IntechOpen

Motion Planning for Dynamic Agents

Edited by Zain Anwar Ali and Amber Israr

Motion Planning for Dynamic Agents

Edited by Zain Anwar Ali and Amber Israr

Published in London, United Kingdom

Motion Planning for Dynamic Agents
http://dx.doi.org/10.5772/intechopen.1000319
Edited by Zain Anwar Ali and Amber Israr

Contributors
Zain Anwar Ali, Amber Israr, Raza Hasan, Yasuhiko Kamiyama, Nohaidda Sariff, Zool Hilmi Ismail, Ahmad Shah Hizam Md Yasir, Denesh Sooriamoorthy, Puteri Nor Aznie Fahsyar Syed Mahadzir, Zaharuddeen Haruna, Muhammed Bashir Mu'azu, Abubakar Umar, Glory Okpowodu Ufuoma, Halil Utku Unlu, Dimitris Chaikalis, Vinicius Gonçalves, Anthony Tzes, G. Kassahun Berisha, Hafte Tkue, Yalemzerf Getnet

Notice

Statements and opinions expressed in the chapters are these of the individual contributors and not necessarily those of the editors or publisher. No responsibility is accepted for the accuracy of information contained in the published chapters. The publisher assumes no responsibility for any damage or injury to persons or property arising out of the use of any materials, instructions, methods or ideas contained in the book.

First published in London, United Kingdom, 2024 by IntechOpen
IntechOpen is the global imprint of INTECHOPEN LIMITED, registered in England and Wales, registration number: 11086078, 5 Princes Gate Court, London, SW7 2QJ, United Kingdom

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

Additional hard and PDF copies can be obtained from orders@intechopen.com

Motion Planning for Dynamic Agents
Edited by Zain Anwar Ali and Amber Israr
p. cm.
Print ISBN 978-0-85466-059-9
Online ISBN 978-0-85466-058-2
eBook (PDF) ISBN 978-0-85466-060-5

Meet the editors

Engr. Dr. Zain Anwar Ali obtained a Ph.D. in Control Theory and Control Engineering from Nanjing University of Aeronautics and Astronautics (NUAA), China. In 2017, he was selected as a highly talented foreign expert by the Ministry of China, Beijing. In 2018–2019, he received research funding from the Higher Education Commission (HEC), Pakistan, and began collaborating with several universities in China, including NUAA, Donghua University, Shanghai University, and Southeast University, with grants provided by the National Nature Science Foundation of China (NSFC). Dr. Ali is an associate professor and editor at Sir Syed University of Engineering and Technology (Syed UET), Pakistan. In 2020 and 2021, he completed his Post Doctor Tier-I in Control Engineering from Southeast University, Nanjing, China, and Post Doctor Tier-II in Systems Science from Beijing Normal University, China. In 2021, he rejoined Sir Syed UET as an associate professor and an editor of the SSUET research journal. In 2022, he joined Jiaying University, China as an associate professor and worked on the Smart Agriculture project. In late 2023, he joined the Centre for Ocean Energy Research (COER) at the National University of Ireland (NUI), Maynooth as a senior postdoctoral researcher on the Control Co-Design of Heterogeneous Arrays of Wave Energy Converter project.

Dr. Amber Israr is an accomplished and highly regarded associate professor in the Electronic Engineering Department, Sir Syed University of Engineering and Technology, Pakistan. With a wealth of expertise, she specializes in designing and analyzing computer communication networks, computer network security, real-time computer systems, and artificial intelligence. Her commitment to academic excellence is evident through her deep knowledge and understanding of diverse subjects within electronic engineering. Dr. Israr is recognized for her exceptional teaching skills, demonstrating a strong ethical foundation that serves as an example to her students. She adeptly employs various teaching strategies and techniques, ensuring an engaging and effective learning experience for her students. Dr. Israr obtained a Ph.D. in Electronic Engineering from Sir Syed University of Engineering and Technology with her thesis, ”Prediction-Based Greedy Routing Strategies for Wireless Networks of Moving Objects”.

Contents

Preface

Motion planning for dynamic agents is a broad study bridging the areas of robotics, control theory, and artificial intelligence. The speed at which these fields have advanced in recent years has opened the door to previously unimaginable possibilities for autonomous systems, ranging from flexible robotic platforms that interact with people in dynamic situations to self-driving vehicles that navigate dense metropolitan landscapes. The demand for effective and flexible motion planning algorithms for dynamic agents grows as the lines between the virtual and real worlds become less distinct.

This book guides readers on an extensive examination of motion planning for dynamic agents, focusing on the complexities of navigating uncertain and dynamic situations with entities. Autonomous cars unmanned aerial vehicles, mobile robots, and even humanoids are all included in the broad category of systems that are referred to as "dynamic agents." These agents work in dynamic environments where barriers move, environmental conditions vary, and interactions with other agents provide an extra degree of complexity, in contrast to classic motion planning situations where the environment is static.

The book focuses on the relationship between motion planning and artificial intelligence, emphasizing machine learning techniques that help agents become more skilled at making decisions by allowing them to gain experience. The most recent advancements in motion planning techniques for dynamic agents cover a significant portion of the book. Innovative approaches, such as cooperative motion planning, control barrier functions, and optimal trajectory tracking, are analyzed and shown with an emphasis on how well they may be applied in various situations.

Motion Planning for Dynamic Agents is a comprehensive guide for scholars, professionals, and learners who are interested in the exciting field of navigating dynamic agents in complex environments. With this book, we want to provide readers with a road map that will lead them through the mathematical foundations, real-world applications, and ethical aspects of motion planning for dynamic agents. This book seeks to be a beacon of hope, shedding light on the future in this fascinating and dynamic field as it continues to develop.

