proposed MPFM under the complex partial known environment, **Left:** The agent collides the black (unknown) obstacle. **Right:** The agent re-plans the path according to the perceived environment.

detected by the agent's sensor, its color is changed in yellow. In the fist step, the agent trajectory is a straight line, as there is no obstacle in the field of the agent (as shown in 5.6a). Then the agent's trajectory is updated based on the movements of the obstacles as shown in 5.6b, 5.6c, 5.6d, 5.6e, 5.6g. The results prove the feasibility and effectiveness of the algorithm in complex environments with known, unknown and dynamic obstacles.

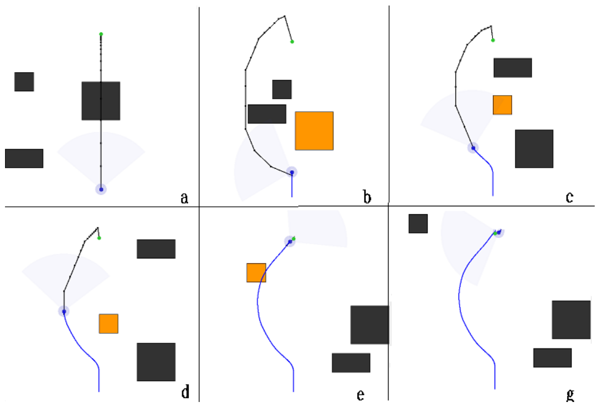


**Figure 5.6:** The proposed MPFM in a dynamic environment with three dynamic obstacles in time sequence. At each time two or three obstacles move simultaneously. The black obstacles are not sensed by the UAV. The yellow obstacle is sensed by the UAV. (a): the original path; (b)-(g): the re-planed paths by the propose MPFM to avoid the dynamic obstacles.

## 5.3 System Setup, Identification and Localization

An Ar.Drone 2.0, a commercial and low-cost micro UAV is used as the flying platform in this study. A description of its main characteristics, system identification, sensory equipment, system setup and localization are presented in this section.

### 5.3.1 Ar.Drone 2.0 Description and Coordinates System

Ar.Drone 2.0 has four propeller blades arranged symmetrically around a central unit which includes the sensory equipment and the circuit board. There are four basic motions of this UAV: pitch, roll, throttle, yaw and translational movements over $x$, $y$ and $z$, as shown in Figure 5.7 (**Left**). Notice that the translational movement on the $x$ axis is achieved by rotational movement along the transversal $y$ axis (pitch). A similar conclusion can be drawn for rotation over roll and translational movement on $y$.

It is worth mentioning that the coordinate system described above $(x; y; z)$, represents a relative coordinate system used by the internal controllers (low

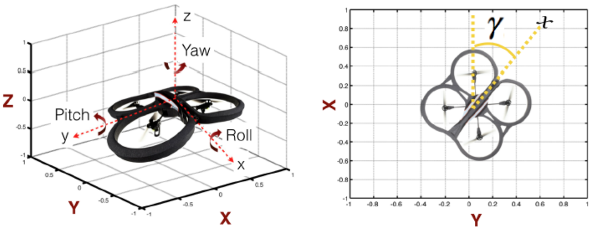


**Figure 5.7:** The movements of an Ar.Drone 2.0 in the absolute (**Left**) and the relative planes and the UAV's displacements on $(x;y)$ planes respect to the absolute plane (**Right**).

layer). Using such a coordinate system instead of absolute coordinates $(e.g., X;Y;Z)$ in the high layer will yield to errors. For example, notice that by rotating the quadrotor, the relative coordinates $(x;y)$ will change with respect to the absolute coordinates, as depicted in Figure 5.7 **(Right)**. In which, the angular rotation of the $XY$ coordinate systems with respect to the absolute $xy$ coordinate system is $\gamma$. It is possible to state that the relation between the two-coordinate systems depends directly on this angle. The sensor system consists of the following motion sensors, which together form the Inertial Measurement Unit (IMU).

- a three-axis accelerometer of 50 mg precision
- a three-axis gyroscope with $2000^0$/s precision
- a three-axis magnetometer of $6^0$ precision

**Ar.Drone Layers:** The Ar.Drone executes an operative system to read the sensors, to manipulate the speed of the motors, and also to autonomously control the quadrotor in four degrees of freedom. This on-board system is called the low-layer and is represented also by a black-box. The low-layer uses WiFi communication to connect with the high-layer, which has other control laws. Figure 5.8 describes the two layers present in the system.

The IMU provides the software with pitch, roll and yaw angle measurements. The communication between the Ar.Drone and the command station is performed via Wi-Fi connection within a 50 meters range. The Ar.Drone 2.0 is equipped with two cameras on the bottom and front, with the resolutions of

320 x 240 pixels at 30 frames per second (fps) and 640 x 360 pixels at 60 fps, respectively.

**Ar.Drone Communication:** There are four main communication services to connect with the AR.Drone.

First, the control is done by sending AT commands on the UDP port 5556. The transmission latency of the control commands is critical to the user experience.

Second, information about the drone (e.g. its status, position, speed, engine rotation speed, etc.), called *navdata*, is sent by the drone to its client on the UDP port 5554.

Third, a video stream is sent by the Ar.Drone to the client device on port 5555 with TCP protocol. Given this protocol has a confirmation step, it presents a video streaming with time delay of 360 ms approximately. The images from this video stream can be decoded using Software Development Kits (SDK). The embedded system uses a proprietary video stream format, based on a simplified version of the H.263 UVLC (Universal Variable Length Code) format.

Fourth, a communication channel, called "control port", is the TCP port 5559. It is used to retrieve configuration data, and to acknowledge important information such as the sending of configuration information.

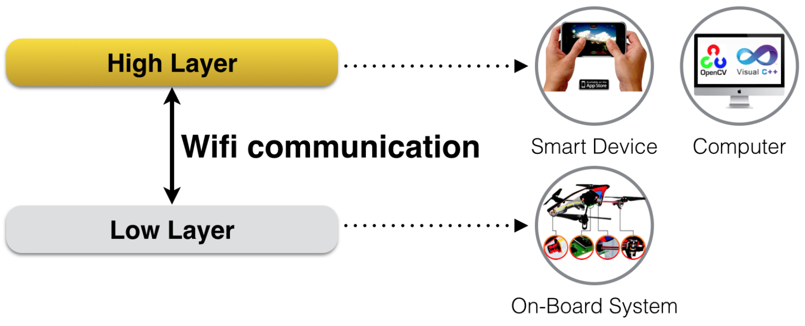


**Figure 5.8:** The quadrotor's layers: The low-layer represent the electronic assistance and the embedded operative system on the Ar.Drone; the high-layer represent the pilot.

### 5.3.2 Analysis of Inputs and Outputs and System Identification

The developed SDK mode allows the quadrotor to transmit and receive the information roll angle (rad), pitch angle (rad), the altitude (m), yaw angle (rad) and the linear velocities on longitudinal/transversal axes (m/s). They are denoted by $\{\theta_{out}, \phi_{out}, \zeta_{out}, \psi_{out}, \dot{x}, \dot{y}\}$ respectively. The system is executed by four inputs $\{V_{in}^{x}, V_{in}^{y}, \dot{\zeta}_{in}, \dot{\psi}_{in}\}$ which are the linear velocities on longitudinal/ transversal axes, vertical speed and yaw angular speed references as depicted in Figure 5.9.

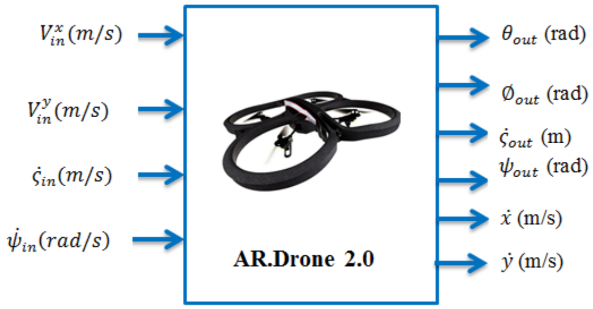


**Figure 5.9:** Inputs and outputs of an Ar.Drone 2.0.

The Ar.Drone is a multi-variable and naturally unstable system. However, due to the internal low layer control implemented in the embedded operative system, it is considered as a Linear Time Invariant (LTI) system, which is possible to be decomposed into multiple SISO loops. The transfer functions are obtained via parametric identification using the prediction error method and Pseudo-Random Binary Signal (PRBS) input signals [181]. Given the information obtained from the input steps experiments, a new signal was designed to excite the system in different frequencies using the *idinput* function of MATLAB. This time, the output is the integrated speed and the input is combination of a step signal, a *PRBS* signal and a chirp signal with mean equal to zero and frequencies between 0 $Hz$ and four times the inverse of the time constant from the previous experiment $\approx 3\,Hz$, see Figure 5.10. Sampling times of 5 ms for yaw and 66 ms for the other degrees of freedom are chosen based on the analysis of dynamics characteristic. The identified transfer functions are:

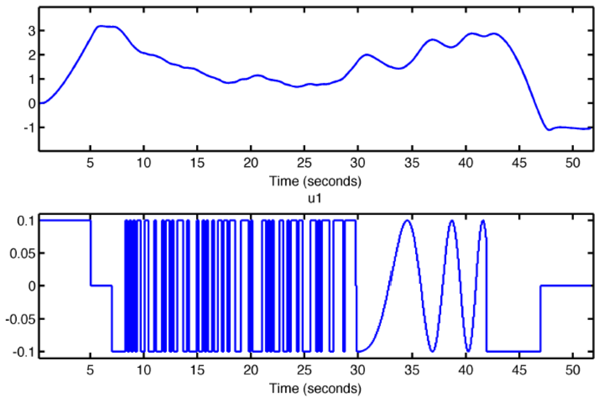


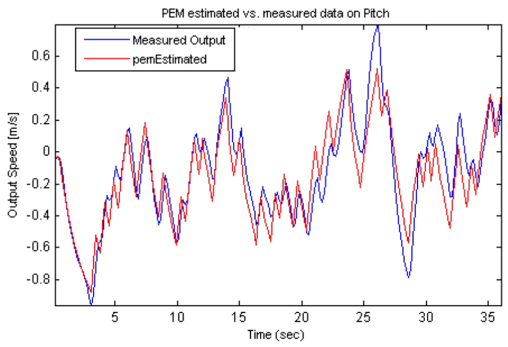
**Figure 5.10:** The applied signal (bellow) and the output (top) for translational movement in $y$ axis.

$$\begin{aligned} H_x(s) &= \frac{x(s)}{V_{in}^x(s)} = \frac{7.27}{s(1.05s+1)} \\ H_y(s) &= \frac{y(s)}{V_{in}^y(s)} = \frac{7.27}{s(1.05s+1)} \\ H_{altitude}(s) &= \frac{\zeta_{out}(s)}{\dot{\zeta}_{in}(s)} = \frac{0.72}{s(0.23s+1)} \\ H_{yaw}(s) &= \frac{\psi_{out}(s)}{\dot{\psi}_{in}(s)} = \frac{2.94}{s(0.031s+1)} \end{aligned} \tag{5.5}$$

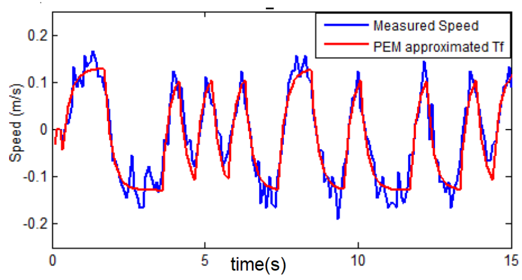
The validations of the pitch/roll, altitude and yaw's transfer functions are presented in Figure 5.11. They are made using a different set of data to prove that the quadrotor's movements are well approximated by the transfer functions.

## 5.4 Image Processing and Obstacle Detection

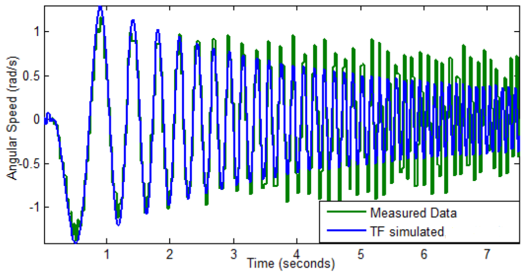
Interacting with the environment by sensing is an essential issue of an autonomous flight. There are several possibilities of obstacle detection on a quadrotor. Ultrasonic sensors are a good option for raw estimations of



**(a)** The comparison of the PEM-estimated the pitch/roll's transfer function response and the measured data (speed in m/s).



**(b)** The comparison of the PEM-estimated the altitude's transfer function response and the measured data (speed in m/s).



**(c)** The comparison of the PEM-estimated the yaw's transfer function response and the measured data (speed in m/s)

**Figure 5.11:** The validations of the pitch/roll, altitude, yaw transfer functions of the Ar.Drone 2.0 with different sampling time.

the nearby obstacles while long range laser sensors and depth cameras can give even more detailed information. The proposed method in this chapter considers both unknown and known obstacles, where the known obstacles are predefined from the beginning. The unknown obstacle information is obtained by image processing. This subsection presents the operations and techniques developed in OpenCV and C++ software for the autonomous vision applications. In this application, the size of a triangular pattern was used to estimate the distance between the quadrotor and an obstacle. Therefore, the image should be provided at a high quality.

### 5.4.1 Camera Calibration

The camera calibration is introduced as this process provides better image. Basically, OpenCV performs the calibration based on several patterns' images taken by the camera. For example, in theory, the chessboard pattern requires at least two snapshots. To find the correction parameters, some pictures of the chessboard were taken from different perspective. The algorithm is implemented to detect the corners and the intersections on the chessboard in order to calculate the parameters of the correction matrix. This experiment is necessary to be carried out once. This is open source process, please refer to [182]. The image obtained with and without distortion is presented in Figure 5.12.

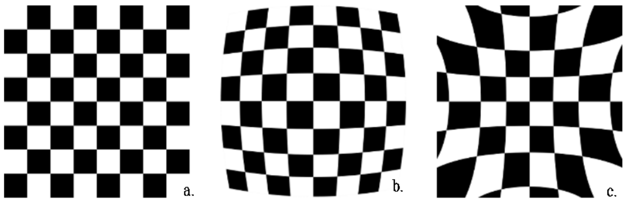


**Figure 5.12:** The image obtained with (a). No distortion; (b). Positive radial distortion; (c). Negative radial distortion.

### 5.4.2 Pattern Recognition

The object detection and segmentation are the most important and challenging fundamental tasks of computer vision. They are critical parts in many applications, such as image search and scene understanding. However, it is still an open problem due to the variety and complexity of the object classes and backgrounds.

Recently, there have been developed various methods and parameters which can be used to identify a pattern in a picture, e.g. color, shape, texture, histogram techniques. The color segmentation and shape segmentation are simple and fast techniques for object recognition.

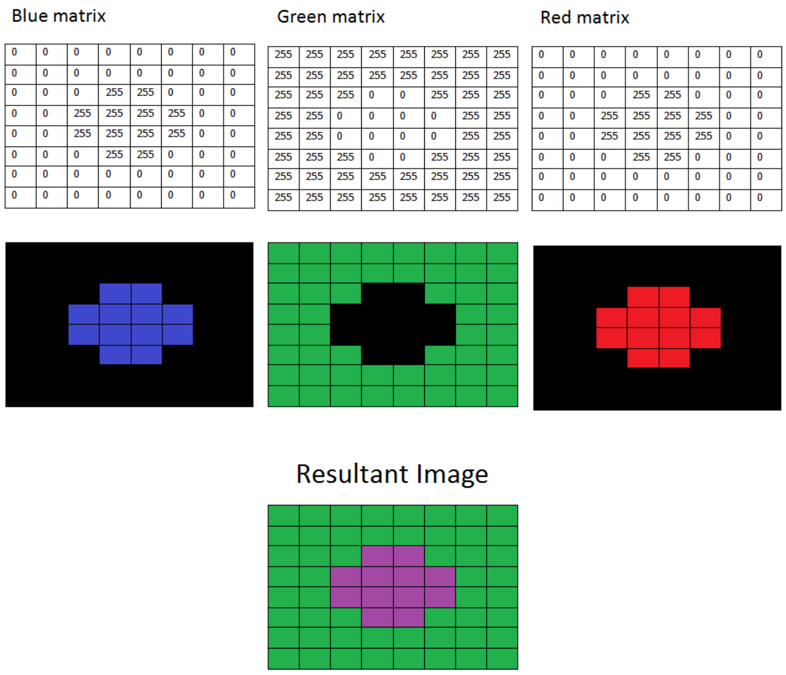
## Color Segmentation

The easiest way to detect and segment an object from an image is the color-based method. The object and the background should have a significant color difference in order to successfully segment the objects. The OpenCV usually captures images and videos in 8-bit, unsigned integer, BGR format. In other words, the captured images can be considered as three matrices; BLUE, GREEN and RED (hence the name BGR) with integer value ranges from 0 to 255. The Figure 5.13 shows how a color image is represented using the three matrices, in which each small box represents a pixel of the image.

The HSV color space also consists of three matrices, HUE, SATURATION and VALUE. In the OpenCV, value ranges for HUE, SATURATION and VALUE are respectively 0-179, 0-255 and 0-255. The HUE represents the color. The SATURATION represents the amount to which that respective color is mixed with white. The VALUE represents the amount to which that respective color is mixed with black. Here, the HUE is unique for a specific color distribution of an object. The SATURATION and VALUE may vary according to the lighting condition of that environment, as well as the surface of the object.

The Hue values of the basic colors:

- Orange 0-22
- Yellow 22- 38
- Green 38-75
- Blue 75-130
- Violet 130-160
- Red 160-179


Blue matrix

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 255 | 255 | 0 | 0 | 0 |
| 0 | 0 | 255 | 255 | 255 | 255 | 0 | 0 |
| 0 | 0 | 255 | 255 | 255 | 255 | 0 | 0 |
| 0 | 0 | 0 | 255 | 255 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Green matrix

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 0 | 0 | 255 | 255 | 255 |
| 255 | 255 | 0 | 0 | 0 | 0 | 255 | 255 |
| 255 | 255 | 0 | 0 | 0 | 0 | 255 | 255 |
| 255 | 255 | 255 | 0 | 0 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |

Red matrix

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 255 | 255 | 0 | 0 | 0 |
| 0 | 0 | 255 | 255 | 255 | 255 | 0 | 0 |
| 0 | 0 | 255 | 255 | 255 | 255 | 0 | 0 |
| 0 | 0 | 0 | 255 | 255 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Figure 5.13:** The image in BGR format.

The HSV color space is the most suitable for the color-based image segmentation. Therefore, in this application, the color space is converted from BGR to HSV.

### Shape Segmentation

The color recognition process is not a robust technique, as it depends on the background's color and the on light condition. Aside from the color, other information can be used to identify an object such as, the shape and the pattern. The shape is more stable with the changes of the light.

As a consequence, the final implemented method was designed as a combination of the color and the shape segmentation in order to obtain a more robust process. The implementation of this algorithm is presented as following steps (Figure.5.14).

1. The received image from the drone's camera is an RGB image.

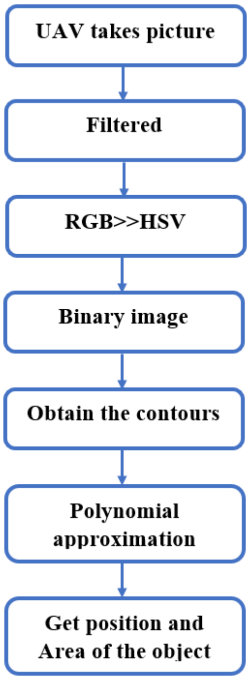


**Figure 5.14:** The pattern recognition algorithm.

2. The loaded image is filtered according to the application. A Gaussian filter is used to reduce the high frequency noise (e.g, the pepper noise and the salt noise).
3. The filtered image is converted from the default RGB space to the HSV space.
4. This Hue value is appropriately set with the object's color.
5. Choosing suitable threshold to obtain the binary image.
6. Obtain the contours of the objects.
7. Using polynomial approximation. For example, in this application, the interested object is detected as shown in Figure 5.15.
8. Calculate the object's position.

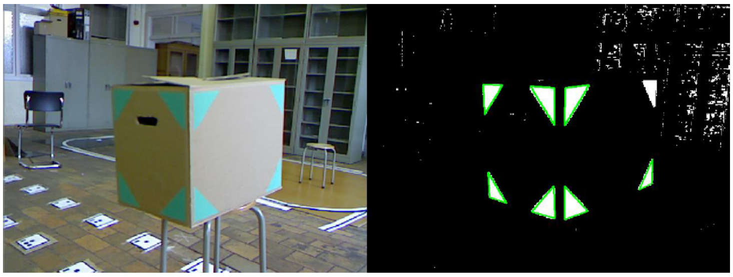


**Figure 5.15:** The detected obstacle by the Ar.Drone 2.0.

## 5.5 System Setup, Localization and Obstacle Avoidance

This section introduces the system setup, the localization method and the vision-based obstacle avoidance algorithm.