Zain Anwar Ali
Associate Professor,
School of Physics and Electronic Engineering,
JiaYing University,
Meizhou, Guangdong, China

Dr. Amber Israr
Associate Professor, EED,
Sir Syed University of Engineering and Technology,
Karachi, Pakistan

Chapter 1

Introductory Chapter: Motion Planning for Dynamic Agents

Zain Anwar Ali

1. Introduction

Dynamic agents enclose an extensive range of ventities covering mobile robots, autonomous vehicles, drones, unmanned aerial vehicles (UAVs), unmanned ground vehicles (UGVs), unmanned underwater vehicles (UUVs), and other objects in motion. An integral aspect in the development of these agents, especially the autonomous ones, is to devise a course of action enabling them to make their plans in various situations. Motion planning is the computational process of navigating the above-mentioned entities from one to another place while avoiding collisions, optimizing for assessment criteria, and adapting to varying conditions. It generates a feasible and safe trace to follow and enables efficient interactions between dynamic agents and their environment [1, 2]. This introductory chapter conceptualizes motion planning and brings insights into the various motion planning approaches for dynamic agents along with their applications, significance, and existing challenges that require future considerations and future potential. This chapter emphasizes the integral role of motion planning in shaping the future of robotics and autonomous systems.

Motion planning is comprised of two components namely path planning and trajectory planning. The path refers to a sequence of configuration vectors considering independent attributes like position, orientation, linear or angular velocity, steering angle, acceleration, and many others. A trajectory is an order of spatio-temporal states, feasible for agent dynamics, in free space. First, the trajectory planner generates an array of feasible and collision-free trajectories then, it selects an optimal one founded on cost function optimization. These components ensure safe and efficient motion, especially in complex, unknown, and ever-changing environments. Therefore, motion planning is a crucial aspect of robotics and automation [3]. Motion planning approaches are applied according to the application and its requirements for dynamic agents. Some widely employed approaches are sampling-based, optimization-based, model predictive control (MPC), bio-inspired, and artificial intelligence (AI)-based approaches as illustrated in **Figure 1**. These methods offer opportunities, however, face some restrictions [4].

The first category is sampling-based approaches that are computationally efficient and operative in high-dimensional spaces. These methods are further categorized as random and de-randomized or deterministic sampling-based approaches [5]. Algorithms that randomly sample the configuration space for generating feasible paths are referred to as random sampling-based methods. These are successful approaches for robotic motion planning problems. These aim to conduct a probabilistic search that examines the configuration space with independently and identically distributed random samples. This assessment is enabled through a collision

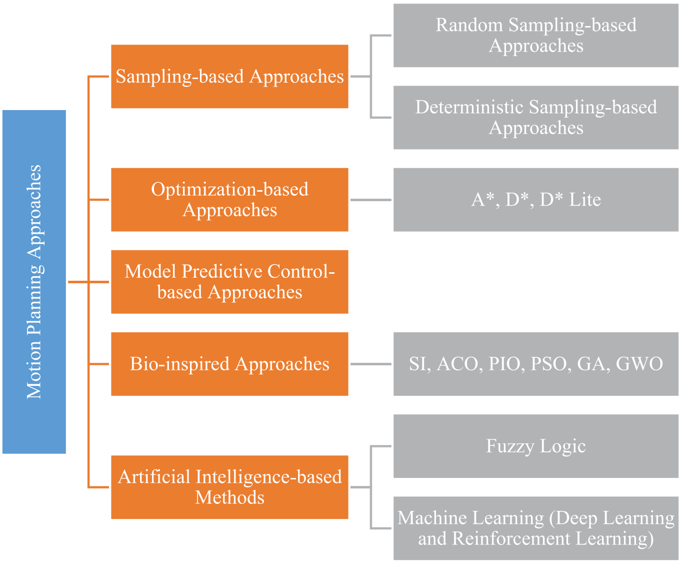

Motion Planning Approaches
Sampling-based Approaches
Random Sampling-based Approaches
Deterministic Sampling-based Approaches
Optimization-based Approaches
A*, D*, D* Lite
Model Predictive Control-based Approaches
Bio-inspired Approaches
SI, ACO, PIO, PSO, GA, GWO
Artificial Intelligence-based Methods
Fuzzy Logic
Machine Learning (Deep Learning and Reinforcement Learning)

Figure 1.
Classification of motion planning approaches.

detection module, referred to as a "Black Box" in motion planning. Probabilistic roadmaps (PRMs), rapidly exploring random trees (RRTs), its asymptotically optimal version RRT*, fast marching tree algorithm (FMT∗), closed-loop rapidly exploring random tree (CL-RRT), etc., are random sampling-based approaches [6]. On the other hand, deterministic sampling-based algorithms, for instance, physics-based motion planner (PMP). PMP remarkably simplifies the certification process, enables the employment of offline computation, and even in certain cases drastically makes various operations simple [7]. Next are optimization-based approaches that determine the optimal path by reducing a cost function. They are significant for applications where path quality is considered integral [8]. Optimization-based strategies include the A* search algorithm, dynamic A* (D*), and D* Lite. The A* algorithm, developed from a bidirectional search optimization strategy, is fast with a good heuristic and determines an optimal path. The D* algorithm maintains a partial, optimal cost map, and works in unbounded environments. It is memory and computationally efficient. The D* Lite algorithm combines heuristic and incremental search that covers multiple navigation tasks in a dynamic environment. It is easy to understand and analyze [9].