Our approach consists of three major components: a localization, an IMU data process and a cascade control as shown in Figure 5.16. The pattern-based localization allows the Ar.Drone 2.0 to determine its location and its orientation in a working space. The IMU data process delivers and receives signals between the quadrotor and the ground station. The last component is a cascade control, which guides the quadrotor to follow the designed trajectories.

The IMU chip delivers the velocities of each axis of the quadrotor, making it possible to obtain the position by integrating them. However, the small errors in the sensors and the numerical integration can lead to large errors in the position estimation over time. To overcome the limitations, an approach based on the ground patterns is proposed. This is a more accurate technique, but it also has some shortcomings (e.g., computational time). First, the velocity data of the drone in the local coordinate system is transformed into the world coordinate system, then the position is retrieved by using the Euler integration:

$$\begin{aligned} velocity_{world} &= velocity_{local} * \begin{bmatrix} \cos(yaw) & -\sin(yaw) \\ \sin(yaw) & \cos(yaw) \end{bmatrix} \\ pos_{world} &= prevpos_{world} + velocity_{world} \triangle time \end{aligned} \tag{5.6}$$

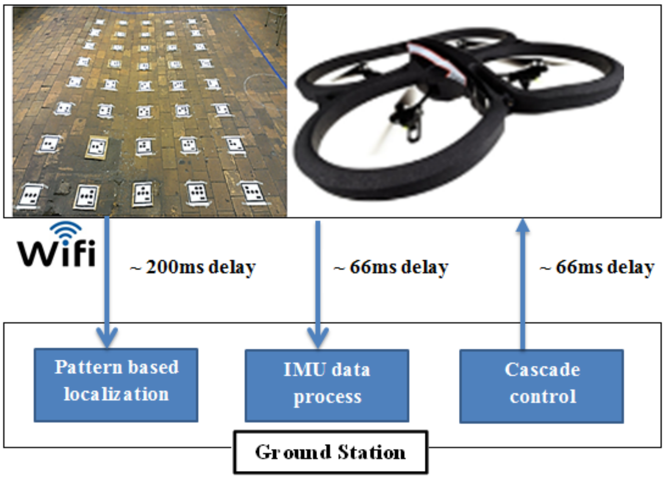


**Figure 5.16:** Our navigation approach of the Ar.Drone 2.0.

The cameras represent the main source of information for the system. Therefore, it is important to know the parameters which describe the relations between the pixels and the meters. These parameters were obtained by several experiments which take pictures from the cameras at different distances between the camera and a reference object with known characteristics (e.g., the area, the width, the height and the color). Figures 5.17 and 5.18 show the experimental data and the approximations obtained by frontal and bottom cameras.

The bottom camera uses a grid of ground patterns to estimate the pose of the drone. Each pattern inside this grid represents certain $(x, y)$ coordinates which are calculated based on the information from the first and the second rows. Each row includes three bits, the white and the black ones are corresponding to ’0’ and ’1’. Figure 5.19 represents the position of the ground pattern with the $x$ and the $y$ coordinates calculated as below:

$$\begin{aligned} x &= 2^0 x_0 + 2^1 x_1 + 2^2 x_2 \\ y &= 2^0 y_0 + 2^1 y_1 + 2^2 y_2 \end{aligned} \tag{5.7}$$

The rectangle at the right bottom of the pattern is used to approximate the UAV’s orientation. The distance between two patterns is 50 cm to ensure that there is always at least one pattern visible in one image.

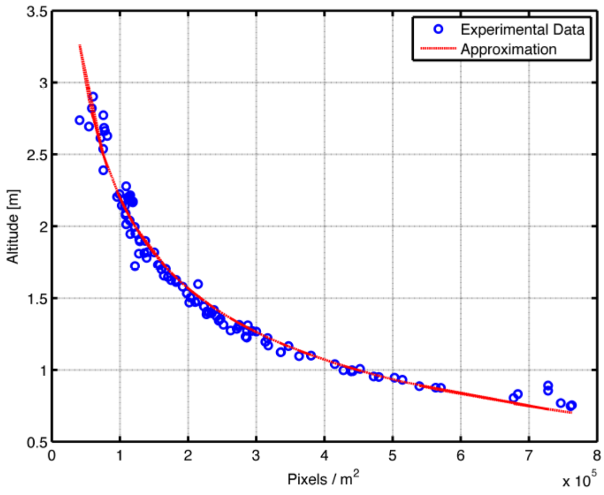


**Figure 5.17:** The experimental data and the approximations obtained for the bottom camera characterization.

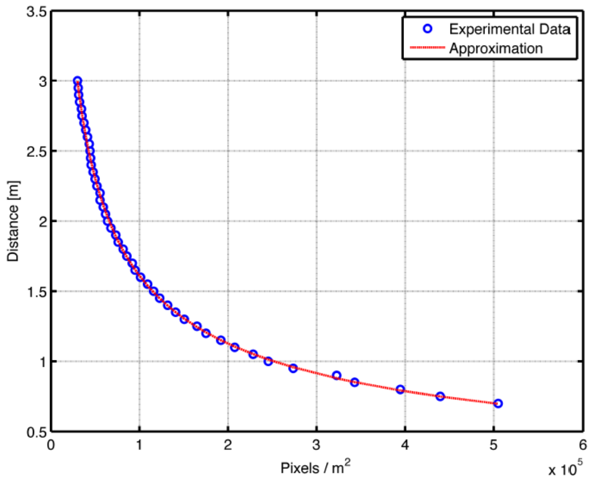


**Figure 5.18:** The experimental data and the approximations obtained for the frontal camera characterization.

Because the quadrotor has different pitch angles during the flight, the image center from the bottom camera is not always pointing perpendicularly to

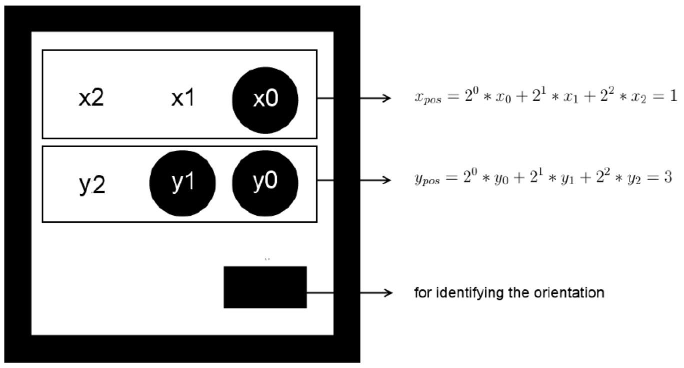

x2
x1
x0
y2
y1
y0
$x_{pos} = 2^0 * x_0 + 2^1 * x_1 + 2^2 * x_2 = 1$
$y_{pos} = 2^0 * y_0 + 2^1 * y_1 + 2^2 * y_2 = 3$
for identifying the orientation

**Figure 5.19:** A ground pattern representing position ($x = 1, y = 3$).

the ground (Figure 5.20(a)). Therefore, a correction of this center needs to be made using the information from the on-board pitch sensors. This offset is dependent on the pitch/roll angles of the quadrotor and the field of view (FOV) of the camera which can be calculated using trigonometry rules, as illustrated in Figure 5.20(b). In order to correct the offset for the $x$ and $y$ coordinates, the following relationships can be used:

$$\begin{aligned} offset_x &= \frac{2 * tan(roll) * tan(FOV_X/2)}{image_{width}} \\ offset_y &= \frac{2 * tan(pitch) * tan(FOV_X/2)}{image_{width}} \end{aligned} \tag{5.8}$$

The image processing algorithm is depicted in the flowchart from Figure 5.21. In which, the input image is converted into gray image, then a suitable threshold is applied to find the contours. This threshold depends on the light condition and therefore has to be appropriately chosen.

In this method, the localization is based on the sensor fusion approach which fuses data from the bottom camera and IMU, as shown in Figure 5.22. A time delay is presented in the camera signal due to the communication protocols. Time delay directly depends on the resolution of the camera, i.e: 0.33 seconds approximately for a size of 320x240 pixels. Figure 5.23 explains the information channels for cameras and IMU speed sensors. The sensor fusion localization allows Ar.Drone 2.0 to determine its location and its orientation in the working space.

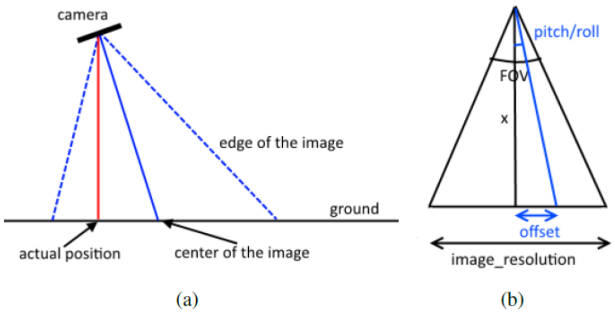


**Figure 5.20:** (a) The center of the image versus the actual position of the quadrotor; (b) Calculation of the offset to the image's center.

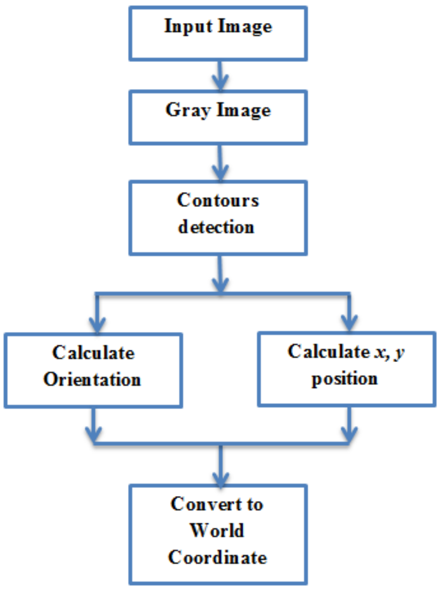


**Figure 5.21:** The image process flowchart of the pattern-based localization.

Figure 5.24 shows an estimation of the position with the odometer (IMU), the optical sensor (camera) and the combination which fuses two sensors. It is clear that it is not possible to only use the camera for position estimation as it starts to drift very quickly (because of the UAV's drift). The IMU has a robust position estimation but it is not very accurate. The sensor fusion combines the advantages of both signals. The fusion signal has an accurate estimation without noise.

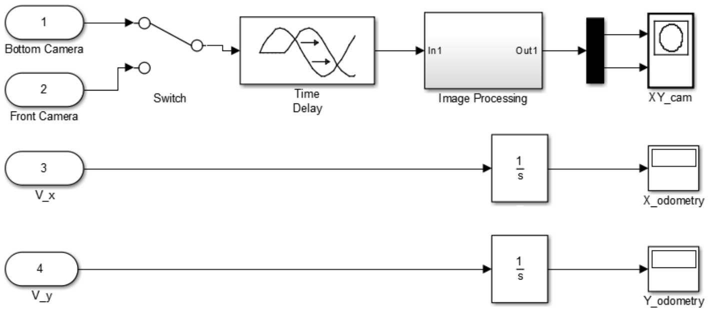


**Figure 5.22:** The sensors for the localization of the Ar.Drone in the $x, y$ planes.

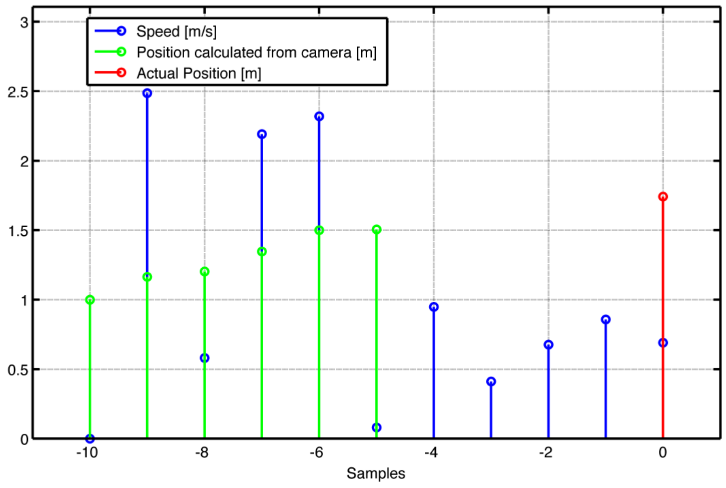


**Figure 5.23:** The combination approach to eliminate the time delay on the video signals. The green signals: the calculated position from the video signals with the time delay of 5 samples; the blue signals: the quadrotor's speed; and the red signals: the final calculated position.

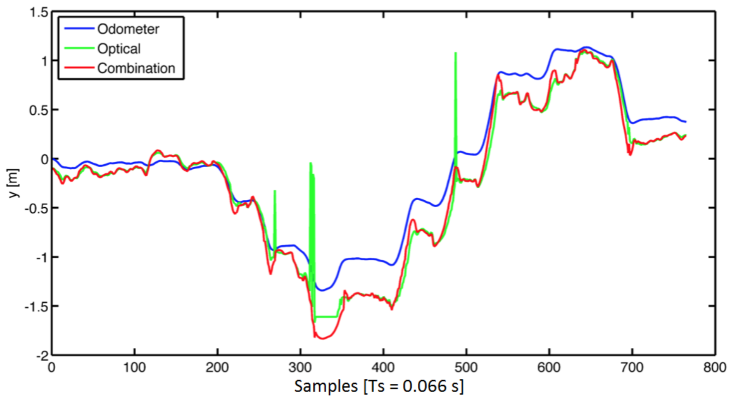


**Figure 5.24:** The position values in an open loop obtained from the image processing-optical sensor (green), the IMU-odometry (blue) and the fused response (red).

## 5.6 Control Parameters Optimization Based on Particle Swarm Optimization Algorithm

Unlike the existing approaches, in our work, the PID controllers of the UAV are designed and implemented based on the proposed multi-objective particle swarm optimization (MOPSO) with an accelerated update methodology which is presented in Chapter 4. This algorithm aims to facilitate the convergence to the optimal set of the PID parameters. This section is structured as follows. A MPSO-based PID controller approach is described. Then, a multi-objective optimization approach is presented. The final subsection presents the experiments' results.

### 5.6.1 PSO-based PID Type Controller Approach

Among the different control strategies available, PID controllers stand out and they are chosen in this autonomous flight for its simplicity and its easy implementation. Starting from the transfer function of a PID controller:

$$G_{PID}(s) = K_p + \frac{K_i}{s} + K_d s \tag{5.9}$$

The controller gains $K_p, K_i, K_d$ are chosen to satisfy prescribed performance criteria regarding the settling time ($T_s$), the rise time ($T_r$), the overshoot ($OS$) and the steady-state error ($SSE$). As the PID is a very well-known controller, the definitions of $T_r$, $T_s$, $OS$ and $SSE$ are not mentioned here. The proposed approach based on the MPSO techniques is applied to find the optimal values for the controller parameters that minimize the three desired objective functions such as:

$$\begin{aligned} J_1(X) &= |SSE| \\ J_2(X) &= OS \\ J_3(X) &= T_s - T_r \end{aligned} \tag{5.10}$$

Where $X$ is a set of parameters to be optimized, $X$ = $(K_p, K_i, K_d)$ with $K_p$>0, $K_i \geq 0$, $K_d \geq 0$.

The objective functions are established for the general case. For the case that the user wants the system to have no steady state error, then $K_i$ is set greater than zero ($K_i > 0$). Depend on the values of $K_p, K_i, K_d$, the types of controller can differ, such as P, PI, PD, PID. There are many factors which can be included in the objective functions, such as settling time $T_s$, rise time $T_r$ and control efforts.

The block diagram of the MPSO-based PID controller approach is presented in Figure 5.25. In this procedure, the dimension of the particle is 3. Initially, the MPSO algorithm assigns arbitrary values to $K_p, K_i, K_d$, computes the objective functions and continuously update the controller parameters until the objective functions are optimized.

### 5.6.2 Multi-Objective Optimization Approach

A composite objective optimization of the MPSO-based PID controller is obtained by summing the values of the three mentioned objective functions through the following weighted-sum method.

$$J(X) = \beta_1 J_1(X) + \beta_2 J_2(X) + \beta_3 J_3(X) \tag{5.11}$$

where $\beta_1$, $\beta_2$ and $\beta_3$ are positive constants;// $J_1(X)$, $J_2(X)$ and $J_3(X)$ are the objective functions.// In this application, those values are set as $\beta_1 = 0.35$, $\beta_2 = 0.5$ and $\beta_3 = 0.15$. These values are chosen based on expected response.

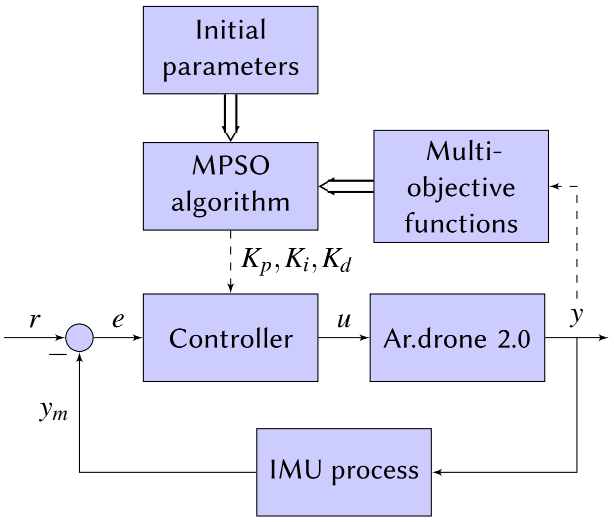


**Figure 5.25:** The MPSO-based PID controller approach.

**Table 5.1:** The optimal control parameters selected by the MPSO algorithm for the Ar.Drone 2.0's PID controllers.

| MPSO-PID | ALTITUDE PID | $X$ PID | $Y$ PID | YAW PID |
|---|---|---|---|---|
| $K_p$ | 7.3967 | 4.2315 | 4.2315 | 23.0011 |
| $K_i$ | 18.7427 | 2.5022 | 2.5022 | 41.4267 |
| $K_d$ | 2.8171 | 3.1780 | 3.1780 | 28.6480 |

The PID controller optimal parameters are obtained through simulations. It is worth noting that the models obtained for $X$ and $Y$ positions are symmetric, therefore their controllers have the same topology and parameters. Figure 5.26 presents the step response obtained for $X(Y)$ controllers using the proposed MOPSO, FRtool [183] and PID tuner MATLAB toolbox. The optimal $K_p$, $K_i$, $K_d$ parameter sets obtained by each Ar.Drone 2.0 controller are shown in Table 5.1. The proposed MOPSO is used to minimize the three cost functions in terms of the settling time/rise time ($T_s - T_r$), overshoot ($OS$) and steady-state error ($SSE$) for each controller.