Model predictive control approaches are extensively used as optimal control methods for high-level trajectory resulting in safe and fast motion planning of robots. MPC is well-suited for systems with dynamic and stochastic constraints. It necessitates a system model for optimizing control over finite-time horizons and feasible trajectories recursively. It considers the updating of the environmental state during its planning process. Authors survey and highlight the perspectives of using MPC for motion planning and control of autonomous marine vehicles (AMVs) [10]. Their research shows that MPC is exceptionally successful in enhancing the guidance and control capabilities of AMVs through effective and practical planning. In multi-robot systems, existing motion planning algorithms often lack addressing for poor coordination and increasing low real-time performance. In such cases, model predictive contouring control (MPCC) allows separating the tracking accuracy and productivity for modifying productivity [11]. The MPCC allows every single robot to exchange other robots' predicted paths while generating collision-free motion parallelly.

Bio-inspired approaches draw inspiration from mechanisms observed in natural systems and the capabilities of dynamic agents to manage dynamic environment complexities [12]. Motion planning based on these approaches offers a pathway to more robust, adaptive, and efficient navigation strategies. Swarm intelligence (SI), particle swarm optimization (PSO), ant colony optimization (ACO), pigeon-inspired optimization (PIO), genetic algorithm (GA), and gray wolf optimizer (GWO) are some bio-inspired approaches. SI leverages the collective behavior of simple agent groups that mimic the coordination of animal swarms. All the robots in the swarm follow local rules leading to intelligent, emergent group behavior [13]. AI-based approaches learn motion policies from data and are gaining more attraction nowadays. These techniques have rapid computation abilities and are widely employed for online planning. AI considers motion planning as a sequential decision-making problem and leverages it with fuzzy logic and machine learning (ML) algorithms covering reinforcement learning (RL) and deep learning (DL) [14]. For instance, ML stimulates the learning behavior and then determines solutions for path planning for fast agent-based robust systems. A deep Q-network (DQN) algorithm is implemented that converges faster and more rapidly learns path-planning solutions for multi-robots [15]. **Table 1** displays the advantages of implementing the above-mentioned motion planning approaches.

The adoption of robotics and automation is increasing in various industries for performing multiple, dangerous, repetitive, and beyond human capabilities tasks. Robots, autonomous vehicles, and drones are deployed from healthcare to surveillance missions. Considering the surveillance missions, communication between a dynamic agent and a control base may be lost [19]. Implementing effective path planning guides the robot preventing it from damage and improving its exploration.

References	Type of dynamic robot	Applied approach	Environment	Opportunities
[9]	Autonomous vehicles	Nonlinear Model Predictive Control (NMPC)-based motion planning	Experiment	Shortens the prediction horizon. Ensures safety. Achieves improvement in lap time
[15]	Multi-robots	DQN	Simulation	Converges faster. More rapidly learns path-planning solutions
[16]	Unmanned aircraft systems (UAS)	MPC-RRT$^{\#}$	Simulation	Generates a conceivable trajectory satisfying dynamic constraints
[17]	Multiple UAVs	A*	Experiment	Improve success possibility. Generates shorter paths in less time
[18]	UAV	Hybrid PSO	Experiment	More feasible and effective in path planning

Table 1.
Opportunities of various motion planning approaches.

In the agriculture sector, autonomous robots and UAVs navigate fields for planting, crop monitoring, spraying fertilizers and pesticides, and even harvesting purposes. In the healthcare setting robots with precise motion planning assist in surgeries, patient care, medicine delivery, and rehabilitation. Considering the manufacturing industry, industrial and service robots implement motion planning for performing various tasks like assembly, pick and pack operations, navigation, and delivery in cluttered environments [20]. Besides these, motion planning enables analysis of various factors such as road conditions, traffic, and weather for self-driving cars. The paramount significance of motion planning is because of several compelling reasons such as safety, efficiency, and adaptability. Navigation of dynamic agents with collision avoidance, obstacle avoidance, and minimal risks ensures the safety of these agents as well as humans and the environment [21]. Motion planning makes autonomous vehicles efficient in generating optimal paths reducing energy consumption, travel time, and operational costs. Furthermore, it allows dynamic agents to adapt and respond in changing and unpredictable environments.

Motion planning faces several challenges due to real-time decision-making, high-dimensional state spaces, uncertainty and complexity, human-robot interactions, and huge data and processing requirements [22]. Quick decision-making for responding to the changing conditions in dynamic and complex environments may affect motion planning. Next is the high-dimensional state space in certain cases which makes feasible path planning computationally intensive. Additionally, uncertainty in sensor measurements as well as environment models must be addressed. More advanced and interconnected systems also increase the complexity of motion planning. Subsequently, environments with robots and humans coexist, therefore, consideration of human intentions and behavior is crucial for safe and natural human-robot interactions. Furthermore, some approaches, especially AI-based methods, require huge amounts of high-quality data for training algorithms [23]. However, collecting, processing, and analyzing data is expensive and difficult which restricts the application of these methods. On the other hand, some approaches lead to undesirable and unpredictable behaviors in complex environments, generating unsafe or suboptimal routes that may lead to accidents. All these challenges must be overcome to harness the full potential of autonomous systems.

The future of dynamic agents looks promising as advancement in research and technology continues to shape its development. Motion planning will play a central role in efficient transportation systems with the integration of advanced sensor technologies, AI, and V2X communication [24]. This will optimize traffic management reduce congestion and enhance public transportation with the coordination of autonomous vehicles. Advanced approaches will facilitate safe human-robot collaborations and interactions that will make collaborative robots or cobots and humanoids more prevalent in various industries. Ongoing research is concentrated on developing more reliable and robust algorithms, like deep reinforcement learning [25]. This will make inroads into motion planning enabling agents to adapt to more complex and uncertain environments while ensuring the safety of dynamic agents.