The results of the proposed approach (the blue solid curves, MPSO-PID) are compared with the PID using FRtool (the red dash-dot curves) and the PID tuner MATLAB toolbox (the yellow 'x' marked curves). Both MPSO-

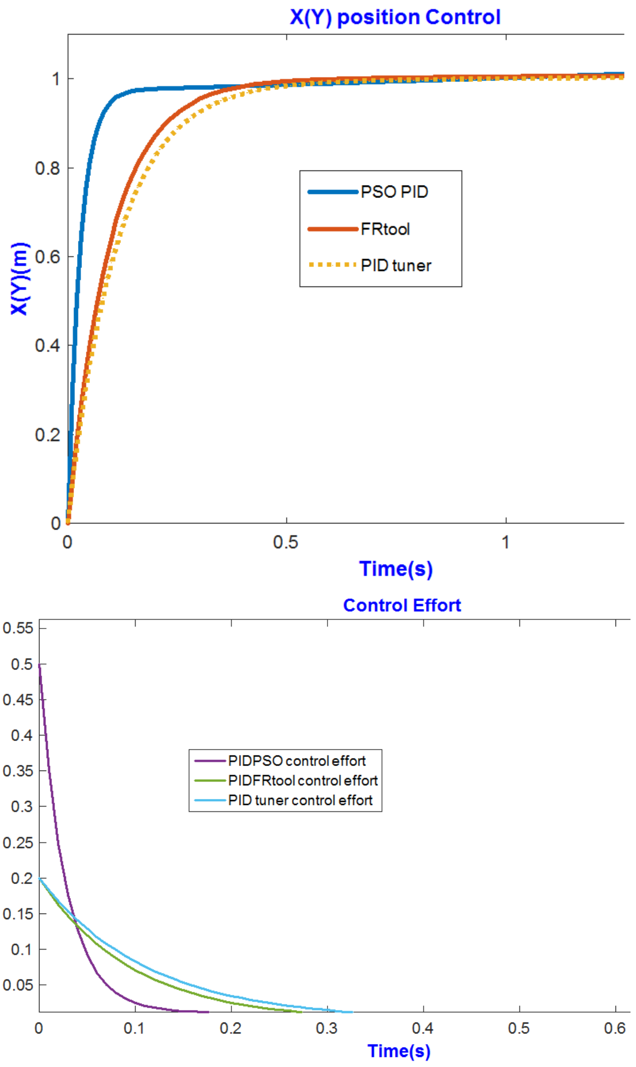


**Figure 5.26:** The $X(Y)$ position step response with the MPSO tuning, Frtool, PID tuner.

PID and PID-FRtool have better performances than the third one, with no overshoot. However, the settling time is clearly smaller for the proposed MPSO-PID controller than for the PID-FRtool and the PID tuner. Regarding the robustness and sensitivity of the approach, the parameters $\beta_1$, $\beta_2$ , $\beta_3$ are modified in the range of 20% of those values. The composite objective optimization is calculated as follows:

$$J(X) = (\beta_1 \pm \Delta\beta_1)J_1(X) + (\beta_2 \pm \Delta\beta_2)J_2(X) + (\beta_3 \pm \Delta\beta_3)J_3(X) \qquad (5.12)$$

where:
$\beta_1$, $\beta_2$, $\beta_3$ are defined in 4.3.
$\Delta\beta_1 \leq 0.2\beta_1$; $\Delta\beta_2 \leq 0.2\beta_2$; $\Delta\beta_3 \leq 0.2\beta_3$.

Figure 5.27 illustrates different MPSO-PID controllers for Ar.Drone 2.0's $X(Y)$ position obtained by randomly changing the weights of the three objective functions. It was observed that all MPSO-PID controllers react very fast and without overshoot and they track the reference input very well. In addition, there are only slightly small differences between the MPSO-PID controllers' outputs. In conclusion, the proposed approach is highly robust and not very sensitive in terms of chances in the weighted constants.

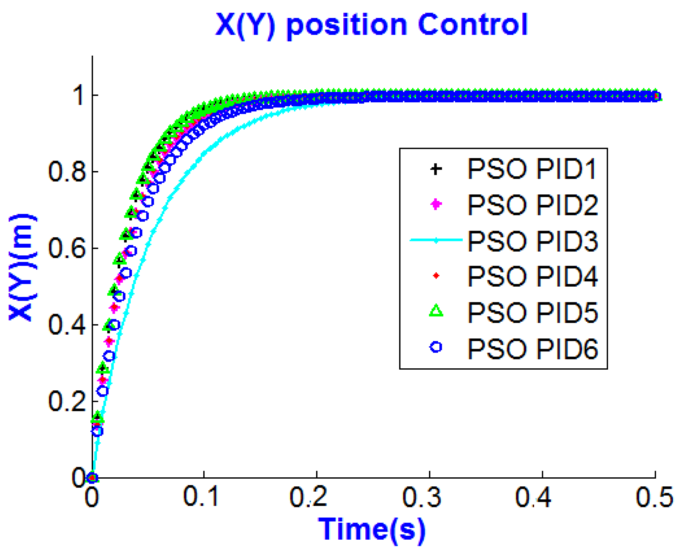


**Figure 5.27:** The simulation results obtained from MPSO-PID $X(Y)$ control in the variance of $\beta_1$, $\beta_2$ , $\beta_3$.

## 5.7 System Requirements and Experimental Validation

### 5.7.1 System Requirements

In order to run the AR.Drone 2.0's applications from a PC or a Laptop, the following items are required:

- One Parrot AR.Drone 2.0.
- Microsoft Windows operating system (Personally worked on Windows 7, Intel core i3 CPU 2.27 GHz processor). The usage and behaviors of the program with other operating systems have not been tested and are out of scope of this work.
- Visual Studio 2013 Professional.
- "cvdrone-master" library. This can be obtained by downloading it freely at https://github.com/puku0x/cvdrone.
- "openCv" library. This can be obtained by downloading it freely at https://opencv.org.

Additional software to be used for simulations:

- Matlab R2015a.
- FRTool.

### 5.7.2 Experimental Validation

The proposed solution structure of this study is depicted in Figure 5.28. At the first stage, the quadrotor uses its sensors to extract the obstacles' information, then it generates a path based on the proposed path planning algorithm. At this phase, the quadrotor should know its pose based on the localization process and it decides how to steer to achieve its goal. Thereafter, the drone regulates its motion to accomplish the desired trajectory. In real experiments, a cascade control is designed such that the Ar.Drone 2.0 accurately performs this task. There are two parts of the cascade controller: an inner-loop controller and an outer-loop controller. The inner-loop controller is performed inside the quadrotor as a black-box. The transfer functions are used to identify the relationships between the inputs and the outputs of this black-box. The outer-loop controller of the MOPSO-PID is represented by the command station, which defines the references to the internal controllers that are located in the low-layer.

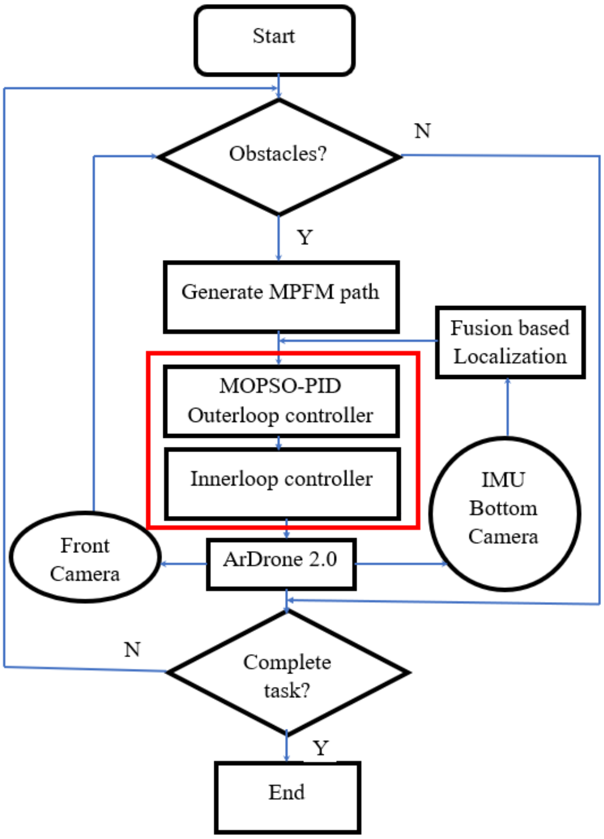


**Figure 5.28:** The proposed solution structure with a cascade control.

In the outer-loop controller, the localization process provides the current Ar.Drone 2.0's pose in the world coordinate based on the sensor fusion technique. The obtained free-collision trajectory using the MPFM is sent to the controller by a list of way-points. However, due to a missing communication, oscillation or uncertain noise, the Ar.Drone 2.0 may be unable follow the designed trajectory. Therefore, it is necessary to generate a compensation path, named $\overrightarrow{Tar}$, that guides the quadrotor to return to the designed trajectory.

To perform the compensation, several definitions are introduced. Suppose that the drone is currently in the position with the two nearest way-points, named *Previous way-point* and *Next way-point*.

- *Path error*: the distance between the drone and the designed path at a time during flight.
- $\overrightarrow{Target}_{WP}$: the vector from *Previous way-point* to the *Next way-point*.
- $\overrightarrow{Target}_{pathline}$: the vector towards the perpendicular path.

A schematic representation is depicted in Figure 5.29. The compensation $\overrightarrow{Tar}$ is only executed when the *Path error* ($L$) is larger than threshold value $L_0$.

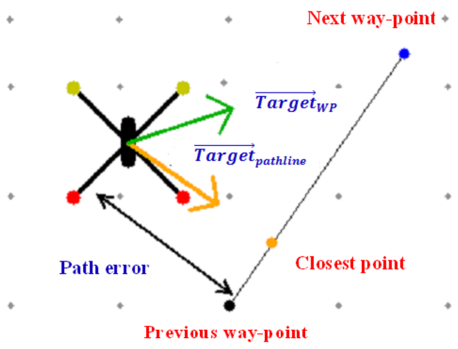

Next way-point
$\overrightarrow{Target}_{WP}$
$\overrightarrow{Target}_{pathline}$
Path error
Closest point
Previous way-point

**Figure 5.29:** A schematic of the proposed compensation. See the text for explanation.

The formula of the target vector is:

$$\overrightarrow{Tar} = \lambda * \overrightarrow{Target}_{WP} + (1-\lambda) * \overrightarrow{Target}_{pathline} \tag{5.13}$$

where $\lambda$ is defined as following:

$$\lambda = \begin{cases} 1-(L/L_0)^n & \text{if } L < L_0 \\ 0 & \text{if } L > L_0 \end{cases}$$

In order the drone to follow the designed trajectory, the $\overrightarrow{Tar}$ should be robust for a small *Path error* and should quickly adapt for a large *Path error*. For our system, $n = 4$ and $L_0 = 0.5\ m$ have been chosen.

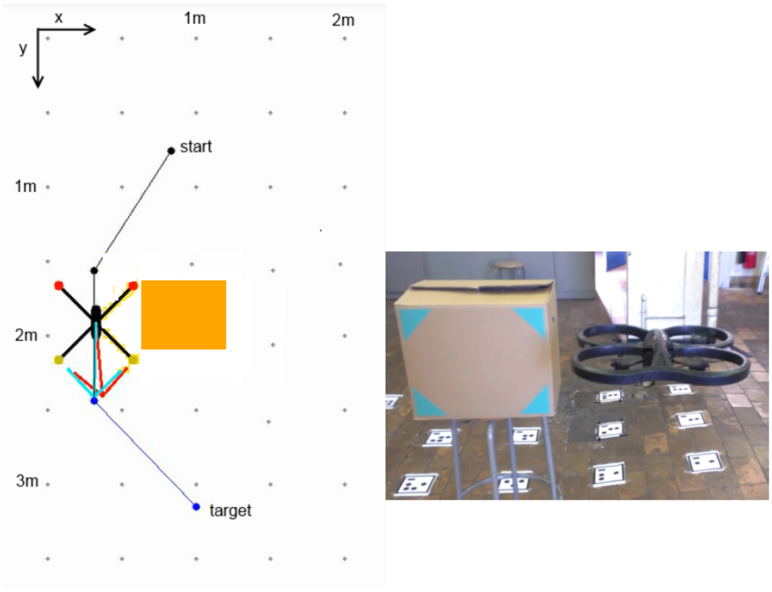


**Figure 5.30:** The virtual vs. the actual environment in the real-time experiment.

In this work, a virtual environment is developed to have on-line monitor and to facilitate off-line tests of the quadrotor navigation tasks, as presented in Figure 5.30. In this figure, the comparison between the virtual environment (**Lelf**) and actual environment (**Right**) is shown. When the drone detects the real obstacle in the actual environment, it is displayed in the virtual environment. The MPFM generates the path taking into account the ArDrone 2.0's dimensions. The start point, the target and the dimensions of the working space are also shown in the virtual environment.

The results obtained with the pattern-based localization, the MPFM and the optimal PID control in the real system are presented in Figure 5.31. The bounded rectangle presents the walls of the indoor working environment. The red obstacles are known obstacles and the yellow one is an unknown obstacle. In the beginning, the MPFM generates the black trajectory which collides with the unknown (yellow) obstacle, since the drone only avoids the red obstacle. However, it updates the path immediately (the local path-blue path) as soon as a new (yellow) obstacle is detected, as shown in Figure 5.31. The green path is the real path obtained in this experiment. It is possible to observe that the drone has few deviations from the desired trajectory, and an acceptable overshoot when sharp bends are performed.

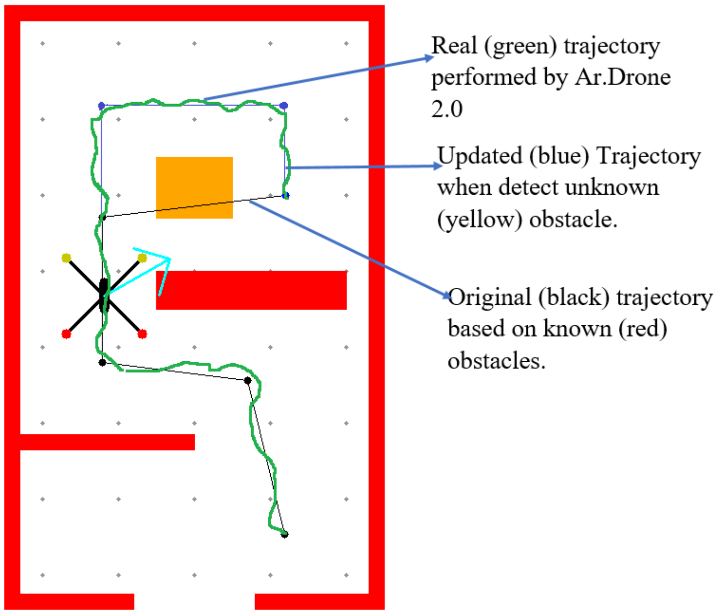

Real (green) trajectory performed by Ar.Drone 2.0
Updated (blue) Trajectory when detect unknown (yellow) obstacle.
Original (black) trajectory based on known (red) obstacles.

**Figure 5.31:** The real-time obstacle avoidance experiments of the Ar. Drone 2.0 with the pattern-based localization, the MPFM and the optimal PID controllers.

## 5.8 Summary

In this chapter, the entire analysis of the Potential Field Method (PFM) for an autonomous navigation and an improved approach in an unknown/dynamic environment is presented. The PFM is peculiarly attractive since it has a simple structure, a low computational complexity and it is easy to implement. In literature, there has been a significant amount of work applied to ground agents' path plannings based on this method. However, this approach sometimes encounters problems when the repulsion from the obstacles exceeded the mobile platform's physical constraints. It is pointed out that the potential field method has several inherent limitations in which the non-reachable target problem is the most serious one and it is worth investigating, since it causes an incomplete path in the navigation task.

The proposed PFM is tackling an autonomous navigation approach for a low-cost commercial Ar.Drone 2.0 using only the on-board visual and internal sensing. The main achievements for this chapter are:

(*i*) A localization method based on the sensor fusion technique regarding the time delay of the camera signals;
(*ii*) A proposed potential field method for path planning in unknown/dynamic environments;
(*iii*) The design and implementation of optimal PID controllers based on the proposed multi-objective particle swarm optimization (MOPSO) with an accelerated update methodology that allow accomplishing the path-following task.

The proposed modified potential field method (MPFM) is able to solve the non-reachable target problem and it successfully avoids the unknown/dynamic obstacles. The real-time experiments demonstrate the feasibility of the proposed strategy, which opens new possibilities of the autonomous navigation for the mobile agent in complex known, unknown and dynamic environments.

However, the main drawback of this approach is that the use of the pattern-based methodology for the localization process makes the UAV's working space limited. Chapter 6 solves this localization's problem and proposes an autonomous landing approach of the UAV in unpredictable environments.

# Chapter 6

# UAV Autonomous Landing in Presence of Uncertain Obstacles and GPS-Denied Environment

Nowadays, a safe autonomous flight at a low or a supper low-altitude is a fundamental requirement of an aircraft that must complete missions close to the ground, and such capability is widely requested. Search and rescue operations of a disaster perception allow different advantages at a low altitude. Similarly, Unmanned Aerial Vehicles (UAVs) performing reconnaissance or the military must fly at a low altitude in the presence of known and unknown obstacles. Until now, the UAVs have to land at the prepared sites with prior knowledge of the obstacle-free trajectories.

Establishing a safety landing on unpredictable environment is a crucial mission of an UAV. In Chapter 5, a fully autonomous navigation approach for the UAV is proposed where the identification, localization, path planning and optimal algorithms are provided.

In this chapter, a novel autonomous landing approach of the Ar.Drone 2.0 in presence of uncertain obstacles and GPS-denied environment is presented. The main contributions of this chapter are: $(i)$ providing a low-cost, high-scalability and easy-to-use autonomous landing of the UAV; $(ii)$ developing a sensor fusion method which fuses the signals of the IMU and the Poxyz for the UAV's localization in unknown environments; (*iii*) developing an obstacle detection algorithm which effectively detects unknown static/dynamic obstacles; (*iv*) a velocity estimation approach based on the UAV's vision system.

The results of this chapter are in revision at *IEEE/ASME Transactions on Mechatronics*.

T. T. Mac, C. Copot, C. M. Ionescu, A Vision Based Approach for UAV Autonomous Landing in Presence of Uncertain Obstacles and GPS Denied Environment, *IEEE/ASME Transactions on Mechatronics*, revision. (A1, rank: Q1, impact factor: 4.357)

## 6.1 Introduction

Multiple researches on UAVs' safe landing have been conducted in the literature. The intelligent landing algorithm of UAVs is described in [184]. In [185], the problem has been investigated to determine a feasible trajectory for landing. In [186], landing on a moving target has been simulated. An autonomous landing of an UAV on static and mobile platforms in absence of GPS is presented in [187]. Among different strategies for landing, the electro-optical sensors have been the most popular modality for landing site evaluation [188]. In [189], a system that uses vision technique for the control, terrain reconstruction, and tracking is presented. Another study is in [190], where the UAV's altitude is obtained by fusing the GPS measurements with the stereo vision provided by two embedded cameras. A novel safe landing-area determination system based on multiple impulse-radio ultra-wide-band (IR-UWB) radar is proposed in [191]. In which, the methods that based on the signals acquired by the IR-UWB for detecting the ground (slope and roughness) and the obstacles are studied.

Light Detection and Ranging (LIDAR) sensors have been extensively investigated for safe landing-area determination of small helicopters [175, 192] and fixed-wing airplanes [193], while the Global Positioning System (GPS) is used for navigation of an aircraft [194–196]. An automatic expert system, based on image segmentation procedures, that assists a safe landing through the recognition and relative orientation of the UAV and the landing platform, is proposed in [197]. In [198], a robust adaptive nonlinear guidance and control design based on the neural networks is presented. The proposed approach is verified through numerous simulation using the six degrees-of-freedom nonlinear model of a prototype UAV. In which, all possible disturbance effects that arise in practice (e.g., modeling inaccuracies, wind disturbances) are investigated. In [143], a bio-mimetic system is proposed that ascertains terrain appearances by using a monocular camera. In [199], a nonlinear controller for vertical take-off and landing processes (VTOL) of an UAV is described. In addition, the UAV uses the measurements of the optical flow to enable hover and to land on a moving platform (e.g., the deck of a sea-going vessel).

A novel method of cooperation between two drones at different altitudes is presented in [200] for the autonomous navigation and landing tasks. With the high flexibility and extensive vision, the higher drone can estimate the position of the lower drone and controls it to finish tracking marker and landing processes. In [201], a real-time system for UAV's pose estimation using a parallel image processing is studied. The system exploits the capabilities of a high-performance Graphics Processing Unit/Central Processing Unit (CPU/GPU) embedded system to provide safely autonomous take-off and landing.

Unlike the existing approaches, in our work, the UAV landing is performed in presence of uncertain obstacles using a sensor fusion algorithm which fuses the signals of a Pozyx and on-board sensors. In order to use the Pozyx, some modifications had to be made to the drone which caused an imbalance at hovering. Additionally, a velocity estimation algorithm is developed based on the on-board vision system. This velocity information can help an autonomous UAV interpret the landing task in complex situations. The main contributions of this chapter are:

- the development of a low-cost, high-scalability and easy-to-use navigation system to assistant the UAV's landing.
- the design and implementation of a sensor fusion technique.
- the velocity estimation (of moving obstacle) approach based on the UAV's vision system.
- the design and implementation of a cascade control for the drone's landing task.

## 6.2 System Setup and Localization Method

In this section, the setup system and the proposed localization based on a sensor fusion technique are presented. Assume that the UAV performs the landing task for a victim's rescue application in presence of uncertain static/moving obstacles with a designed target (victim-marked as a blue area), as shown in Figure 6.1.

Our approach consists of three major components running on a laptop (a ground station) connected to the quadrotor via wireless communications. The first component is a Poxyz, which allows the Ar.Drone 2.0 to determine its coordinates in the working space. The second component is an IMU, which delivers and receives signals between the quadrotor and the ground station. To obtain a more accurate the positions of the UAV, the sensor fusion method is implemented. The last component is a cascade control, which guides the quadrotor to the designed landing position.