Author details

Zain Anwar Ali
School of Physics and Electronic Engineering, JiaYing University,
Meizhou, Guangdong, China

*Address all correspondence to: zainanwar86@hotmail.com

References

[1] Chen M, Herbert SL, Hu H, Ye P, Fisac JF, Bansal S, et al. Fastrack: A modular framework for real-time motion planning and guaranteed safe tracking. IEEE Transactions on Automatic Control. 2021;**66**(12):5861-5876

[2] Dong L, He Z, Song C, Sun C. A review of mobile robot motion planning methods: From classical motion planning workflows to reinforcement learning-based architectures. Journal of Systems Engineering and Electronics. 2023;**34**(2):439-459

[3] Ali ZA, Zhangang H, Hang WB. Cooperative path planning of multiple UAVs by using max–min ant colony optimization along with cauchy mutant operator. Fluctuation and Noise Letters. 2021;**20**(01):2150002

[4] Lin S, Liu A, Wang J, Kong X. A review of path-planning approaches for multiple mobile robots. Machines. 2022;**10**(9):773

[5] Thomason W, Knepper RA. A unified sampling-based approach to integrated task and motion planning. In: The International Symposium of Robotics Research. Cham: Springer International Publishing; 2019. pp. 773-788

[6] Véras LGDO, Medeiros FLL, Guimaráes LNF. Systematic literature review of sampling process in rapidly exploring random trees. IEEE Access. 2019;7:50933-50953

[7] Gholamhosseinian A, Seitz J. A comprehensive survey on cooperative intersection management for heterogeneous connected vehicles. IEEE Access. 2022;**10**:7937-7972

[8] Ali ZA, Han Z. Path planning of hovercraft using an adaptive ant colony with an artificial potential field algorithm. International Journal of Modelling, Identification and Control. 2021;**39**(4):350-356

[9] Vázquez JL, Brühlmeier M, Liniger A, Rupenyan A, Lygeros J. Optimization-based hierarchical motion planning for autonomous racing. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE; 2020. pp. 2397-2403

[10] Wei H, Yang S. MPC-based motion planning and control enables smarter and safer autonomous marine vehicles: Perspectives and a tutorial survey. IEEE/CAA Journal of Automatica Sinica. 2022

[11] Xin J, Yaoguang Q, Zhang F, Negenborn R. Distributed model predictive contouring control for real-time multi-robot motion planning. Complex System Modeling and Simulation. 2022;**2**(4):273-287

[12] Ali ZA, Han Z, Masood RJ. Collective motion and self-organization of a swarm of UAVs: A cluster-based architecture. Sensors. 2021;**21**(11):3820

[13] Darvishpoor S, Darvishpour A, Escarcega M, Hassanalian M. Nature-inspired algorithms from oceans to space: A comprehensive review of heuristic and meta-heuristic optimization algorithms and their potential applications in drones. Drones. 2023;**7**(7):427

[14] Sun H, Zhang W, Yu R, Zhang Y. Motion planning for mobile robots—Focusing on deep reinforcement learning: A systematic review. IEEE Access. 2021;**9**:69061-69081

[15] Yang Y, Juntao L, Lingling P. Multi-robot path planning based on a deep

reinforcement learning DQN algorithm. CAAI Transactions on Intelligence Technology. 2020;**5**(3):177-183

[16] Primatesta S, Osman A, Rizzo A. MP-RRT#: A model predictive sampling-based motion planning algorithm for unmanned aircraft systems. Journal of Intelligent and Robotic Systems. 2021;**103**:1-13

[17] Hu Y, Yao Y, Ren Q, Zhou X. 3D multi-UAV cooperative velocity-aware motion planning. Future Generation Computer Systems. 2020;**102**:762-774

[18] Ali ZA, Zhangang H. Multi-unmanned aerial vehicle swarm formation control using hybrid strategy. Transactions of the Institute of Measurement and Control. 2021;**43**(12):2689-2701

[19] Queralta JP, Taipalmaa J, Pullinen BC, Sarker VK, Gia TN, Tenhunen H, et al. Collaborative multi-robot systems for search and rescue: Coordination and perception. arXiv preprint arXiv: 2008. 2020:12610

[20] Tamizi MG, Yaghoubi M, Najjaran H. A review of recent trend in motion planning of industrial robots. International Journal of Intelligent Robotics and Applications. 2023:1-22

[21] Ali ZA, Masroor S, Aamir M. UAV based data gathering in wireless sensor networks. Wireless Personal Communications. 2019;**106**:1801-1811

[22] Guo H, Wu F, Qin Y, Li R, Li K, Li K. Recent trends in task and motion planning for robotics: A survey. ACM Computing Surveys. 2023

[23] Teng S, Hu X, Deng P, Li B, Li Y, Ai Y, et al. Motion planning for autonomous driving: The state of the art and future perspectives. IEEE Transactions on Intelligent Vehicles. 2023

[24] Ali ZA, Israr A, Alkhammash EH, Hadjouni M. A leader-follower formation control of multi-UAVs via an adaptive hybrid controller. Complexity. 2021;**2021**:1-16

[25] Aradi S. Survey of deep reinforcement learning for motion planning of autonomous vehicles. IEEE Transactions on Intelligent Transportation Systems. 2020;**23**(2):740-759