In outdoor applications, the GPS usually provides a reasonable estimation; however, it is not an effective tool for indoor applications. There are several options for an indoor position estimation, such as the visual system, the on-board IMU or extra sensors. This section concerns with the state estimation of the UAV in an indoor environment using an IMU, a Pozyx, and a sensor fusion technique.

### 6.2.1 IMU Data Processing

The IMU measures the roll, pitch and yaw angles. The velocity estimations are obtained by fusing the information from a 3-axis gyroscope with the information from a 3-axis accelerometer and a magnetometer. The position estimations can be obtained by integrating the velocities. The $x$ position of the drone is calculated in equation 6.1.

$$x_k = x_{k-1} + \begin{bmatrix} \cos(yaw) & -\sin(yaw) & 0 \\ \sin(yaw) & \cos(yaw) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot V_{drone} \cdot \Delta t \qquad (6.1)$$

where:
$V_{drone}$ is velocity of the drone;

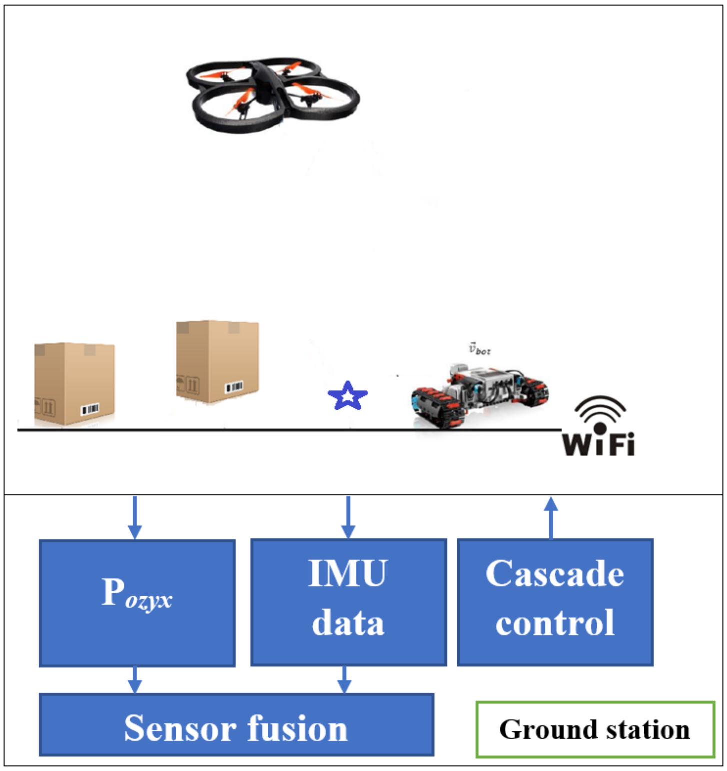


**Figure 6.1:** The proposed landing approach of the Ar.Drone 2.0 in a dynamic environment.

$x_k$, $x_{k-1}$ are positions of the drone in $x$ direction at samples $k$, $k-1$;
$\Delta t$ is sample time.
The $y$ position is calculated in similar approach. A better estimation of the height can be obtained by fusing the velocity in $z$-direction with the information from the ultrasonic sensor.

### 6.2.2 Pozyx

In this study, a Pozyx sensor is equipped on the Ar.Drone 2.0 as shown in Figure 6.2. The position estimation is obtained accurately by using ultra-wide band technology. Please refer to [202] for more details.

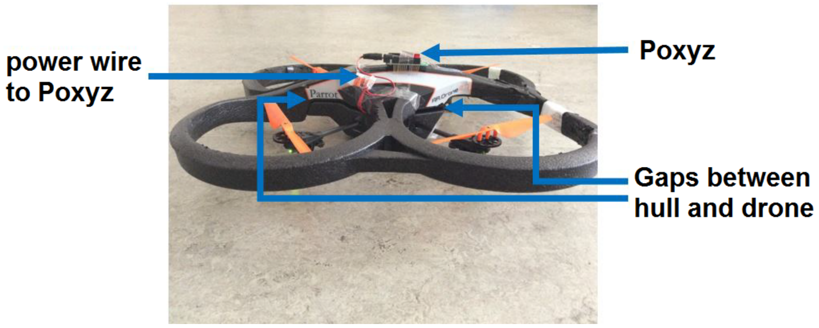


**Figure 6.2:** The Pozyx sensor on the Ar.Drone 2.0.

The following tools/libraries are recommended to start with the Pozyx system.

- The Arduino IDE. Download the IDE from `https://www.arduino.cc/en/Main/Software`.
- The Pozyx Arduino Library. The library can be obtained from `https://github.com/pozyxLabs/Pozyx-Arduino-library/archive/master.zip`
- The Processing IDE. Download the IDE from `https://processing.org/download/processing`.
- Processing libraries and examples. Download `https://github.com/pozyxLabs/Pozyx-processing`.

Three anchors are placed in a room. The anchors should not be placed on one line or on one plane. The absolute positions of these anchors need to be known in advance. By measuring the distance between the Pozyx's tag and each anchor, the position of the Pozyx's tag (the UAV's position) can be estimated. These were filtered out by defining a maximum allowed deviation of the current position compared to the previous one. When the deviation exceeds the threshold value, the current position is set to be equal to the previous one.

The Pozyx offers a robust position estimation. However, this signal is noisy so that it is not recommended to use the Pozyx as the only information source. The results obtained by unfiltered and filtered Pozyx's signals are shown in Figure 6.3.

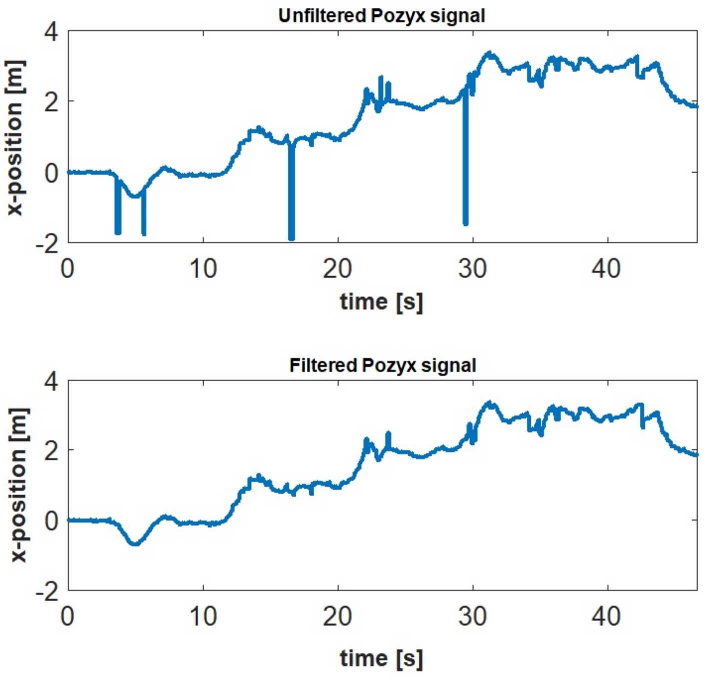


**Figure 6.3:** The unfiltered Poxyz's signal compared to the filtered Poxyz's signal.

### 6.2.3 Sensor Fusion

To combine the advantage of the IMU and the Pozyx, there is opted to use a Kalman filter to fuse these signals. A Kalman filter combines the information of different sensors and the expected states from the physical model to estimate the "true" states.

A Kalman filter consists of two phases: a prediction phase and an update phase. In the prediction phase, the Kalman filter estimates the position and its uncertainty based on the physical model.

$$\hat{x}_k = A \cdot \hat{x}_{k-1} \tag{6.2}$$

$$P_k = A \cdot P_{k-1} \cdot A^T + Q \tag{6.3}$$

where
$\hat{x}_k, \hat{x}_{k-1}$: The predictions of the current state and the previous state;
A: the state-transition model;
Q: the co-variance of process noise;
$P_k, P_{k-1}$: the current and previous estimated co-variances.
In the update phase, the observed measurements are used to find a new best estimate.

$$K = P_k \cdot C^T (C \cdot P_k \cdot C^T + R)^{-1} \tag{6.4}$$

$$\hat{x}_k = \hat{x}_k + K(z_k - C \cdot \hat{x}_k) \tag{6.5}$$

$$P_k = P_k - K \cdot C \cdot P_k \tag{6.6}$$

where:
$K$: the Kalman gain;
$C$: the observation model which maps the true state space into the observed space;
$R$: the co-variance of observation noise;
$z_k$: the measured signal.

The transition matrix $A$ was chosen as in equation 6.7. The variable $dt$ is equal to the mean execution time of the loop. With the used sensors, it is possible to measure the states directly, so the measurement matrix $C$ is equal to a unity matrix.

$$A = \begin{bmatrix} 1 & 0 & 0 & dt & 0 & 0 \\ 0 & 1 & 0 & 0 & dt & 0 \\ 0 & 0 & 1 & 0 & 0 & dt \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.7}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.8}$$

In order to design a Kalman filter, the process noise co-variance $Q$ and the measurement noise co-variance $R$ are added. The co-variances of the Pozyx were then tuned manually until the desired result was obtained.

$$Q = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.3 \end{bmatrix} \tag{6.9}$$

$$R = \begin{bmatrix} 10 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.05 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.05 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.05 \end{bmatrix} \tag{6.10}$$

Figure 6.4 shows an estimation of the position using the Kalman filter, the Pozyx and the IMU. In this experiment, the drone starts at $x \approx 0\ m$, then goes around 21 seconds to $x \approx 1\ m$ and finally goes around 30 seconds to $x \approx 2\ m$. It is not possible to only use the IMU for position estimation as it starts to drift very quickly. The Pozyx has a robust position estimation but it is noisy. The sensor fusion combines the advantages of both signals. It is a robust estimation without noise.

## 6.3 Obstacle Detection and Landing Algorithm

The obstacle detection is one of the key functionalities of a fully autonomous drone. As a proper detection of obstacles has many requirements, it is also one of the biggest challenges in the development process. In the real applications, the UAV cannot be equipped with many extra sensors because of the weight limitation. For the autonomous landing applications, both static/moving obstacles should be detected.

### 6.3.1 Static Obstacles

In order to detect the static obstacles, a solution based on distance measurements is proposed. The algorithm uses the difference in the absolute height

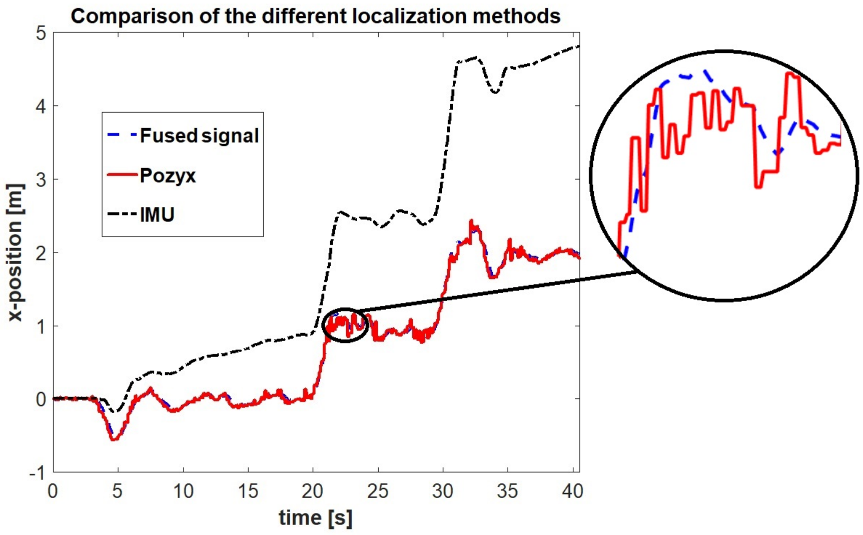


**Figure 6.4:** The comparison of the three different localization methods.

(the distance to the ground level $h_a$) and the relative height (the distance to the nearest obstacle $h_r$) to detect the obstacle. A graphical representation of the heights can be found in Figure 6.5.

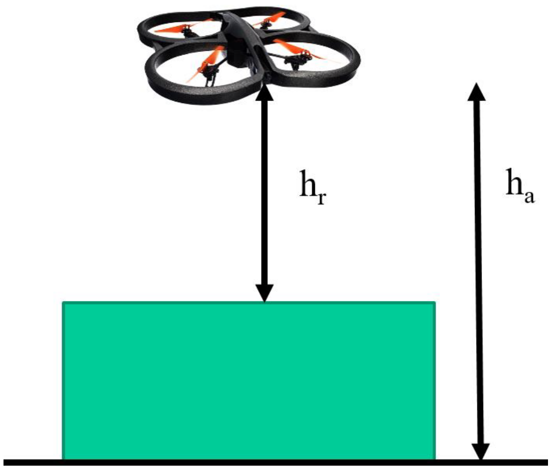


**Figure 6.5:** The absolute height and the relative height.

### 6.3.2 Moving Obstacles

The moving obstacle is marked with a red paper. Based on the color detection algorithm, it is possible to extract the coordinates of the obstacle in the image frame (please refer to Chapter 5 for more details). These coordinates need to be converted to the real-world coordinates to obtain the obstacle's position/ velocity information. Figure 6.6 presents a moving obstacle detection using the color segmentation. The images were acquired by using the UAV's bottom camera.

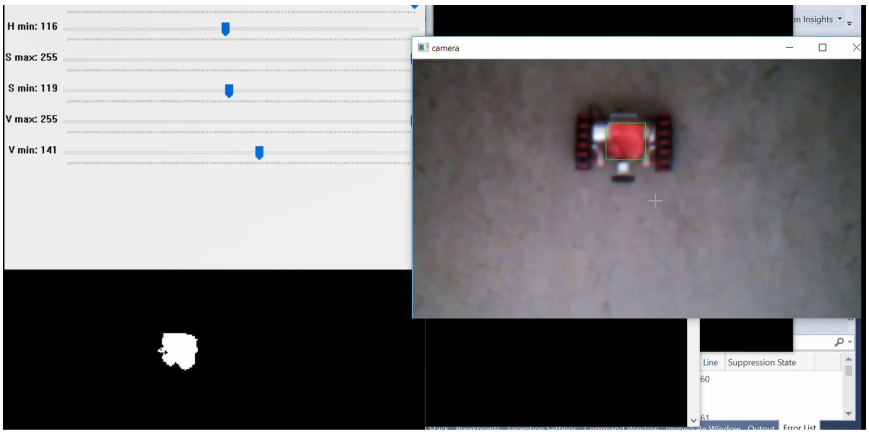

H min: 116
S max: 255
S min: 119
V max: 255
V min: 141

**Figure 6.6:** The color-based tracking moving obstacle.

In order to convert the obstacle's positions from the image coordinates to the world coordinates, the calculation is implemented based on the diagram shown in Figure 6.7. In which, $M$ is the center point of the received image. $P$ is the position of the Lego robot in the image frame. $AOV_x$ is the field of view's angle and $D$ presents the UAV's bottom camera. $O$ is the projection point of $D$ on the horizontal plane. $W$ is the intersection point between $DP$ and the horizontal plane. The following formulas are applied.

$$|DM| = \frac{|KM|}{\tan(AOV_x/2)} \tag{6.11}$$

$$\alpha = \arctan(\frac{|PM|}{|DM|}) \tag{6.12}$$

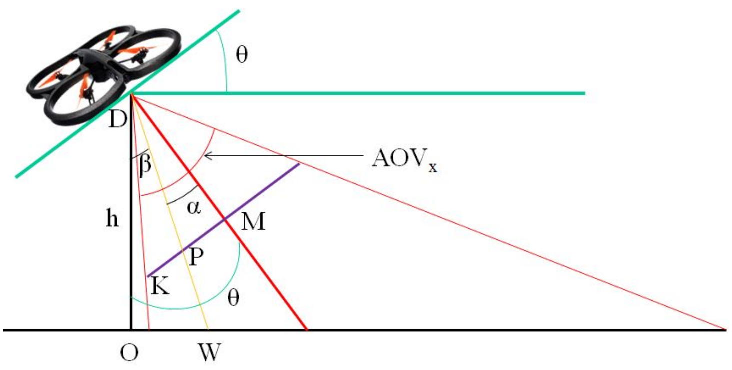


**Figure 6.7:** The conversion from the pixel coordinates to the real-world coordinates.

$\beta$ represents $\theta \pm \alpha$, the sign depends whether $X_P > X_M$ or $X_P < X_M$.
We define $\alpha'$ as:

$$\alpha' = \arctan(\frac{X_P - X_M}{|DM|}) \tag{6.13}$$

We have:

$$\beta = \theta + \alpha' \tag{6.14}$$

$$x_W = \tan(\beta) \cdot h \tag{6.15}$$

with $DO = h$
The absolute world $x$-coordinate of the object is:

$$x_{obj} = x_{drone} + \cos(yaw) \cdot x_W - \sin(yaw) \cdot y_W \tag{6.16}$$

where:
$x_W, y_W$: the world coordinates in $x$ and $y$ directions of point $W$;
$x_{obj}$: the $x$-coordinate of the object;
$x_{drone}$: the $x$-coordinate of the drone;
$yaw$: the yaw angle of the drone.
In a similar approach, the $y$-coordinate of the object (moving obstacle) is:

$$y_{obj} = y_{drone} + \cos(yaw) \cdot y_W + \sin(yaw) \cdot x_W \tag{6.17}$$

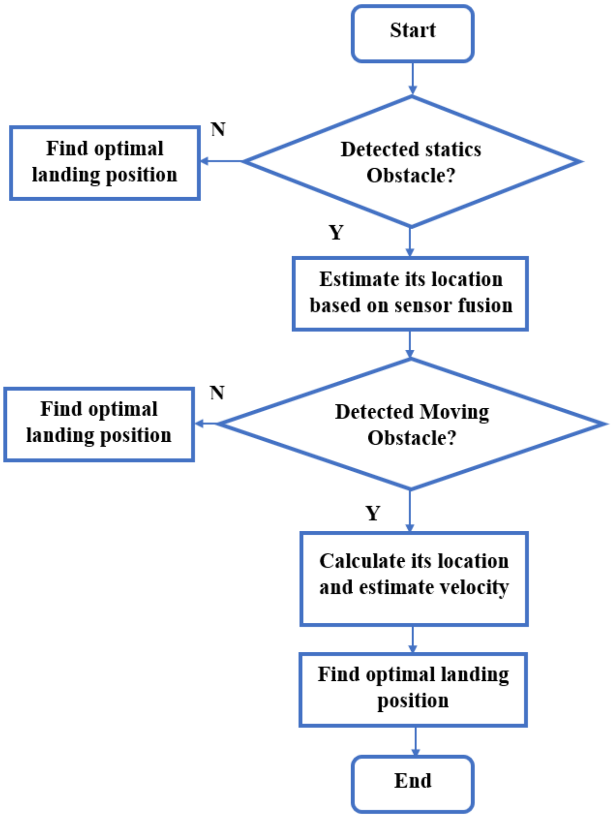

Start
Detected statics Obstacle?
N
Find optimal landing position
Y
Estimate its location based on sensor fusion
Detected Moving Obstacle?
N
Find optimal landing position
Y
Calculate its location and estimate velocity
Find optimal landing position
End

**Figure 6.8:** The UAV's landing algorithm flowchart.

where
$y_{obj}$: the $y$-coordinate of the object;
$y_{drone}$: the $y$-coordinate of the drone.

### 6.3.3 Landing Algorithm

In our approach, the landing algorithm based on the above obstacle detection method is presented in Figure 6.8. In which, depending on a specific situation, the drone decides the best location for its landing task.