Chapter 2

Survey of Methods Applied in Cooperative Motion Planning of Multiple Robots

Zain Anwar Ali, Amber Israr and Raza Hasan

Abstract

Recent advances in robotics, autonomous systems, and artificial intelligence (AI) enable robots to perform complex tasks such as delivery, surveillance, inspection, rescue, and others. However, they are unable to complete certain tasks independently due to specific restrictions. In the last few years, researchers are keenly interested in deploying multi-robots for such tasks due to their scalability, robustness, and efficiency. Multiple robots; mobile robots, unmanned aerial vehicles (UAVs), unmanned ground vehicles (UGVs), and unmanned underwater vehicles (UUVs); are gaining much momentum and versatility in their operations, whereas cooperative motion planning is a crucial aspect of incorporating these robots into boundless applications. The purpose of this review chapter is to present an insightful look into the problem of cooperative motion planning with its solution and a comprehensive assessment of various path-planning techniques, task-based motion planning techniques, and obstacle avoidance protocols. It further explores the role of AI in the motion planning of multi-robots. Moreover, it highlights various applications and existing issues in these applications that require future consideration. This review chapter implies that researchers, industries, and academia should aspire to cooperative motion planning for robotic expansions.

Keywords: AI, cooperative path planning, multiple robots, Mobile robots, UAVs, UGVs, UUVs, path planning, motion planning, obstacle avoidance protocols

1. Introduction

Recent advances in artificial intelligence (AI), robotics, communication, and other technologies are modifying the characteristics of robots. Different robots are deployed for different environments; for example, unmanned aerial vehicles (UAVs) are used for aerospace, unmanned ground vehicles (UGVs) are employed for ground, and unmanned underwater vehicles (UUVs) are applied for underwater. However, in complex scenarios, an individual robot is unable to reach its destination while avoiding collision with any obstacle and fulfilling its task in complex missions [1]. Therefore, a group or swarm of inexpensive and small robots are deployed that coordinate with each other and accomplish a common goal efficiently, rapidly, and robustly. Multi-robots are widely employed for various complex and challenging tasks such as inspection, monitoring, search and rescue, navigation, and security in disaster, marine exploration, manufacturing industries, smart agriculture, military, and other fields.

The cooperation among multiple robots occurs using the available information from the network. Hence, the accurate measurement of a robot's position in correspondence to other robots and the environment provides assistance in avoiding collisions with other robots and obstacles [2]. Researchers are engaged in developing strategies for cooperative motion planning of robots. Coordination may be dynamic or static according to the environment. For cooperative motion planning, the shortest distance, safety distance from obstacles, trajectory smoothness, and computational time are the prime factors that must be considered [3].

Motion planning and path planning are closely related, and motion planning in many places is labeled as path planning. However, in motion planning additional dynamic properties such as velocity and acceleration are taken into consideration. Thus, motion planning is the subset of path planning [4]. For reliable operations of multi-robots, path planning aims to determine a collision-free path in the shortest time. Multiple robot path planning shows high computational complexity and offers optimal solutions.

To better deploy multi-robots, other research areas are task-based motion planning and obstacle avoidance protocols [5]. These approaches assist multiple robots to have a better knowledge of the environment while planning motion and executing tasks so that collision with the obstacles or other agents does not occur. Furthermore, the role of AI techniques in motion planning for operating multi-robots securely and intelligently is evident from their application in diverse studies [6]. Besides all these advances, limitations still exist for future considerations. Therefore, a proper cooperative motion planning approach has always been an non-deterministic polynomial-time hardness (NP-hard) problem, and developing efficient schemes has always been a crucial research topic in the last decades.

The aforementioned discussions have inspired us to examine and compare the widely applied algorithms and methods in the recent literature for carrying out the cooperative motion planning of robots. The novelty of this review chapter lies in covering literature on all robots such as mobile robots and unmanned vehicles; UAVs, UGVs, and UUVs, and presenting it in one place. The main contributions of this chapter are threefold,

- Presenting a review of breakthrough results in the context of motion planning for multi-robots.

- Comparing and analyzing the recent path-planning techniques, task-based motion planning approaches, and obstacle avoidance protocols for multiple robots.

- Exploring the role of AI in motion planning.

- Evaluating various applications of multiple robots.

The structure of this chapter is organized as follows: Section 2 presents an analysis of related work. Section 3 defines the problem of cooperative motion planning and proposes a solution for it. Section 4 evaluates multi-robot path-planning techniques. Section 5 describes task-based motion planning. Section 6 covers optical avoidance protocols for multi-robots. Section 7 explores the role of AI in motion planning. Section

8 presents the applicability of cooperative motion in various fields. Section 9 discusses the conclusion of this chapter and future issues in cooperative motion planning.

2. Related work

Several literature reviews are conducted for the motion planning of multiple robots. The widely focused are mobile robots, and then followed by UAVs and autonomous underwater vehicles (AUVs). All the papers have discussed path-planning strategies from classical to emerging AI techniques, such as reinforcement learning (RL) and machine learning (ML).

Different motion planning techniques are analyzed with a prime focus on highway planning and UGVs [7]. Decision-making and path-generation concepts are discussed to elaborate motion planning. Findings reveal that a huge number of algorithms are reviewed in this chapter. This study includes not only state-of-the-art work but also suggests decomposition methods for highway motion planning and encourages autonomous driving. Various methods developed on motion planning policy are reviewed [8]. This chapter is focused on mobile robots in an unstructured environment but has explored some studies on UAVs. The conventional and emerging deep reinforcement learning (DRL) methods that involve multi-robot systems, meta-learning, and imitation learning are enlightened. Restricted theoretical development along with the low interpretability is suggested to be the main reasons that hinder their real-time applications.