Figure 6.9 presents an example of the UAV's landing application for the victim rescue purpose [203]. A horizontal plane for a landing is presented in Figure

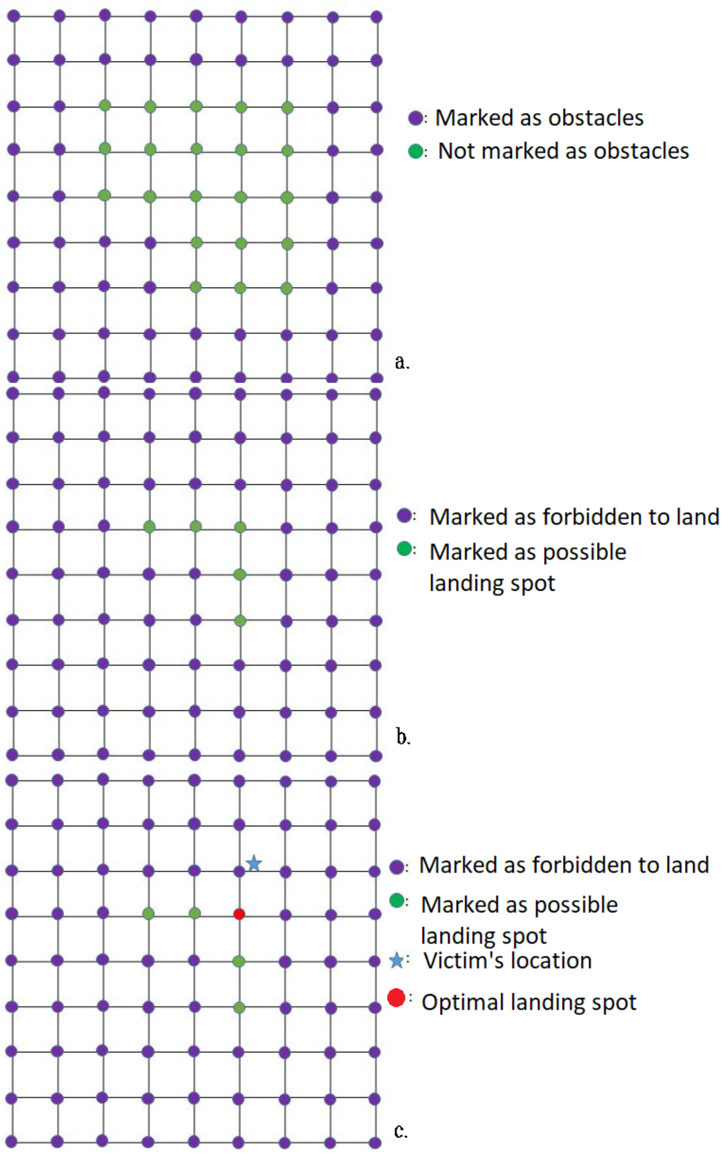

●: Marked as obstacles
●: Not marked as obstacles
a.
●: Marked as forbidden to land
●: Marked as possible landing spot
b.
●: Marked as forbidden to land
●: Marked as possible landing spot
★: Victim's location
●: Optimal landing spot
c.

**Figure 6.9:** The UAV's landing mapped protocol.

6.9. In which, the purple areas are occupied by the obstacles and the green areas are free for the landing task. Taking into account the dimension of the drone, the area that is marked as "obstacle" should be extended by the (rounded up) radius of the drone. Therefore, the real possible landing spots are the green locations, as shown in Figure 6.9b. Figure 6.9c presents the optimal landing's spot in terms of the distance to the target location (the victim's position).

Figure 6.10 presents the UAV's landing results in 3D space with 6 obstacles. The starting point is (0.6,0.4,0.6); the victim position is (0.2,0.6,0).

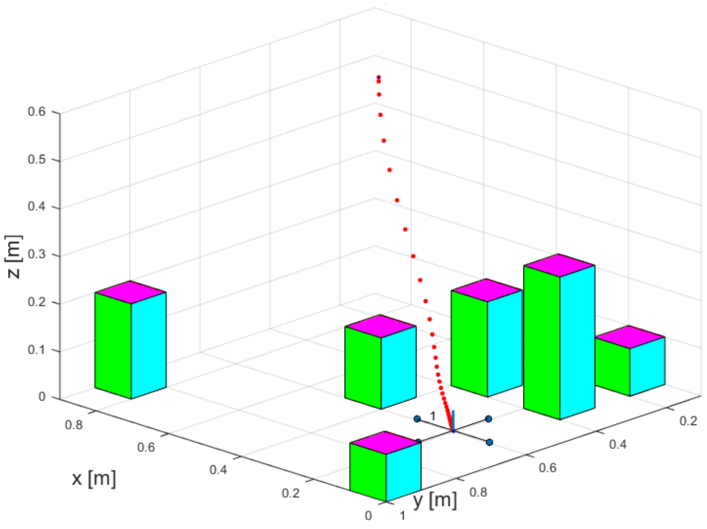


**Figure 6.10:** 3D UAV's landing simulation.

## 6.4 Position Control Design

In this section, the UAV's cascade control is designed for the landing process. The cascade control is divided into an outer-loop and an inner-loop. The outer-loop transforms the landing commands into the UAV. We used the same PID controllers for this outer-loop control as the ones in Chapter 5. A schematic representation is depicted in Figure 6.11.

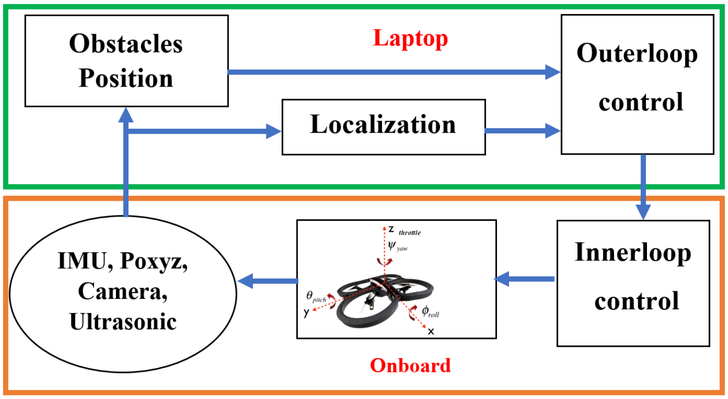


**Figure 6.11:** The cascade control scheme of the quadrotor.

## 6.5 Simulations and Experimental Results

In this section, the experimental results are presented. Firstly, the results of the moving obstacle's velocity estimation are illustrated. Lastly, the UAV landing process in the unknown static/dynamic environment is validated.

### 6.5.1 Velocity Estimation of Moving Obstacles Results

To safely land in presence of a moving obstacle, a crucial requirement is the perception of the object's motions. To validate our approach, the velocity estimation system setup consists of a Lego Mindstorm that is driving with a constant speed of 0.2 m/s and an Ar.Drone 2.0 that estimates the velocity of the robot using its bottom camera. To obtain a proper velocity estimation, a cumulative moving average filter is applied in this application. This result shows that the measurement is reasonable at the first stage (error is smaller than 0.015 m/s), as shown in Figure 6.12. Thereafter, the velocity estimation is quite accurate, as shown in Figure 6.13. The filter is implemented in C++ as described in Algorithm 2.

The results of the experiment suggest that the velocity estimation by the drone's vision system is feasible. Even regarding delays, the accurate velocity estimations of the moving obstacles are possible. This velocity information

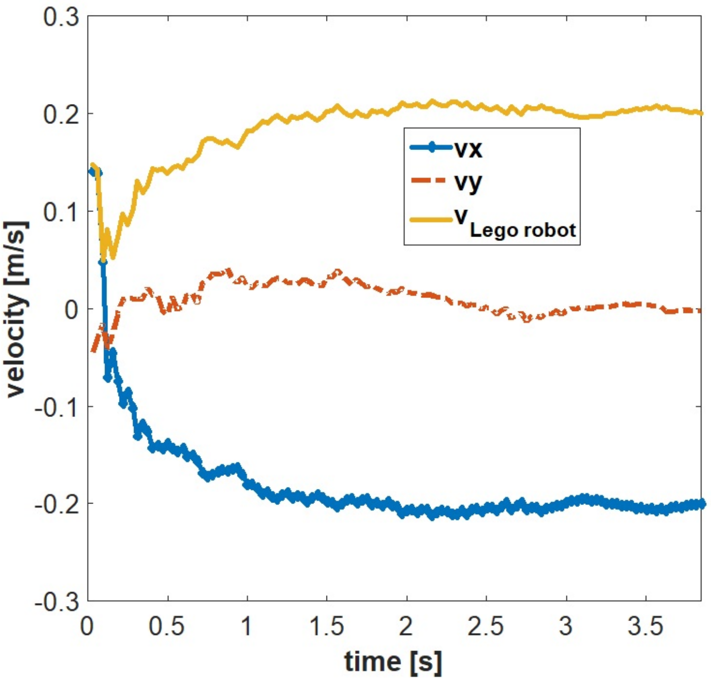


**Figure 6.12:** The cumulative moving average filter that is applied on the velocity measurements of the moving obstacle.

**Algorithm 2 Implementation of the filter - pseudo-code**

```
if $(abs(xpoz - x\_poz\_prev) > 500 \quad \& \quad x\_over\_lim\_counter < 12)$ {
$xpoz\_filter = x\_poz\_prev;$
$x\_over\_lim\_counter = x\_over\_lim\_counter + 1;$
}
else {
$x\_over\_lim\_counter = 0;$
$xpoz\_filter = xpoz;$
}
$x_poz\_prev = xpoz\_filter;$
```

enables the autonomous UAVs to interpret complex situations and to find the

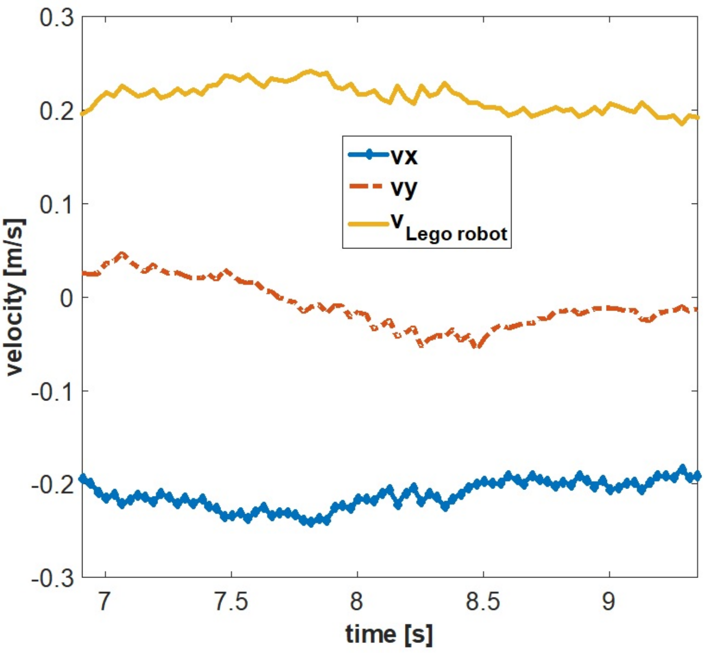


**Figure 6.13:** Moving average filter that is applied on the velocity measurements of the moving obstacle (continued).

suitable position for landing.

## 6.5.2 Autonomous Landing Results

The idea of an autonomous landing task is to find a position in an environment where there is no static obstacle, no potential moving obstacle and the distance to the target is the shortest. As soon as the drone has detected a static obstacle by an ultrasonic sensor, the position of that obstacle is provided by the sensor fusion method. In addition, the moving obstacle is detected by the vision-based approach and its position/velocity information is provided as presented in the previous subsections.

**Case I: Landing in presence of static obstacles.**
The experiment for UAV's landing in presence of static obstacles was conducted in a room with full furniture, such as sofa, chair, table, etc.; the target's

(victim's) location is marked as a blue area on the ground (see Figure 6.14). The results showed that the drone avoided all the obstacles on its path and autonomously landed on the optimal position (victim's position). The intended accuracy had been achieved by using the equipped ultrasonic sensor.

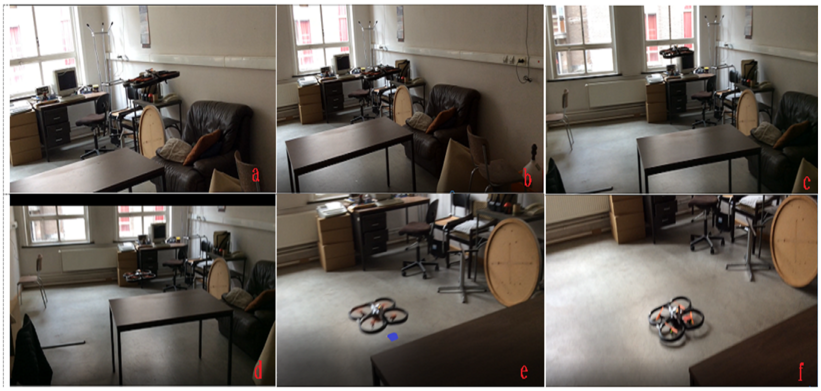


**Figure 6.14:** The UAV's landing in presence of static obstacles results in the sequence of time. a. fly over static obstacle (sofa); b. fly over static obstacle (table); c. fly to the target position; d. closer to the target; e. prepare to landing; f. landing.

**Case II: Landing in presence of moving obstacles.**

In this experiment, a Lego Mindstorms EV3 was used to represent a moving obstacle as depicted in Figure 6.15. The moving obstacle is marked with a red marker and the target's location (victim's position) is marked as the blue area on the ground. Based on the color detection, it is possible to extract the coordinates of the obstacle in the image frame.

In Figure 6.15a, the drone detected the Lego robot as a moving obstacle. Since the target position was on the robot's path, the UAV still followed the robot and flew to the position near the target; however, it did not decide landing immediately. The drone hovered and waited until the robot passed the target position then it safely landed as illustrated in Figure 6.15c.

The results proved that the proposed approach allowed the UAV to land safely and it reached the optimal position in term of the shortest distance to the target.

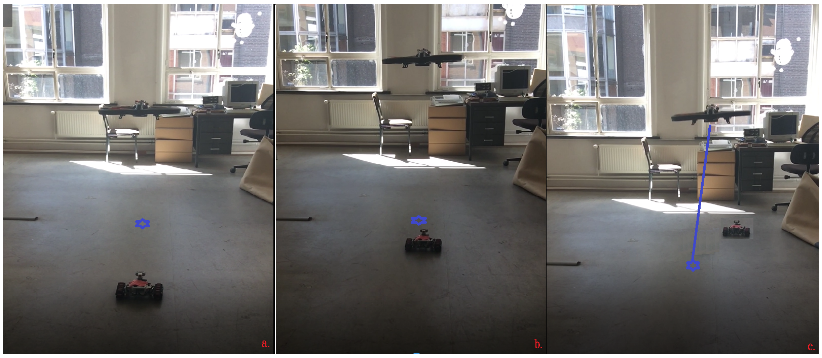


**Figure 6.15:** Landing in presence of dynamic obstacles results in the sequence of time from (a) to (c).

## 6.6 Summary

This chapter presents a novel autonomous landing approach for a commercial Ar.Drone 2.0 in presence of uncertain obstacles and GPS-denied environments. The system is described from electromechanical point of view.

The main achievements for this research are: (i) the development of a low-cost, high-scalability and easy-to-implement navigation system to assist the UAV's landing in terms of avoiding unknown/dynamic obstacles; (ii) the design and implementation of a localization method based on a sensor fusion which fuses the IMU's signal and the Poxyz's signal; (iii) a proposed unknown static obstacle and dynamic obstacle detection approach; (iv) an effective estimation of a moving obstacle's velocity based on the drone's vision system; (v) the development of a cascade control which allows the drone to safely land. The results indicate that the obtained model fits well to the measured data and the designed control strategy is able to effectively control the drone. In addition, the proposed autonomous landing strategy can guide the UAV to avoid static/dynamic obstacles and land on the optimal position.

However, only considering the moving obstacle with a constant speed is the limitation of this work. Future work includes an extension of the proposed approach to multiple UAVs and multiple ground vehicles moving with different speeds. The simultaneous localization and mapping (SLAM) is also a prospective direction to update a map of an unknown environment while simultaneously keeping track of other agents' locations. These futher improvements are suggested in the next chapter.

# Chapter 7

# Conclusions

The main contributions of this thesis are emphasized and several suggestions for possible further research directions are outlined.

## 7.1 Contributions

Each subsystem in the large scale heterogeneous autonomous navigation system may have different dynamic characteristics and it works in various environments. It is necessary to develop safe autonomous navigation strategies, especially path planning algorithms, in different scenarios for different mobile platforms.

The path planning in a diversity of environments; such as static (Chapter 3), cluttered (Chapter 4) and unknown/dynamic (Chapter 5, Chapter 6) working spaces are studied in this thesis. In each application, based on the various purposes, the system's characteristics, the working spaces and the user's experiences, a proper methodology is proposed to accomplish the robot's assignments. Firstly, this problem is studied in a real-life application that highly desires to capture the expertise of an operator (Chapter 3). Secondly, a multi-objective optimization path planing problem taking into account the constraints of the mobile agents is developed (Chapter 4). Lastly, this topic is expanded in unpredictable environments (Chapter 5, Chapter 6).

Besides, the localization, perception and control approaches are examined in the real-time experiments (Chapter 5 and Chapter 6).

In summary, the main added values of this thesis are (*i*) providing an up-to-date literature review of autonomous navigation approaches; (*ii*) the obtained suitable path plannings in different operating environments; (*iii*) the development of a low-cost, high-scalability and easy-to-use system to assist the autonomous navigation and landing of an UAV; (*iv*) A localization system based one the sensor fusion technique and (*v*) the design and implementation of optimal controllers. The main contributions of each chapter are summarized as follows.

**Literature Review**

An extended review of robot path planning algorithms in last decades has been provided. The heuristic path planning methods analyzed in this chapter consist of neural network, fuzzy logic and nature-inspired methods (genetic algorithm, particle swarm optimization and ant colony optimization). The applications of the reviewed studies are summarized in Table B.1, Appendix B.

The classical methods are easy to implement. Therefore, they are preferred in many real-time motion planning applications and may obtain good results as shown by RRT, PFM, PRM. However, the classical approaches often require precise information about the robot's working environment, thus more accurate sensors should be used in the real-time applications. Although they work very effectively, they provide no theoretical guarantees for reaching an optimal solution. Compared to the classical methods, the heuristic methods are indicated to be more "intelligent" and more "advanced" as they can match to both uncertain and incomplete information in constantly changing environments (i.e. neural network, fuzzy logic methods) and achieve optimal solutions (nature-inspired methods). Thus, they are employed in complex environments. However, they have some shortcomings, such as the necessary learning phase and the high computational time. In many applications, the computational capacity of the robot controller unit is limited, therefore the approaches with lower computational cost, such as RRT, PFM are much preferred.

The heuristic-based algorithms promise new directions for the autonomous navigation. They have the ability of finding the optimal solutions in dynamic environments. However, they are time-consuming. Despite the fact that a great amount of the theoretical works has been validated in simulations and off-line processing, there is a gap between the theoretical and experimental works. The current robot autonomous navigation algorithms have not yet obtained the level of robustness and reliability required for the real-world applications. Autonomous navigation in unknown/dynamic environments maintains a challenging aspect for researchers.

There is no uniquely defined the best solution for all applications. A suitable path planning algorithm depends on the user's purposes, system characteristics, working spaces and experiences. In addition, fuzzy logic, integrated method and new potential method are promising approaches in robot autonomous navigation domain. Those approaches were the investigated research topics of Chapters 3, 4 and 5.

**Fuzzy-Based Autonomous Navigation of a Mobile Robot in Outdoor Environments**

The development of an autonomous navigation system that highly desires to capture the expertise of an operator is produced in this chapter. An autonomous geophysical measurement platform (AGMP) is investigated as a case study in this chapter. The AGMP works in outdoor environments with different terrains, such as smooth, rough or rocky ones, therefore it should also be able to adjust its speed with changes in terrain. The main contributions of this chapter are:

- an intelligent control algorithm which easily adapts to different terrains (e.g. a high speed where its road is smooth, a low speed where its road is rocky) and obstacles (e.g. speed up in case no obstacle and slow down otherwise).
- an effective speed control strategy of the geophysical measurement platform is proposed for archaeological prospection.
- the development of a modular structure that can be extended to incorporate new behaviors.

Although the proposed fuzzy control effectively works for the AGMP's implementation, in other applications where mobile robots are required to perform their tasks in environments without any knowledge of operators. In addition, an optimal collision-free path under certain constraints is desired. Therefore, it is necessary to investigate the robot path planning as a constrained multi-objective optimization problem. This approach and the proposed optimal path planning have been introduced in Chapter 4.

## Hierarchical Global Path Planning Approach on Multi Objective Particle Swarm Optimization

This chapter studies a multi-objective optimal path considering the abilities of mobile robots in a cluttered environment. From the literature review provided in Chapter 2, it has been found that each of the robot path planning approaches has its own limitations and so far, an individual algorithm cannot perfectly solve the robot path planning. Therefore, integrated methods are better solutions for this problem. A hierarchical approach which combines several algorithms is proposed in this chapter to reduce the complexity of operations and to improve the quality of the robot's trajectory. The aim of the approach is to the generate optimal collision-free paths focused on minimizing the path length and maximizing the smoothness regarding the limitations of the robot's characteristics.