Researchers survey studies that applied ML for control and motion planning in the navigation of mobile robots [9]. They compare and contrast ML approaches with classical approaches in the context of navigation. Findings reveal that classical navigation issues are required to be examined with an ML perspective. It further evaluates that despite advances, classical approaches are unable to solve navigation problems. A comprehensive and clear understanding related to opportunities, limitations, relationships, and the future of different motion planning algorithms is presented [3]. Traditional algorithms to policy gradient reinforcement learning algorithms are discussed for intelligent robots. This study paves the way for improved motion planning algorithms. Optimization methods are discussed in Ref. [10] for motion planning of UAVs. Findings reveal that swarm-based optimization approaches are preferred by researchers due to their exceptional ability in complex scenarios.

A survey is carried out to evaluate various methods of motion planning and task planning for the cooperative working of multiple mobile robots [6]. A taxonomy based on system capabilities is proposed in this study that applies to single-robot systems and multi-robot systems. Various motion planning methods, from classical to reinforcement learning (RL) approaches, are reviewed for single-robot and multi-robots [11]. It covers different types of robots such as UAVs, wheeled mobile robots (WMRs), AUVs, etc. It concludes that motion planner based on RL is model-free and achieves unification of the local planner and the global planner but shows various limitations that hinder its real-time applications.

3. Problem definition and solution proposed

Motion planning is to determine a sequence of feasible robot configurations called trajectories that allow moving a robot from its initial stage to its final destination with

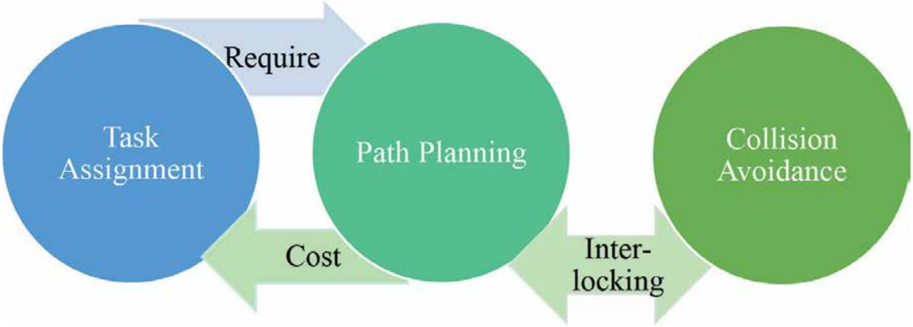


Figure 1.
An integrated architecture for cooperative motion planning.

collision avoidance and obstacle avoidance for completing a given task. It involves various variables such as robots' dynamics, kinematics, environment, and task constraints. Motion planning optimizes a robot's motion by enhancing its throughput and minimizing its cycle time. It can be applied to verify a process's feasibility and to estimate potential problems before deploying robots. The major problem in the development of robots especially autonomous vehicles is to devise a way in which they are capable enough to make their plans in different situations. Therefore, motion planning is essential in the deployment of multiple robots in an environment consisting of obstacles. The degree of motion planning problem varies according to a couple of factors whether all the obstacles' information regarding their locations, sizes, and motion, is known before the deployment of the robot or whether the obstacles are stated or dynamic in an environment.

The solution to cooperative motion planning is an integrated architecture for multi-robots. **Figure 1** shows this architecture is comprised of three layers: the task assignment layer, the past planning layer, and the collision avoidance layer. The task assignment layer must receive the cost which is a distance function for reaching the final destination. This cost is received when communication occurs between the task assignment layer and the path-planning layer. The path-planning layer is called again and again until a feasible solution is obtained for task assignment. Then, the path-planning layer uses the information about obstacles that are known before the mission starts and computes an original collision-free path. When any unmapped obstacle appears, the collision avoidance layer determines a collision-free path efficiently locally detouring from the original path. It also prevents collision between two agents, when an agent reaches a target point during path following. If the agent cannot reach the required target points promptly, then the task assignment layer redistributes the task among other agents in the neighborhood. It is duly considered that by explicitly considering the characteristics of a vehicle or robot, this solution can be further refined.

4. Multi-robot path planning

Various path-planning algorithms have been proposed and grouped on different criteria, such as a study categorizes the path-planning algorithms on coordination criteria into deliberative approaches and reactive approaches [12]. Deliberative approaches are further classified into evolutionary algorithms and road map-based algorithms, whereas the reactive approaches are further grouped into potential field algorithms and modern predictive control approaches. Another study classifies the path-planning algorithms according to their application in static and dynamic environments [13]. The main categories are classical algorithms and soft computing techniques. Classical algorithms are comprised of cell decomposition, road map, Voronoi-diagram, and potential field. On the other hand, soft computing techniques include artificial neural networks (ANN), fuzzy logic, and hybrid and evolutionary techniques. According to a recent study on multi-robot, path-planning approaches are grouped into classical approaches, heuristic algorithms, AI techniques, and bio-inspired algorithms [14]. **Figure 2** shows classification of these path-planning algorithms.

4.1 Classical approaches

Classical approaches usually involve a predefined graph that requires high computational space and time. These techniques do not ensure completeness and are not capable of re-plan the path in the application. These approaches are classified into graph-based approaches, artificial potential field (APF), and sampling-based approaches. A study proposed APF for AUV flocks that are enabled with software-defined networking (SDN) [15]. Results show that the suggested path-planning scheme allows efficient path planning.