The proposed method has a three-level structure. In the first level, the triangular decomposition is used to divide the robot's working environment into obstacle configuration space and free configuration space. Next, in the second level, Dijkstra's algorithm is applied to find the collision-free path from the starting point to the goal point. Finally, a constrained multi-objective particle swarm optimization (CMOPSO) with an accelerated update methodology based on Pareto dominance principle is proposed to find the optimal path. In summary, the main contributions are the development of:

- a novel hierarchical global path planning method for mobile robots in cluttered environments;
- a proposed particle swarm optimization algorithm with an accelerated update methodology based on Pareto dominance principle;
- an optimal global robot path method in terms of the path length and the path smoothness considering the constraints of mobile robots.

The hierarchical global path planning approach, however, is more suitable for the environments where all the information about the obstacles and the targets are provided in advance.

### Potential Field Method of an UAV's Autonomous Navigation

Chapter 5 implements an autonomous methodology for a low-cost commercial Ar.Drone 2.0 in unknown/dynamic environments using only on-board visual and internal sensors. As an UAV is a complex system where electromechanical dynamics is involved, a robust autonomous navigation system is a crucial aspect. The method is proposed to trigger the UAV's identification, localization, detection of obstacles, optimal path planning and control algorithms. The added values include:

- the development of a position-estimation method using a sensor fusion technique in a structured environment. This localization process presents how to get the UAV's states (the position and the orientation), dealing with data provided by an optical sensor and an inertial measurement unit (IMU). Such a data fusion scheme takes also into account the time delay of the camera signals due to the communication protocols;
- The improved potential field method which is capable of performing obstacle avoiding in an unknown/dynamic environment and solving the non-reachable goal problem;
- the design and implementation of an optimal Proportional-Integral-Derivative (PID) controller based on the novel multi-objective particle swarm optimization with an accelerated update methodology. This algorithm facilitates convergence to the PID's optimal parameters.

Experimental results validate the effectiveness of the proposed approach.

### UAV Autonomous Landing in Presence of Uncertain Obstacles and GPS-Denied Environment

In this chapter, a vision-based approach for an UAV's autonomous landing is developed in presence of static/moving obstacles and GPS-denied environment. The proposed image processing algorithm allows the UAV to immediately detect the operating environment through its sensors and to appropriately decide the actions. The main achievements for this chapter are:

- the design and implementation of a sensor fusion method which fuses the signals of the Pozyx and on-board sensors.
- the velocity estimation method (of a moving obstacle) based on the UAV's vision system. This velocity information can help an autonomous UAV to interpret landing task in complex situations.
- the development of a low-cost, high-scalability and easy-to-use navigation system to assist landing of an UAV.
- a method to detect static/dynamic obstacles using information obtained from the UAV's camera.

## 7.2 Directions for Future Research

Now, mobile agents are on the road of a fully autonomous navigation. Still much research is required, hence several recommendations for future work are listed below.

### Drones Cooperation

One prospective direction is the development of cooperative multiple drones. This produces the possibility for the drones to share maps and location of obstacles while navigating. Here, the issue of collisions between a swarm of agents needs to be prevailed. Distributed control algorithms maybe useful to investigate.

### Formation Control of UGVs Using UAVs

Each robot in the formation is an autonomous system, specialized on developing certain tasks, enhancing the flexibility of the overall system and allowing easy adaptation to different specifications. The application of UGVs to real-world scenarios requires the consideration of many challenging details that may increase the complexity of the implementation steps. The ability to interact with dynamic, changing environments is important. The agents must be able to handle various unexpected events that can disrupt the formation, thus requiring obstacle avoidance, formation repair and changing in the formation. For which, the higher levels of decision become vital. Techniques from artificial intelligence that allow identification of strategies and tactics may be needed. Furthermore, the robots themselves must satisfy dynamic constraints, such as velocity and acceleration bounds. However, UGVs have limited view of their working environment. Therefore, the formation control of UGVs and UAVs is a promising research direction.

### Energy Consumption Optimization

The short flight time is a disadvantage of the UAV systems and several research groups are working on this field. This problem could be solved by using optimization algorithms in order to improve the energy consumption and the duration of the experiments.

### Obstacle Information

The information about the obstacle should not be limited to its position and velocity but also to its projection of path in the future (i.e. predicted direction). Additional information can drastically enhance the safety. An autonomous drone should be able to differentiate between land and sea; between the obstacles that are controlled by humans, such as cars and animals which have many unpredictable behaviors.

### Simultaneous Localization and Mapping (SLAM)

SLAM is a prospective direction for autonomous navigation. The essential problem in SLAM is to estimate the robot's location and the map of the environments, typically represented by a set of geometric features. SLAM algorithms have to take into account a variety of parameters, i.e., sensors, map representation, robot dynamics, environmental dynamics, the integration of sensor measurements and the robot's control system over time. The development of SLAM systems in the era of autonomous navigation has put into question how to reduce the computational complexity and make use of SLAM algorithms to operate in real-time applications.

**Large-Scale Homogeneous/Heterogeneous Autonomous Navigation System**

This thesis developed path planning algorithms for each subsystem in a diversity of environments; such as static, cluttered and unknown/dynamic ones of a large-scale heterogeneous autonomous navigation system. The localization, obstacle avoidance and optimal control methodologies are also investigated. Further study could be to integrate all the proposed algorithms in such a large-scale system.

# Appendices

Appendix A

# List of the Author's Publications

## Journals Cited in Web of Science

1. T. T. Mac, C. Copot, D. T. Tran, R. De Keyser, A hierarchical global path planning approach for mobile robots based on multi-objective particle swarm optimization, *Applied Soft Computing*, vol. 59, pp. 68-76, 2017.
2. T. T. Mac, C. Copot, R. De Keyser, C. M. Ionescu, The development of an autonomous navigation system with optimal control of an UAV in partly unknown indoor environment, *Mechatronics*, vol. 49, pp. 187-196, 2018.
3. T. T. Mac, C. Copot, C. M. Ionescu, A vision based approach for UAV autonomous landing in presence of uncertain obstacles and GPS-denied environment, *IEEE/ASME Transactions on Mechatronics*, revision.
4. T. T. Mac, C. Copot, D. T. Tran, R. De Keyser, Heuristic approaches in robot path planning: a survey, *Autonomous and Robotics System*, vol. 86, pp. 13-28, 2016.

## Conference Proceedings Cited in Web of Science

1. T. T. Mac, C. Copot, D. T. Tran, R. De Keyser, A hierarchical global path planning based on multi-objective particle swarm optimization,

*IEEE International Conference on Methods and Models in Automation and Robotics*, Miedyzdroje, Poland, pp. 930-935, 2016.

2. T. T. Mac, C. Copot, D. T. Tran, R. De Keyser, Ar.Drone UAV control parameters tuning based on particle swarm optimization algorithm, *IEEE International Conference on Automation, Quality and Testing, Robotics*, Cluj-Napoca, Romania, pp. 475-480, 2016.

3. T. T. Mac, C. Copot, A. Hernandez, R. De Keyser, Improved potential field method for unknown obstacle avoidance using UAV in indoor environment, *IEEE Intentional Symposium on Applied Machine Intelligence and Informatics*, Herlany, Slovakia, pp. 345-350, 2016.

4. T. T. Mac, C. Copot, L. Verdonck, R. De Keyser, Speed control strategy of geophysical measurement platform for archaeological prospection: conceptual study, *IEEE International Conference on Systems, Man, and Cybernetics*, Budapest, Hungary, pp. 3748-3753, 2016.

5. C. Copot, A. Hernandez, T. T. Mac, R. De Keyser, Collision-free path planning in indoor environment using a quadrotor, *IEEE International Conference on Methods and Models in Automation and Robotics* , Miedzyzdroje, Poland, pp. 351-356, 2016.

6. T. T. Mac, C. Copot, C. M. Ionescu, Detection and estimation of moving obstacles for an UAV, *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Auckland, New Zealand, submitted.

7. C. Copot, T. T. Mac, C. M. Ionescu, PID based particle swarm optimization in offices light control, *IFAC Conference on Advances in Proportional-Integral-Derivative Control*, Ghent, Belgium, pp. 382-387, 2018.

8. T. T. Mac, C. Copot, C. M. Ionescu, Design and implementation of a real-time autonomous navigation system applied to Lego robots, *IFAC Conference on Advances in Proportional-Integral-Derivative Control*, Ghent, Belgium, pp. 340-345, 2018.

9. C. Copot, T. T. Mac, C. M. Ionescu, An application to robot manipulator joint control by using constrained PID based PSO, *IEEE International Symposium on Applied Computational Intelligence and Informatics*, Timisoara, Romania, accepted.

10. J. Juchem, S. Lefebvre, T. T. Mac, C. M. Ionescu, An analysis of dynamic lighting control in landscape offices, *IFAC Conference on Advances in Proportional-Integral-Derivative Control*, Ghent, Belgium, pp. 232-237, 2018.

## Others

1. T. T. Mac, C. Copot, R. De Keyser, D. T. Tran, T. Vu MIMO fuzzy control for autonomous mobile robot, *Journal of Automation and Control Engineering* , vol. 4(1), pp. 65-70, 2016.

2. T. T. Mac, D. T. Tran, T. Vu, Path Following of mobile robot using Fuzzy controller, *Vietnam Journal of Science and Technology*, vol 53 (2), pp. 221-231, 2015.

3. T. T. Mac, C. Copot, D. T. Tran, R. De Keyser, MIMO fuzzy control for autonomous mobile robot, *International Conference on Intelligent and Automation Systems*, Ho Chi Minh, Vietnam, pp. 277-282, 2015.

4. T. T. Mac, C. Copot, D. T. Tran, R. De Keyser, Hybrid potential bacterial foraging optimization algorithm with robot swarm obstacle avoidance, *34th Benelux Meeting on Systems and Control*, Amsterdam, the Netherlands, pp.113-113, 2015.

5. T. T. Mac, C. Copot, D. T. Tran, R. De Keyser, Towards a navigation system for an autonomous UAV in indoor environment, *35th Benelux Meeting on Systems and Control*, pp.138-138, 2016.

# Appendix B

# Application of NN-FL-GA-PSO-ACO-PFM for the Robot Path Planning

| **Applications of NN-FL-GA-PSO-ACO-PFM for the robot path planning** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Approach | Authors | Kinematic model | Obstacle shape | Static Obstacle | Dynamic Obstacle | Static Target(s) | Dynamics Target(s) | Adapted velocities | Simulation | Real system | Year |
| Neural | S.X. Yang [57] | N | Rectangle | Y | Y | N | Y | N | Y | N | 2000 |
| | E. A. Antonelo [58] | N | Arbitrary | Y | N | Y | N | N | Y | N | 2010 |
| | D. Janglova[3] | Y | Arbitrary | Y | N | Y | N | N | Y | Y | 2004 |
| | S.H. Dezfoulian [59] | N | Arbitrary | Y | N | Y | N | N | Y | N | 2013 |
| | M.K. Singh [4] [60] | Y | Arbitrary | Y | N | Y | N | N | Y | Y | 2009 |
| | M.K. Singh [61] | Y | Arbitrary | Y | N | Y | N | N | Y | Y | 2011 |
| | M. Sagban [62] | Y | Arbitrary | Y | N | Y | N | N | Y | Y | 2012 |
| | I. Engedy [63] | Y | With markers | Y | Y | Y | N | Y | Y | Y | 2009 |
| NN/ PSO | X. Chen [64] | Y | Circle | Y | Y | Y | Y | Y | Y | Y | 2006 |
| NN | D. L. Cruz [99] | Y | Rectangle | Y | N | Y | N | N | Y | Y | 2017 |
| NN/FL/PFM | Q. Q. Wu [102] | Y | Arbitrary | Y | Y | Y | N | N | Y | N | 2017 |
| | H. Chang [5] | Y | Corridor | Y | N | Y | N | N | Y | Y | 2013 |
| | V.M. Peri [66] | N | Arbitrary | Y | Y | Y | N | N | Y | Y | 2005 |
| | J.H. Lilly [67] | N | Circle | Y | N | Y | N | N | Y | N | 2007 |
| FL | A. Zhu [68] | N | Arbitrary | Y | Y | Y | N | N | Y | N | 2004 |
| | Y. Tanaka [69] | Y | Arbitrary | Y | N | Y | N | N | Y | Y | 2015 |
| | A. Foudil [70] | N | Arbitrary | Y | Y | Y | N | Y | Y | Y | 2014 |
| | L. Li [71] | N | Arbitrary | Y | N | Y | N | N | N | Y | 2009 |
| | C.H.Chao[72] | N | Arbitrary | Y | N | Y | N | N | N | Y | 2009 |
| | G. Mester [73] | N | Arbitrary | Y | N | Y | N | Y | Y | N | 2008 |
| FL-GA | T. Lee [74] | N | Arbitrary | Y | N | Y | N | Y | Y | Y | 2003 |
| FL−PSO | M. S. Masmoudi [103] | Y | Arbitrary | Y | N | Y | N | Y | Y | Y | 2016 |
| FL−PSO | C. Hong [104] | Y | Circle | Y | N | Y | N | N | Y | N | 2016 |
| FL | N. Kumar [105] | Y | Arbitrary | Y | N | Y | N | Y | Y | N | 2017 |
| NN-FL | D.R. Parhi [6] | N | Arbitrary | Y | N | Y | N | Y | Y | Y | 2008 |
| | S.K. Pradhan [106] | N | Arbitrary | Y | N | Y | N | Y | Y | Y | 2006 |
| PAFARTNA | R. Araujo [75] | Y | Arbitrary | Y | Y | Y | N | N | Y | Y | 2006 |
| NN-FL | A. Zhu [7] | N | Arbitrary | Y | N | Y | N | Y | Y | Y | 2009 |
| | M.M. Joshi [76] | N | Arbitrary | Y | N | Y | N | Y | Y | N | 2011 |
| NN-FL, GA−Anfis, PFM | N. B.Hui [77] | Y | Circle | N | Y | Y | N | Y | Y | N | 2006 |
| NN-FL, PSO | Y. Gou [107] | Y | Rectangle | Y | N | Y | N | N | Y | N | 2017 |
| NN-FL, A* | Z. Shi [108] | Y | Arbitrary | Y | N | Y | N | N | Y | N | 2016 |
| GA | Y. Hu [79] | N | U shape, Rectangle | Y | N | Y | N | N | Y | N | 2004 |
| | S.X. Yang [80] | N | Arbitrary | Y | Y | Y | N | N | Y | N | 2006 |
| GA based DPPA | S.C.Yun [81] | N | Parallelepiped | Y | Y | Y | N | N | Y | Y | 2011 |
| GA-FL, A* | B.K Oleiwi [8] | Y | Circle | Y | Y | Y | N | N | Y | N | 2014 |
| GA-FL | B. Karim [82] | N | Point | N | N | N | Y | Y | Y | N | 2013 |

| **Applications of NN-FL-GA-PSO-ACO-PF for the robot path planning (continued)** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| GA– PSO | H.C. Huang [83] | N | U shape, Rectangle | Y | N | Y | N | N | Y | N | 2011 |
| GA–FL | S.M.R. Farshchi [84] | Y | Cylinder | Y | Y | Y | N | N | Y | N | 2011 |
| GA | A.K. Karami [85] | N | Trap shape | Y | N | Y | N | N | Y | N | 2015 |
| GA, Adaptive FL | A. Bakdi [109] | N | Trap shape | Y | N | Y | N | N | Y | N | 2017 |
| GA/ A* | K. Jose [110] | Y | Arbitrary | Y | N | Y | N | Y | Y | Y | 2016 |
| PSO/ NPSO | E. Masehian [9] | N | Arbitrary | Y | N | Y | N | N | Y | Y | 2010 |
| PSO | Y. Zhang [87] | N | Arbitrary | Y | N | Y | N | N | Y | N | 2013 |
| | Y. Hao, [88] | N | Arbitrary | Y | Y | Y | N | N | Y | N | 2007 |
| | Y. Wang [89] | Y | Circle | Y | Y | Y | N | N | Y | N | 2009 |
| PSO– Fuzzy, Lyapunov | K.D. Sharma [90] | N | Rectangle | Y | N | Y | N | N | Y | N | 2006 |
| Stochastic PSO | C. Xin [91] | N | Various | Y | N | Y | N | N | Y | N | 2006 |
| Fuzzy, Darwinian, PSO | M.S. Couceiro [92] | N | Arbitrary | Y | Y | Y | N | Y | Y | Y | 2012 |
| IPSO/IGSA | P.K.Das [112] | N | Arbitrary | Y | Y | Y | N | Y | Y | Y | 2016 |
| A-RPSO | M. Dadgar [113] | N | Arbitrary | Y | N | Y | N | N | Y | N | 2016 |
| PSO | L. Hu [114] | N | Arbitrary | Y | N | Y | N | N | Y | N | 2017 |
| ACO | B. Englot [93] | N | Rectangle | Y | N | Y | N | N | Y | N | 2011 |
| | R. Iser [94] | N | Various | Y | N | Y | N | N | Y | N | 2010 |
| | X. Chen [95] | N | Various | Y | N | Y | N | N | Y | N | 2013 |
| ACO- Fuzzy | M.A.P. Garcia [96] [10] | N | Rectangle | Y | N | Y | N | N | Y | Y | 2007 2009 |
| Improved ACO | J. Bai [97] | N | Circle | Y | N | Y | N | N | Y | N | 2011 |
| PFM- ACO | D. Zhao [98] | N | Rectangle U shape | Y | N | Y | N | Y | Y | N | 2006 |
| ACO | C. Luo [115] | N | Rectangle | Y | N | Y | N | N | Y | N | 2016 |
| ACO/A* | H. Wang [116] | N | Various | Y | N | Y | N | N | Y | N | 2016 |
| ACO | R. Uriol[117] | N | Various | Y | N | Y | N | N | Y | N | 2017 |
| New PF | S.S. Ge [119] | N | Point | N | Y | N | Y | Y | Y | N | 2002 |
| | L.Huang [120] | N | Circle | Y | N | N | Y | Y | Y | N | 2009 |
| | L.Huang [121] | N | Circle | Y | N | N | Y | Y | Y | N | 2012 |
| | L.Valbuena [122] | Y | Cube | Y | N | Y | N | N | Y | Y | 2012 |
| GA-FL, GA-NN, PFM | N. B. Hui [123] | Y | Circle | N | Y | N | Y | Y | Y | Y | 2009 |
| PFM/ ISS | M. Guerra [100] | Y | Various | Y | N | Y | N | N | Y | N | 2016 |
| ePFC | A. C. Woods [125] | Y | Various | Y | N | Y | N | N | Y | Y | 2017 |
| PFM | N. Malone [126] | N | Various | Y | Y | Y | N | N | Y | N | 2017 |
| Predict PFM | Y. Rasekhipour [127] | Y | Various | Y | N | Y | N | Y | Y | N | 2017 |

**Table B.1:** Applications of NN-FL-GA-PSO-ACO-PFM for the robot path planning.

# Bibliography

[1] P. K. Mohanty and D. R. Parhih. *Controlling the motion of an autonomous mobile robot using various techniques: a review.* Journal of Advance Mechanical Engineering, 4:24–39, 2013.

[2] F. Kendoul. *Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems.* Journal of Field Robotics, 29(2):315–378, 2012.

[3] D. Janglova. *Neural networks in mobile robot motion.* International Journal of Advanced Robotic Systems, 1:15–22, 2004.

[4] M. K. Singh and D. R. Parhi. *Intelligent neuron controller for navigation of mobile robot.* In International Conference on Advances in Computing, Communication and Control, pages 123–128, Mumbai, Maharashtra, India, 2009.

[5] H. Chang and T. Jin. *Command fusion based fuzzy controller design for moving obstacle avoidance of mobile robot.* Future Information Communication Technology and Applications, Lecture Notes in Electrical Engineering, pages 905–913, 2013.

[6] D. R. Parhi. *Neuro-fuzzy navigation technique for control of mobile Robots.* Intech, pages 409–4324, 2008.

[7] A. Zhu and S. X. Yang. *An adaptive Neuro-fuzzy controller for robot navigation.* Recent Advances in Intelligent Control Systems, pages 277–307, 2009.

[8] B. K Oleiwi, R. Jarrah, H. Roth, and B. I. Kazem. *Multi objective optimization of trajectory planning of non-holonomic mobile robot in*

*dynamic environment using enhanced GA by fuzzy motion control and A**. Neural networks and artificial intelligence, communications in computer and information science, 440:34–49, 2014.

[9] E. Masehian and D. Sedighizadeh. *Multi-objective PSO and NPSO based algorithms for robot path planning.* Advances in Electrical and Computer Engineering, 10 (4):69–76, 2010.

[10] M. A. P. Garcia, O. Montiel, O. Castillo, R. Sepulveda, and P. Melin. *Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation.* Advances in Soft Computing, 9:1102–1110, 2009.

[11] K. M. Passino and S. Yurkovich. *Fuzzy Control.* Addison Wesley Longman, Inc., 1998.

[12] F. Valdez, P. Meli, and O. Castillo. *A survey on nature-inspired optimization algorithms with fuzzy logic for dynamic parameter adaptation.* Expert Systems with Applications, 41:6459–6466, 2014.

[13] R. Siegwart and I. R. Nourbakhsh. *Introduction to autonomous mobile robot.* Massachusetts Institute of Technology press, Cambridge, U.S.A, 2004.

[14] M. G. Mohanan and A. Salgoankar. *A survey of robotic motion planning in dynamic environments.* Robotics and Autonomous Systems, 2017.

[15] F. Ingrand and M. Ghallab. *Deliberation for autonomous robots: A survey.* Artificial Intelligence, 247:10–47, 2017.

[16] A. Kaff, D. Martin, F. Garcia, A. de la Escalera, and J. A. Armingol. *Survey of computer vision algorithms and applications for unmanned aerial vehicles.* Expert Systems With Applications, 92:447–463, 2017.

[17] Y. Q. Qin, D. B. Sun, N. Li, and Y. G. Cen. *Path planning for mobile robot using the particle swarm optimization with mutation operator.* In International Conference on Machine learning and Cybernetics, pages 2473–2478, Shanghai, China, 2004.

[18] P. Raja and S. Pugazhenthi. *Heuristic approaches in robot path planning: a survey.* International Journal of Physical Sciences, 7(9):1314–1320, 2012.

[19] H. Zhang, J. Butzke, and M. Likhachev. *Combining global and local planning with guarantees on completeness.* In IEEE International Conference on Robotics and Automation, pages 4500–4506, St Paul, USA, 2012.

[20] L. C. Wang, L. S. Yong, and M. H. A. Jung. *Hybrid of global path planning and local navigation implemented on a mobile robot in indoor environment.* In IEEE International Symposium on Intelligent Control, pages 821–826, Mexico, Mexico, 2001.

[21] Z. Bi, Y. Yimin, and Y. Wei. *Hierarchical path planning approach for mobile robot navigation under the dynamic environment.* In IEEE Conference on Industrial Informatics, pages 372–376, Daejeon, Korea, 2008.

[22] E. Masehian and D. Sedighizadeh. *Classic and heuristic approaches in robot motion planning-a chronological review.* Proceeding of World Academy of Science, Engineering and Technology, pages 101–106, 2007.

[23] J. Rosell and P. Iniguez. *Path planning using harmonic functions and probabilistic cell decomposition.* In International Conference on Robotics and Automation (ICRA), pages 1803–1808, Barcelona, Spain, 2005.

[24] M. Seda. *Roadmap method vs. cell decomposition in robot motion planning.* In International Conference on Signal Processing, Robotics and automation, pages 1803–1808, Greece, 2005.

[25] F. A. Cosio and M. A. P. Castaneda. *Autonomous robot navigation using adaptive potential fields.* Mathematical and Computer Modeling, 40:1141–1156, 2004.

[26] J. Sfei, M. Saad, and H. S. Hassane. *An improved artificial potential field approach to real-time mobile robot path planning in an unknown environment.* In IEEE International Symposium on Robotic and Sensors Environments, pages 208–213, Montreal, Canada, 2011.

[27] S. K. Pradhan, D. R. Parhi, A. K. Panda, and R. K. Behera. *Potential field method to navigate several mobile robots.* Application Intelligence, 25:321–333, 2006.

[28] Y. Kang, M. Lee, C. Kim, S. Yoon, and C. Noh. *A study of cluster robots line formatted navigation using potential field method.* In IEEE International Conference on Mechatronics and Automation, pages 1723–1728, Beijing, China, 2011.

[29] D. H. Kim. *Escaping route method for a trap situation in local path planning.* International Journal of Control, Automation and Systems, pages 495–500, 2009.

[30] D. H. Kim and S. Shin. *Local path planning using a new artificial potential function configuration and its analytical design guidelines.* Advanced Robotics, 20:115–135, 2006.

[31] L. Xin, Y. Yin, and C. J. Lin. *A new potential field method for mobile robot path planning in the dynamic environment.* Asian Journal of Control, 11 (2):214–225, 2009.

[32] Q. Zhang, S. Yue, Q. Yin, and Y. Zha. *Dynamic obstacle-avoiding path planning for robots based on modified potential field method.* Intelligent Computing Theories and Technology, 7996:332–342, 2013.

[33] N. N. Singh, A. Chatterjee, and A. Rakshit. *A two-layered subgoal based mobile robot navigation algorithm with vision system and IR sensors.* Emerging Research in Artificial Intelligence and Computational Intelligence, Communications in Computer and Information Science, 237:325–334, 2011.

[34] H. Liu, W. Wan, and H. Zha. *A study of cluster robots line formatted navigation using potential field method.* In IEEE International Conference on Robotics and Automation (ICRA), pages 994–1001, Anchorage, Alaska, USA, 2010.

[35] S. Candido, Y. T. Kim, and S. Hutchinson. *An improved hierarchical motion planner for humanoid robots.* In IEEE-RAS International Conference on Humanoid Robots, pages 654–661, Daejeon, Korea, 2010.

[36] J. Lee, O. Kwon, L. Zhang, and S. E. Yoon. *A selective retraction-based RRT planner for various environments.* IEEE Transaction on Robotics, 30 (4):1002–1011, 2014.

[37] S. Karaman and E. Frazzoli. *Sampling-based algorithms for optimal motion planning.* The International Journal of Robotics Research, 30 (7):846–894, 2011.

[38] M. Elbanhawi and M. Simic. *Sampling-based robot motion planning: a review.* IEEE Access, 2:56–77, 2014.

[39] Z. Yan and N. Jouandeau. *ACS-PRM: Adaptive cross sampling based probabilistic road-map for multi-robot motion planning.* Intelligent autonomous systems 12, advances in intelligent systems and computing, 193:843–851, 2013.

[40] M. T. Rantanen and M. Juhola. *A configuration deactivation algorithm for boosting probabilistic road-map planning of robots.* International Journal of Automation and Computing, 9 (2):155–164, 2012.

[41] A. N. Nazif, A. Davoodi, and P. Pasquier. *Multi-agent area coverage using a single query road-map: A swarm intelligence approach.* Advances in Practical Multi-Agent Systems, Studies in Computational Intelligence, 325:95–112, 2011.

[42] D. Hsum, J. C. Latombe, and H. Kurniawati. *On the probabilistic foundations of probabilistic Roadmap planning.* International Journal of Robotics Research, 25 (7):627–643, 2006.

[43] A. L. Ladd and L. Kavraki. *Measure theoretic analysis of probabilistic path planning.* IEEE Transactions on Robotics and Automation, 20 (2):229–242, 2004.

[44] C. Moon and W. Chung. *Kinodynamic planner Dual-Tree RRT (DT-RRT) for two-wheeled mobile robots using the rapidly exploring random tree.* IEEE Transaction on Industrial Electronics, 62 (2):1080–1090, 2015.

[45] S. Karaman, M. W. Walter, A. Perez, E. Frazzoli, and S. Teller. *Anytime motion planning using the RRT*.* In IEEE International Conference on Robotics and Automation (ICRA), pages 1478–1483, Shanghai, China, 2011.

[46] A. C. Murtra, E. Trulls, O. Sandoval, J. Perez-Ibarz, D. Vasquez, J. M. Mirats-Tur, M. Ferrer, and A. Sanfeliu. *Autonomous navigation for urban*

*service mobile robots.* In IEEE/RSJ international conference on intelligent robots and systems (IROS), pages 4141–4146, Taipei, Taiwan, 2010.

[47] E. Trulls, A. Corominas Murtra, J. Peez-Ibarz, and G. Ferrer. *Autonomous navigation for mobile service robots in urban Pedestrian environments.* Journal of Field Robotics, 28 (3):329–354, 2011.

[48] I. Ardiyanto and J. Miura. *Real-time navigation using randomized kinodynamic planning with arrival time field.* Robotics and Autonomous Systems, 60:1579–591, 2012.

[49] D. J. Webb and J. Berg. *Kinodynamic RRT*: asymptotically optimal motion planning for robots with linear dynamics.* In IEEE International Conference on Robotics and Automation (ICRA), pages 5054–5061, Karlsruhe, Germany, 2013.

[50] S. M. LaValle and J. J. Kuffner. *Randomized kinodynamic planning.* International Journal of Robotics Research, 20 (5):378–400, 2001.

[51] S. Karaman and E. Frazzoli. *Optimal kinodynamic motion planning using incremental sampling-based methods.* In IEEE Conference on Decision and Control, pages 7681–7687, Atlanta, Georgia, USA, 2010.

[52] B. Lau, C. Sprunk, and W. Burgard. *Efficient grid-based spatial representations for robot navigation in dynamic environments.* Robotics and Autonomous Systems, 61:1116–1130, 2013.

[53] B. Park, J. Choi, and W. K. Chung. *An efficient mobile robot path planning using hierarchical Road-map representation in indoor environment.* In IEEE International Conference on Robotics and Automation (ICRA), pages 180–186, Minnesota, USA, 2012.

[54] V. R. Desaraju and J. P. How. *Decentralized path planning for multi-Agent teams in complex environments using rapidly-exploring random trees.* In IEEE International Conference on Robotics and Automation (ICRA), pages 4956–4961, Shanghai, China, 2011.

[55] G. S. Aoude, B. D. Luders, D. S. Levine, and J. P. How. *Decentralized path planning for multi-agent teams in complex environments using rapidly-exploring random trees.* In IEEE International Conference on Robotics and Automation (ICRA), pages 6058–6063, Shanghai, China, 2011.

[56] S. Chamberland, E. Beaudry, L. Clavien, F. Kabanza, F. Michaud, and M. Lauriay. *Motion planning for an omnidirectional robot with steering constraints.* In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 4305–4310, Taipei, Taiwan, 2010.

[57] S. X. Yang and M. Meng. *An efficient neural network approach to dynamic robot motion planning.* Springer-Verlag Berlin Heidelberg, 138:143–148, 2000.

[58] E. A. Antonelo and B. Schrauwen. *Supervised learning of internal models for autonomous goal-oriented robot navigation using reservoir computing.* In IEEE International Conference on Robotics and Automation (ICRA), pages 2959–2964, Alaska, USA, 2010.

[59] S. H. Dezfoulian, D. Wu, and I. S. Ahmad. *A generalized neural network approach to mobile robot navigation and obstacle avoidance.* Intelligent Autonomous Systems, 12:25–42, 2013.

[60] D. R. Parh and M. K. Singh. *Real-time navigational control of mobile robots using an artificial neural network.* Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 223 (7):1713–1725, 2009.

[61] M. K. Singh and D. R. Parhi. *Path optimization of a mobile robot using an artificial neural network controller.* International Journal of Systems Science, 42 (1):107–120, 2011.

[62] M. A. Sagban and R. Dhaouadi. *Neural-based navigation of a differential-drive mobile robot.* In International Conference on Control Automation Robotics and Vision (ICARCV), pages 353–358, Guangzhou, China, 2012.

[63] I. Engedy and G. Horvath. *Artificial neural network based mobile robot navigation.* In IEEE International Symposium on Intelligent Signal Processing, pages 241–246, Budapest, Hungary, 2009.

[64] X. Chen and Y. Li. *Smooth formation navigation of multiple mobile robots for avoiding moving obstacles.* International Journal of Control, Automation, and Systems, 4 (4):466–479, 2006.

[65] T. S. Hong, D. Nakhaeinia, and B. Karasfi. *Application of fuzzy logic in mobile robot navigation.* Fuzzy Logic -Controls, Concepts, Theories and Applications, pages 21–36, 2012.

[66] V. M. Peri and D. Simon. *Fuzzy logic control for an autonomous robot.* Fuzzy Information Processing Society, Annual Meeting of the North American, pages 337–342, 2005.

[67] J. H. Lilly. *Evolution of a negative-rule fuzzy obstacle avoidance controller for an autonomous vehicle.* IEEE Transaction Fuzzy Systems, 15:718–728, 2007.

[68] A. Zhu and S. X. Yang. *A Fuzzy logic approach to reactive navigation of behavior-based mobile robots.* In IEEE International Conference on Robotics and Automation (ICRA), pages 5045–5050, New Orleans, USA, 2004.

[69] Y. Tanaka, Y. Ji, A. Yamashita, and H. Asama. *Fuzzy based traversability analysis for a mobile robot on rough terrain.* In IEEE International Conference on Robotics and Automation (ICRA), pages 3965–3970, Washington, USA, 2015.

[70] A. Foudil, F. Mohammed, E. Muhammed, H. Ramdane, A. M. Khalid, A. Mansour, and M. Hassan. *A hierarchical fuzzy control design for indoor mobile robot.* International Journal of Advanced Robotic Systems, pages 1–16, 2014.

[71] L. Li, M. Zhang, L. Guo, and Y. Zhao. *Stereo vision based obstacle avoidance path-planning for cross-country intelligent vehicle.* In IEEE International Conference on Fuzzy Systems and Knowledge Discovery, pages 463–467, Tianjin, China, 2009.

[72] C. H. Chao, B. Hsueh, M. Hsiao, S. Tsai, and T. Li. *Fuzzy target tracking and obstacle avoidance of mobile robots with a stereo vision system.* International Journal of Fuzzy Systems, 11 (3):183–191, 2009.

[73] G. Mester. *Obstacle avoidance and velocity control of mobile robots.* In International symposium on intelligent Systems and Interpretation, pages 1–5, Subotica, Serbia, 2008.

[74] T. Lee and C. Wu. *Fuzzy target tracking and obstacle avoidance of mobile robots with a stereo vision system.* Journal of Intelligent and Robotic Systems, 37 (2):177–191, 2003.

[75] R. Araujo. *Prune-Able Fuzzy ART Neural architecture for robot map learning and navigation in dynamic environments.* IEEE transactions on Neural Networks, 17 (5):1235–1249, 2006.

[76] M. M. Joshi and M. A. Zaveri. *Reactive navigation of autonomous mobile robot using neuron-fuzzy system.* International Journal of Robotics and Automation (IJRA), 2 (3):128–145, 2011.

[77] N. B. Hui, V. Mahendar, and D. K. Pratihar. *Time-optimal, collision-free navigation of a car-like mobile robot using neuron-fuzzy approaches.* Fuzzy Sets and Systems, 157:2171–2204, 2006.

[78] M. Alajlan, A. Koubaa, I. Chaari, H. Bennaceur, and A. Ammar. *Research on robot path planning based on fuzzy neural network and particle swarm optimization.* In Global path planning for mobile robots in large-scale grid environments using genetic algorithms, pages 1–8, Sousse, Tunisia, 2013.

[79] Y. Hu and S. X. Yang. *A knowledge based genetic algorithm for path planning of a mobile robot.* In IEEE international Conference on Robotics and Automation (ICRA), pages 4350–4355, New Orleans, LA, USA, 2004.

[80] S. X. Yang, Y. Hu, and M. Q. H. Meng. *Knowledge based GA for path planning of multiple mobile robots in dynamic environments.* In IEEE Conference on Robotics, Automation and Mechatronics, pages 1–6, Bangkok, Thailand, 2006.

[81] S. C. Yun, S. Parasuraman, and V. Ganapathy. *Dynamic path planning algorithm in mobile robot navigation.* In IEEE Symposium on Industrial Electronics and Applications, pages 364–369, Langkawi, Malaysia, 2011.

[82] B. Karim and Q. Zhu. *Genetic fuzzy logic control technique for a mobile robot tracking a moving target.* International Journal of Computer Science, 10(1):607–613, 2013.

[83] H. C. Huangand C. C. Tsai. *Global path planning for autonomous robot navigation using hybrid meta heuristic GA-PSO algorithm.* In IEEE SICE Annual Conference, pages 1338–1343, Tokyo, Japan, 2011.

[84] S. M. R. Farshchi, S. A. N. Hoseini, and F. Mohammadi. *A novel implementation of G-fuzzy logic controller algorithm on mobile robot motion planning problem.* Computer and Information Science, 4(2):102–114, 2011.

[85] A. K. Karami and M. Hasanzadeh. *An adaptive genetic algorithm for robot motion planning in 2D complex environments.* Computers and Electrical Engineering, 43:1–13, 2015.

[86] C. Yang and D. Simon. *A new particle swarm optimization technique.* In IEEE International Conference on Systems Engineering, pages 164–169, Las Vegas, Nevada, USA, 2005.

[87] Y. Zhang, D. Gong, and J. Zhang. *Robot path planning in uncertain environment using multi-objective particle swarm optimization.* Neurocomputing, 103:172–185, 2013.

[88] Y. Hao, W. Zu, and Y. Zhao. *Real-time obstacle avoidance method based on polar coordination particle swarm optimization in dynamic environment.* In IEEE Conference on Industrial Electronics and Applications, pages 1612–1617, Harbin, China, 2007.

[89] Y. Wang, P. Chen, and Y. Jin. *Trajectory planning for an unmanned ground vehicle group using augmented particle swarm optimization in a dynamic environment.* In IEEE International Conference on Systems, Man, and Cybernetics, pages 4341–4346, San Antonio, TX, USA, 2009.

[90] K. D. Sharma, A. Chatterjee, and A. Rakshit. *A PSO-Lyapunov hybrid stable adaptive fuzzy tracking control approach for vision-based robot navigation.* Neurocomputing, 61:1908–1914, 2012.

[91] C. Xi and L. Yangmin. *Smooth path planning of a mobile robot using stochastic particle swarm optimization.* IEEE on Mechatronics and Automation, 61:1722–1727, 2006.

[92] M. S. Couceiro, J. A. T. Machado, R. P. Rocha, and N. M. F. Ferreira. *A fuzzified systematic adjustment of the robotic Darwinian PSO.* Robotics and Autonomous Systems, 60:1722–1727, 2006.

[93] Y. Wang, P. Chen, and Y. Jin. *Multi-goal feasible path planning using ant colony optimization.* In IEEE International Conference on Robotics and Automation (ICRA), pages 2255–2260, Shanghai, China, 2011.

[94] R. Iser and F. M. Wahl. *AntSLAM: global map optimization using swarm intelligence.* In IEEE International Conference on Robotics and Automation (ICRA), pages 265–272, Alaska, USA, 2010.

[95] X. Chen, Y. Kong, X. Fang, and Q. Wu. *A fast two-stage ACO algorithm for robotic path planning.* Neural Computer and Application, 22:313–319, 2013.

[96] M. A. P. Garcia, O. Montiel, O. Castillo, and R. Sepulveda. *Optimal path planning for autonomous mobile robot navigation using ant colony optimization and a fuzzy cost function evaluation.* Advances in Soft Computing, 41:790–798, 2007.

[97] J. Bai, L. Chen, H. Jin, R. Chen, and H. Mao. *Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation.* Lecture Notes in Electrical Engineering, 125:141–146, 2012.

[98] D. Zhao and J. Yi. *Robot planning with artificial potential field guided ant colony optimization algorithm.* Lecture Notes in Computer Science, 4222:222–231, 2006.

[99] D. L. Cruz and W. Yu. *Path planning of multi-agent systems in unknown environment with neural kernel smoothing and reinforcement learning.* Neurocomputing, 233:34–42, 2017.

[100] M. Guerra, D. Efimov, G. Zheng, and W. Perruquetti. *Avoiding local minima in the potential field method using input-to-statestability.* Control Engineering Practice, 55:174–184, 2016.

[101] A. M. Zou, Z. G. Hou, S. Y. Fu, and M. Tan. *Neural networks for mobile robot navigation: a survey.* Springer-Verlag Berlin Heidelberg, pages 1218–1226, 2006.

[102] Q. Q. Wu, Y. J. Lu, and X. Q. Liu. *Artificial neural network based mobile robot navigation.* In IEEE International Computer Conference on Wavelet Active Media Technology and Information Processing, pages 114–118, Chengdu, China, 2017.

[103] M. S. Masmoudi, N. Krichen, M. Masmoudi, and N. Derbel. *Fuzzy logic controllers design for omni-directional mobile robot navigation.* Journal of Intelligent and Robotic Systems, 49:901–919, 2016.

[104] C. Hong, C. W. Park, and J. H. Kim. *Evolutionary dual rule-based fuzzy path planner for omni-directional mobile robot.* Fuzzy Systems, 49:767–774, 2016.

[105] N. Kumar, M. Takacs, and Z. Vamossy. *Robot navigation in unknown environment using fuzzy logic.* In IEEE International Symposium on Applied Machine Intelligence and Informatics, pages 279–283, Herlany, Slovakia, 2017.