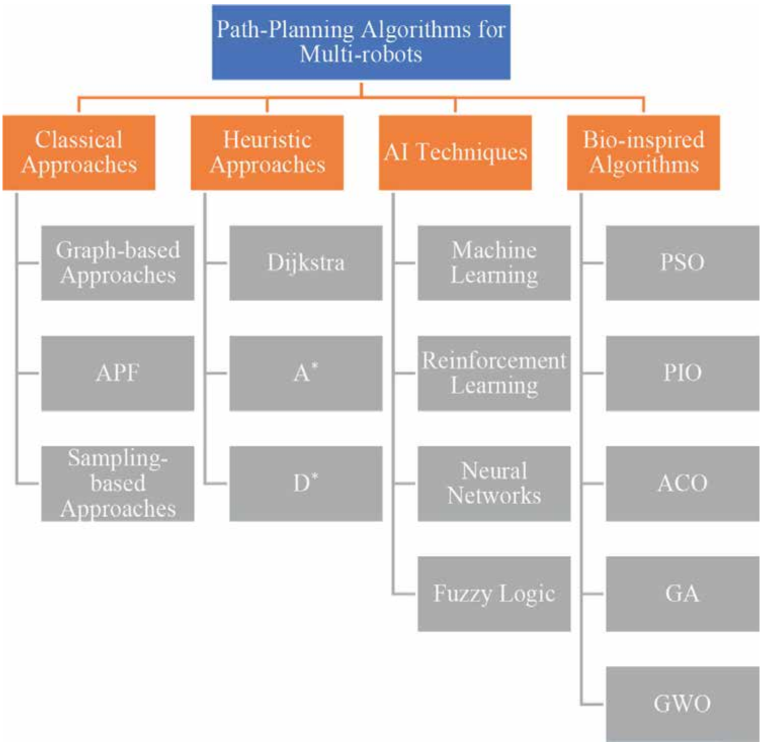


Figure 2.
Path-planning approaches for multi-robots.

4.2 Heuristic approaches

The heuristic approaches solve the problems that cannot be addressed by other approaches and estimate an approximate solution rather than an exact solution. Therefore, these algorithms are also called approximation algorithms. These algorithms are easily applicable and develop cost functions to evaluate the path. It searches the subspace of the search space and generates only near-optimal results. Moreover, they require lower space and runtime. A-star (A*) search algorithm and D* algorithm are extensively applied through heuristic algorithms. The A* algorithm with a distributed velocity perception strategy is proposed for the cooperative motion planning of multiple UAVs in three-dimensional (3D) [4]. Simulations suggest that the applied algorithm enables UAVs to take less time and shorter paths for reaching their destinations safely.

4.3 Artificial intelligence techniques

Intelligent systems have gained more attention these days. AI techniques are developed to overcome the limitations of traditional reinforcement learning. AI-based algorithms and models possess self-learning abilities and have completed characteristics for the path planning of multiple robots with faster convergence. Researchers have focused more on machine learning (ML) algorithms, reinforcement learning (RL), neural networks (NN), fuzzy logic, etc. A study has developed a new mobile edge computing (MEC) platform for multi-UAVs and has suggested an RL framework for path planning [16]. Simulations show the feasibility and effectiveness of the platform.

4.4 Bio-inspired algorithms

Bio-inspired techniques are inspired by the behavior of animals and use particles to generate paths. They are primary algorithms for the path planning of multiple robots because they show computational efficiency and have powerful implementations. Bio-inspired techniques include particle swarm optimization (PSO), pigeon-inspired optimization (PIO), ant colony optimization (ACO), genetic algorithm (GA), gray wolf optimizer (GWO), and other bio-inspired techniques. A novel switching delayed particle swarm optimization (SDPSO) algorithm is proposed for UAVs [17]. Simulation results evaluate that the proposed technique shows robustness and quickly plans paths of high quality. **Table 1** presents a comparative analysis of path-planning algorithms applied in different studies.

References	Deployed platform	Path planning	Evaluation	Advantages
[4]	Multi-UAVs	Velocity aware A*	Simulations	• Produces shorter paths • Assists in reaching the destination in less time
[15]	AUV flock	APF-SDN	Simulations	Allows efficient path planning
[16]	Multi-UAVs	MEC-RL	Simulations	Provides flexibility and effectiveness
[17]	UAVs	SDPSO	Monte-Carlo simulations	• Shows robustness • Quickly plans paths of high quality

Table 1.
Comparative analysis of path-planning algorithms.

5. Task-based motion planning

For performing a task, a robot has to determine feasible paths then while executing the plan, it has to consider the complexity of the environment before stepping ahead to accomplish its goals. This whole procedure is classified into task planning and motion planning. Task planning is referred to computing solvable plans for decomposing a long-horizon task into several elementary subtasks in a discrete space, whereas motion planning is referred to transforming a subgoal into a set of perimeters for achieving that subgoal in a continuous space. As both concepts share a few similar designs, task-based motion planning is introduced for integrating the discrete planning methods with continuous planning methods. Researchers referred to it as task and motion planning (TAMP) [18]. TAMP chooses the chronology of high-level actions for the robot. Then, it selects the hybrid parameter values that find a way for performing these actions. Finally, it prefers low-level motion for executing these actions safely. However, solutions to this merger vary in the manner in which their joint search space is explored. **Figure 3** illustrates TAMP approaches are classified into classical approaches, learning approaches, and hybrid approaches [19].