[106] S. K. Pradhan, D. R. Parhi, and A. K. Panda. *Neuro-fuzzy technique for navigation of multiple mobile robots.* Fuzzy Optimal Decision Making, pages 255–288, 2006.

[107] Y. Gou, W. Wang, and S. Wu. *Research on robot path planning based on fuzzy neural network and particle swarm optimization.* In IEEE Conference on Control And Decision Conference, pages 2146–2150, Chongqing, China, 2017.

[108] Z. Shi, H. Zhang, J. Zhou, and J. Wei. *An adaptive path planning based on improved Fuzzy Neural network for multi-robot systems.* Handbook of Research on Design, Control, and Modeling of Swarm Robotics, pages 319–343, 2016.

[109] A. Bakdi, A. Hentout, H. Boutami, A. Maoudj, O. Hachour, and B. Bouzouia. *Optimal path planning and execution for mobile robots using genetic algorithm and adaptive fuzzy logic control.* Robotics and Autonomous Systems, 89:95–109, 2017.

[110] K. Jose and D. K. Pratihar. *Task allocation and collision-free path planning of centralized multi-robots system for industrial plant inspection using heuristic methods.* Robotics and Autonomous Systems, 80:34–42, 2016.

[111] I. Chaari, A. Kouba, H. Bennaceur, A. Ammar, M. Alajlan, and H. Youssef. *Design and performance analysis of global path planning techniques for autonomous mobile robots in grid environments.* International Journal of Advanced Robotic Systems, 14(2):1–15, 2017.

[112] P. K. Das, H. S. Behera, and B. K. Panigrahi. *A hybridization of an improved particle swarm optimization and gravitational search algorithm*

*for multi-robot path planning.* Swarm and Evolutionary Computation, 28:14–28, 2016.

[113] M. Dadgar, S. Jafari, and A. Hamzeh. *A PSO-based multi-robot cooperation method for target searching in unknown environments.* Neurocomputing, 177:62–74, 2016.

[114] L. Hu, W. Naeem, E. Rajabally, and I. Salter. *COLREGs-compliant path planning for autonomous surface Vehicles: a multi-objective optimization approach.* IFAC-PapersOnLine, 50 (1):13662–13667, 2017.

[115] C. Luo, Y. Xiao, S. X. Yang, and G. E. Jan. *Improving vehicle navigation by a heading-enabled ACO approach.* In IEEE World Automation Congress, pages 1–6, Rio Grande, Puerto Rico, 2016.

[116] H. Wang, B. Yan, and X. Li. *On optimal path planning for UAV based patrolling in complex 3D topographies.* In IEEE International Conference on Information and Automation, pages 986–990, Ningbo, China, 2016.

[117] R. Uriol and A. Moran. *Mobile robot path planning in complex environments using ant colony optimization algorithm.* In IEEE International Conference on Control, Automation and Robotics, pages 15–21, Nagoya, Japan, 2017.

[118] F. Tan, J. Yang, J. Huang, T. Jia, W. Chen, and J. Wang. *A navigation system for family indoor monitor mobile robot.* In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 5978–5983, Taipei, Taiwan, 2010.

[119] S. S. Ge and Y. J. Cui. *Dynamic motion planning for mobile robots using potential field method.* Autonomous Robots, 13:207–222, 2002.

[120] L. Huang. *A Potential field approach for controlling a mobile robot to track a moving target.* In IEEE International Symposium on Intelligent Control, pages 65–70, Singapore, Singapore, 2007.

[121] L. Huang. *Velocity planning for a mobile robot to track a moving target a potential field approach.* Robotics and Autonomous Systems, 57:55–63, 2009.

[122] L. Valbuena and H. G. Tanner. *Hybrid potential field based control of differential drive mobile robots.* Journal of Intelligent and Robotic Systems, 68:307–322, 2012.

[123] N. B. Hui and D. K. Pratihar. *A comparative study on some navigation schemes of a real robot tackling moving obstacles.* Robotics and Computer-Integrated Manufacturing, 25:810–828, 2009.

[124] H. Chiang, N. Malone, K. Lesser, M. Oishi, and L. Tapia. *Path-guided artificial potential fields with stochastic reachable sets for motion planning in highly dynamic environments.* In IEEE International Conference on Robotics and Automation (ICRA), pages 2347–2354, Washington, USA, 2015.

[125] A. C. Woods and H. M. La. *A novel potential field controller for use on aerial robots.* IEEE Transactions on Systems, Man, and Cybernetics: Systems, PP(99):1–12, 2017.

[126] N. Malone, H. Chiang, K. Lesse, M. Oishi, and L. Tapia. *Hybrid dynamic moving obstacle avoidance using a stochastic reachable set-based potential field.* IEEE Transactions on Robotics, 33 (5):1124–1138, 2017.

[127] Y. Rasekhipour, A. Khajepour, S. Chen, and B. Litkouhi. *A potential field-based model predictive path-planning controller for autonomous road vehicles.* IEEE Transactions on Intelligent Transportation Systems, 18 (5):1255–1267, 2017.

[128] K. Zhou, A. Leck Jensen, C. G. Sorensen, P. Busato, and D. D. Bothtis. *Agricultural operations planning in fields with multiple obstacle areas.* Computers and Electronics in Agriculture, 109:12–22, 2014.

[129] A. Chevalier, M. V. D. Bossche, C. Copot, and R. De Keyser. *Formation control strategies for emulation of field covering.* In International Conference on System Theory, Control and Computing, pages 526–531, Sinaia, Romania, 2014.

[130] A. G. Cerezo, A. Mandow, J. Martinez, J. G. de Gabriel, J. Morales, A. Cruz, A. Reina, and J. Seron. *Development of ALACRANE: a mobile robotic assistance for exploration and rescue missions.* In International

Workshop on Safety, Security and Rescue Robotics, pages 1–6, Rome, Italy, 2007.

[131] K. Zhou, A. Leck Jensen, C. G. Sorensen, P. Busato, and D. D. Bothtis. *Autonomous field measurement in outdoor areas using a mobile robot with RTK GNSS.* IFAC-PapersOnLine, 48 (4):480–485, 2015.

[132] I. Trinks. *Nondestructive subsurface mapping in field archeology.* Encyclopedia of Global Archeology, Springer Science Business Media: New York, 2014.

[133] J. R. Henderson, J. M. Conrad, and C. Pavlich. *Using a CAN bus for control of an all-terrain vehicle.* In IEEE Southeast Conference, pages 1–5, Lexington, USA, 2014.

[134] A. Cortner, J. M. Conrad, and N. A. BouSaba. *Components of an autonomous all-terrain vehicle.* In Proceedings of IEEE Southeastcon, pages 1–5, Orlando, Florida, USA, 2012.

[135] L. Bascetta, G. A. Magnani, P. Rocco, and A. M. Zanchettin. *Design and implementation of the low-level control system of an all-terrain mobile robot.* In International Conference on Advanced Robotics (ICAR), pages 1–6, Munich, Germany, 2009.

[136] M. Linderoth and K. Soltesz. *Nonlinear lateral control strategy for nonholonomic vehicles.* In American Control Conference, pages 3219–3224, Seattle, USA, 2008.

[137] M. Heinzelmann, D. Jordan, and C. Murer. *Amiternum and the upper Aterno valley: a sabine-roman town and its territory.* Journal of Roman Archeology, 23:55–83, 2010.

[138] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser. *Heuristic approaches in robot path planning: a survey.* Robotics and Autonomous Systems, 86:13–28, 2016.

[139] D. Cagigas. *Hierarchical $D^*$ algorithm with materialization of costs for robot path planning.* Robotics and Autonomous Systems, 5 (2):190–208, 2005.

[140] N. Ghita and M. Kloetzer. *Trajectory planning for a car-like robot by environment abstraction.* Robotics and Autonomous Systemsy, 60:609–619, 2012.

[141] H. T. Chiang, N. Malone, K. Lesser, M. Oishi, and L. Tapia. *Path guided artificial potential fields with stochastic reachable sets.* In IEEE International Conference on Robotics and Automation (ICRA), pages 2347–2354, Washington, USA, 2015.

[142] A. N. Nazif, A. Davoodi, and P. Pasquier. *Multi-agent area coverage using a single query road map: A swarm intelligence approach.* Advances in Practical Multi-Agent Systems, Studies in Computational Intelligence, 325:95–112, 2011.

[143] N. N. Singh, A. Chatterjee, and A. Rakshit. *A two-layered subgoal based mobile robot navigation algorithm with vision system and IR sensors.* Emerging Research in Artificial Intelligence and Computational Intelligence, Communications in Computer and Information Science, 237:325–334, 2011.

[144] H. Duan and L. Huang. *Imperialist competitive algorithm optimized artificial neural networks for UCAV global path planning.* Neurocomputing, 125:166–171, 2014.

[145] B. Ster. *An integrated learning approach to environment modeling in mobile robot navigation.* Neurocomputing, 57:215–238, 2014.

[146] R. A. Jarrah, A. Shahzad, and H. Roth. *Path planning and motion coordination for multi-robots system using probabilistic Neuro-fuzzy.* IFAC-Papers OnLine, 48 (10):46–51, 2015.

[147] M. Davoodi, F. Panahi, A. Mohades, and S. N. Hashemi. *Clear and smooth path planning.* Applied Soft Computing, 32:568–579, 2015.

[148] Y. Zhang, D. Gong, and J. Zhang. *Robot path planning in uncertain environment using multi-objective particle swarm optimization.* Neurocomputing, 103:172–185, 2013.

[149] A. Kaveh and K. Laknejadi. *A novel hybrid charge system search and particle swarm optimization method for multi-objective optimization.* Expert Systems with Applications, 38:15475–15488, 2011.

[150] X.C. Lai and S.Ge. *Hierarchical incremental path Planning and situation dependent optimized dynamic motion planning considering accelerations.* IEEE Transactions on Systems, Man, and Cybernetics—part B: Cybernetics, 37 (6):1514–1554, 2007.

[151] L. Zou, Q. Guo, X. Xu, and H. Fu. *A hierarchical path planning approach based on A* and least squares policy iteration for mobile robots.* Neurocomputing, 170:257–266, 2015.

[152] X. Yang, M. Moallem, and R. V. Patel. *A layered goal-oriented fuzzy motion planning strategy for mobile robot navigation.* IEEE Transactions on Systems, Man, and Cybernetics-part B: Cybernetics, 37 (6):1514–1554, 2007.

[153] R. Kala, A. Shukla, and R. Tiwari. *Robotic path planning in static environment using hierarchical multi-neuron heuristic search and probability based fitness.* Neurocomputing, 74:2314–2335, 2011.

[154] A. Kaveh. *A combinational optimization problem: optimal generalized cycle bases.* Computer Methods in Applied Mechanics and Engineering, 20:39–51, 2009.

[155] A. Kaveh and V. Kalatjari. *Topology optimization of trusses using genetic algorithm, force method and graph theory.* International Journal for Numerical Methods in Engineering, 20:771–791, 2003.

[156] Y. Li and Y. Guan. *A path planning method to robot soccer based on Dijkstra's algorithm.* Advances in Intelligent and Soft Computing, 149:89–95, 2012.

[157] *http://www.mediafire.com/file/konp8j11ov3c1b8.* Last access on 10 January, 2018.

[158] C. A. Coello, C.Pulido, and G. T. Lechuga. *Handling multiple objectives with particle swarm optimization.* IEEE Transactions on Evolutionary Computation, 8(3):256–279, 2004.

[159] Y. Wu. Zhang and L. Wang. *UCAV path planning by fitness-scaling adaptive chaotic particle swarm optimization.* Mathematical Problems in Engineering, 2013:1–9, 2013.

[160] X. S. Yang, S. Deb, and S. Fong. *Accelerated particle swarm optimization and support vector machine for business optimization and applications.* IEEE Transactions on Evolutionary Computation, 136:53–66, 2011.

[161] M. Barbehenn. *A Note on the complexity of Dijkstra's algorithm for graphs with weighted vertices.* IEEE Transasctions on Computers, 47 (2):263, 1998.

[162] M. T. Jensen. *Reducing the run-time complexity of multi-objective EAs: The NSGA-II.* IEEE Transasctions on Evolutionary Computation, 7 (5):503–515, 2015.

[163] A. Hernandez, H. Murcia, C. Copot, and R. De Keyser. *Towards the development of a smart flying sensor: illustration in the field of precision agriculture.* Sensors (Basel), 15(7):16688–16709, 2015.

[164] A. Hernandez, C. Copot, J. D. Cerquera, H. Murcia, and R. De Keyser. *Formation control of UGVs using an UAV as remote vision sensor.* IFAC Proceedings Volumes, 47 (3):11872–11877, 2014.

[165] N. Dijkshoorn. *Simultaneous localization and mapping with the Ar.Drone.* Master's thesis, University of Amsterdam, Amsterdam, the Netherland, 2012.

[166] G. Dullerud. *Modeling, identification and control of a quad-rotor drone using low-resolution sensing.* Master's thesis, University of Illinois at Urbana-Champaign, Champaign, IL, USA, 2012.

[167] P. Bouffard. *On-board model predictive control of a quadrotor helicopter: design, implementation, and experiments.* Master's thesis, University of California, Berkeley, CA, USA, 2012.

[168] R. Ritz, M. Muller, M. Hehn, and R. DAndrea. *Cooperative quadrocopter ball throwing and catching.* In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 4972–4978, Algarve, Portugal, 2012.

[169] H. Alvarez, L. M. Paz, and J. Sturm. *Collision avoidance for quadrotors with a monocular camera, experimental robotics.* Tractsaction in Advanced Robotics, 109:195–209, 2015.

[170] T. T. Mac, C. Copot, T. D. Tran, and R. De Keyser. *A hierarchical global path planning for mobile robot based on multi-objective particle swarm optimization.* Applied Soft Computing, 59:68–76, 2017.

[171] H. S. Hassane J. Sfeir, M. Saad. *An improved artificial potential field approach to real-time mobile robot path planning in an unknown environment.* In IEEE International Symposium on Robotic and Sensors Environments, pages 208–213, Montreal, Canada, 2011.

[172] S. S. Ge and Y. J. Cui. *New potential functions for mobile robot path planning.* Tractsaction in Advanced Robotics, 16 (5):615–620, 2010.

[173] F. A. Cosfo and M. A. P. Castaida. *Autonomous robot navigation using adaptive potential fields.* Mathematical and Computer Modelling, 40:1141–1156, 2004.

[174] L. Valbuena. *Hybrid potential field based control of differential drive mobile robots.* Intelligent Robot Systems, 68:307–322, 2012.

[175] S. Scherer, S. Singh, L. Chamberlain, and M. Elgersma. *Flying fast and low among obstacles, methodology and experiments.* International Journal of Robotics Research, 27 (5):549–574, 2008.

[176] G. Li, A. Yamashita, H. Asama, and Y. Tamura. *An efficient improved artificial potential field based regression search method for robot path planning.* In IEEE International Conference on Mechatronics and Automation, pages 1227–1232, Chengdu, China, 2012.

[177] T. T. Mac, C. Copot, A. Hernandez, and R. De Keyser. *Improved potential field method for unknown obstacle avoidance using UAV in indoor environment.* In IEEE International Symposium on Applied Machine Intelligence and Informatics, pages 345–350, Herlany, Slovakia, 2016.

[178] J. Li and Y. Li. *Dynamic analysis and PID control for a quadrotor.* In IEEE International Conference on Mechatronics and Automation, pages 573–578, Beijing, China, 2011.

[179] Y. Song, B. Xian, Y. Zhang, X. Jiang, and X. Zhang. *Towards autonomous control of quadrotor unmanned aerial vehicles in a GPS-denied urban*

*area via laser ranger finder.* Optik - International Journal for Light and Electron Optics, 126 (23):3877–3882, 2015.

[180] J. Engel, J. Sturm, and D. Cremers. *Scale-aware navigation of a low-cost quadrocopter with a monocular camera.* Robotics and Autonomous Systems, 62 (11):1646–1656, 2014.

[181] L. Ljung. *System identication: theory for the user.* Prentice-Hall, 2007.

[182] *https://docs.opencv.org/2.4/modules/calib3d/.* Last access on 20 March, 2018.

[183] R. De Keyser and C. M. Ionescu. *FRTool: a frequency response tool for CACSD in Matlab.* In IEEE Conference on Computer Aided Control Systems Design, pages 2275–2280, Munich, Germany, 2006.

[184] F. Kendoul. *Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems.* Journal of Field Robotics, 29 (2):315–378, 2012.

[185] J. Sprinkle, J. Eklund, and S. Sastry. *Deciding to land a UAV safely in real time.* In IEEE Proceedings of the American Control Conference, pages 3506–3511, Portland, USA, 2005.

[186] S. Saripall and G. Sukhatme. *Landing a helicopter on a moving target.* In IEEE International Conference on Robotics and Automation, pages 203–2035, Rome, Italy, 2007.

[187] F. Alarcon, D. Santamaria, and A. Daniel. *UAV helicopter relative state estimation for autonomous landing on moving platforms in a GPS-denied scenario.* IFAC-PapersOnLine, 48 (9):37–42, 2015.

[188] W. Bagen, J. Hu, and Y. Xu. *A vision-based unmanned helicopter ship board landing system.* In IEEE International Congress on Image and Signal Processing, pages 203–2035, Tianjin, China, 2009.

[189] C. De Wagter and J. Mulder. *Towards vision-based UAV situation awareness.* In AIAA Conference on Guidance, Navigation, and Control, pages 1–16, San Francisco, California, USA, 2005.

[190] K. H. Hsia, S. F. Lien, and J. P. Su. *Height estimation via stereo vision system for unmanned helicopter autonomous landing.* In IEEE International Computer Communication Control and Automation Symposium, pages 257–260, Tainan, Taiwan, 2010.

[191] Y. H. Shin, S. Lee, and J. Seo. *Autonomous safe landing-area determination for rotorcraft UAVs using multiple IR-UWB radars.* Aerospace Science and Technology, 69:617–624, 2017.

[192] Y. Chong, Y. Cai, and Q. Chen. *Multi-resolution visual fiducial and assistant navigation system for unmanned aerial vehicle landing.* Aerospace Science and Technology, 67:249–256, 2017.

[193] Y. F. Shen, Z. U. Rahman, A. Krusienski, and J. Li. *A vision-based automatic safe landing-site detection system.* IEEE Transactions on Aerospace and Electronic Systems, 49(1):294–311, 2013.

[194] J. Seo and T. Walter. *Future dual-frequency GPS navigation system for intelligent air transportation under strong ionospheric scintillation.* IEEE Transactions on Intelligent Transportation Systems, 15 (5):2224–2236, 2014.

[195] S. Gao, L. Xue, Y. Zhong, and C. Gu. *Random weighting method for estimation of error characteristics in SINS/GPS/SAR integrated navigation system.* Aerospace Science and Technology, 46:22–29, 2015.

[196] J. Seo, T. Walter, and P. Enge. *Availability impact on GPS aviation due to strong ionospheric scintillation.* IEEE Transactions on Aerospace and Electronic Systems, 47 (3):1963–1973, 2011.

[197] J. A. G. Pulido, G. Pajares, S. Dormido, and J. M. De la Cruz. *Recognition of a landing platform for unmanned aerial vehicles by using computer vision-based techniques.* Expert Systems With Applications, 76:152–165, 2017.

[198] R. A. Pradeep and P. Radhakant. *Robust auto landing off fixed-wing UAVs using neuro-adaptive design.* Control Engineering Practice, 60:218–232, 2017.

[199] B. Herisse, T. Hamel, R. Mahony, and F. X. Russotto. *Landing a VTOL unmanned aerial vehicle on a moving platform using optical flow.* IEEE Transactions on Robotics, 28 (1):77–89, 2012.

[200] M. F. Lee, S. F. Su, J. W. Yeah, H. M. Huang, and J. Chen. *Autonomous landing system for aerial mobile robot cooperation.* In IEEE International Symposium on Soft Computing and Intelligent Systems, pages 1306–1311, Kitakyushu, Japan, 2014.

[201] A. Benini, M. J. Rutherford, and K. P. Valavanis. *Real-time, GPU-based pose estimation of a UAV for autonomous takeoff and landing.* In IEEE International Conference on Robotics and Automation (ICRA), pages 3463–3470, Stockholm, Sweden, 2016.

[202] *https://www.kickstarter.com/projects/pozyx/pozyx-accurate-indoor-positioning-for-arduino.* Last access on 10 January, 2018.

[203] P. Van de Voorde, S. Gautama, A. Momont, C. M. Ionescu, P. De Paepe, and N. Fraeyman. *The drone ambulance [A-UAS]: golden bullet or just a blank?* Last access on 10 January, 2018, 116:46–48, 2017.