In certain cases, various tasks are assigned to multiple robots in chronological order of tasks. Once the multi-robots are deployed, the dynamic adjustment of tasks leads to difficulties in distribution and path re-planning. An optical reference point built on an improved fruit fly optimization algorithm (ORPFOA) is utilized to solve this path-planning issue with changing tasks in the 3D oilfield inspection by cooperative multi-UAVs [20]. Results show that the proposed algorithm solves not only the path-planning issue but also the assignment issue of the initial and new tasks. Two algorithms based on task assignment: The marginal-cost algorithm (MCA) and regret-based MCA (RMCA) are developed to carry out task assignment simultaneously with path planning for multi-agent robots [21]. Large neighborhood search (LNS) is also added that enhances solution sustainability. The applied strategies allow each robot to carry multiple packages. Local motion planning (LMP) is integrated with a robust H∞ decentralized feedforward reference tracking fault-tolerant control (FTC) for a hybrid team system comprised of UAVs and biped robots [22]. The suggested strategy shows effectiveness in the search and rescue tasks.

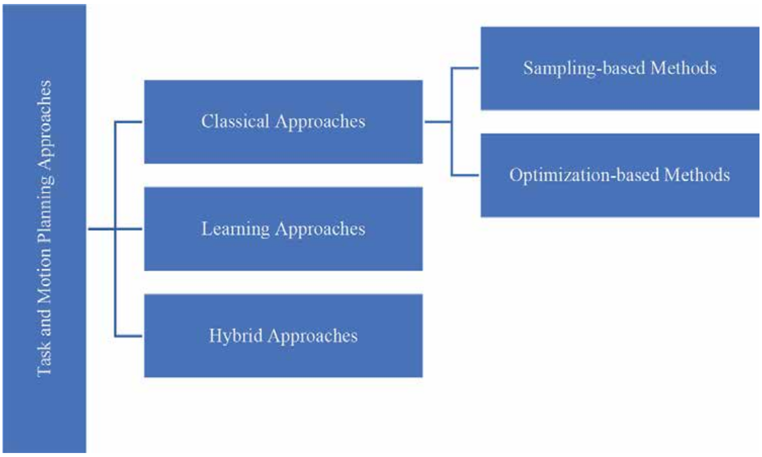


Figure 3.
Classification of task and motion planning approaches.

However, instead of AI technologies, bio-inspired algorithms are suggested widely to enable real-time applications [23]. **Table 2** highlights that most of the studies applied bio-inspired algorithms for task-based motion planning in simulations to assure their significance for real-time applications. Cooperative UUVs are used for target detection task scheduling merged with an underwater acoustic environment [24]. In this study, task scheduling is built on the GA algorithm. Findings reveal that it refines the cooperative detection ability at the task level and solves the optimization issue. A hybrid algorithm is suggested with the estimation of the distributed algorithm (EDA) and the GA [1]. Simulations show EDA-GA plans stable and quality paths. An online adjustment strategy is also proposed that maintains complete coverage and reduces the effect on circular paths.

References	Deployed robot	Task-based motion planning technique	Focus	Environment	Performance
[1]	UAVs and UGVs	EDA-GA with online adjustment strategy	Surveillance task	Simulation	• Plans stable and quality paths • The online strategy maintains complete coverage and reduces the effects on circular paths
[20]	Multi-UAVs	ORPFOA	Oilfield inspection	Simulation	• Solves path planning faster • Allows higher optimizing precision
[21]	Multi-agent robots	MCA and RMCA with LNS	Pickup and delivery	Numerical simulation	• Performs task assignments simultaneously with path planning • Allows each robot to carry multiple packages
[22]	UAVs and biped robots	LMP-H∞ decentralized feedforward reference tracking FTC	Search and rescue	Simulation	Shows effectiveness in task and motion planning
[24]	UUVs	Task scheduling based on GA and integrated with an underwater acoustic environment	Target detection	Simulation	• Refines the cooperative detection ability at the task level • Solves the optimization issue

Table 2.
Review of various task-based motion planning techniques applied on different multiple robots.

6. Obstacle avoidance protocol

Multi-robots are destined to explore more distance in less time with collision avoidance. When multiple robots are deployed, they sense other agents and obstacles in the path, avoid them, and continue their motion. Therefore, collision avoidance is an integral part of part planning of robots. Collision avoidance protocols are categorized into two main approaches: centralized methods and decentralized methods. The centralized methods are efficient for smaller numbers of robots, whereas the decentralized methods are less expensive computationally and more effective for large groups of robots. A decentralized architecture namely, spot auction-based robotic collision avoidance scheme (SPARCAS) is applied for collision avoidance and M^* is used for path planning of UGVs [25]. It results in prioritization and dynamic handling.

In various robots, the collision avoidance control system is either deliberate or reactive. Deliberate models are plan-driven and computationally expensive as it requires *a priori* data of the environment. Contrary, reactive architectures are faster and require only real-time sensor information. Such as a deep RL method, policy proximal optimization (PPO) with curriculum learning is used for AUVs [26]. The applied methodology effectively plans paths with collision avoidance in 3D.

References	Deployed platforms	Obstacle avoidance protocol	Obstacles	Evaluation indexes	Performance
[23]	UAV flock	MMACO-CM MMACO-DE	Tornados and mountains	Convergence time and route taken	• MMACO-DE takes lesser time • Reduces distance and saves time
[25]	Multiple robots	SPARCAS-M^*	Obstacles	Path length, average payment, average waiting, and time	Allows prioritization and dynamic handling
[26]	AUVs	PPO-curriculum learning	Obstacles or ocean current	Collision rate, success rate, and average tracking error	Effectively plans path with collision avoidance in 3D
[27]	Multi-USVs	APFF	Obstacles	Time, route length, path length	Improves safety
[28]	Multi-UGVs	OWS Q-learning with collision avoidance cooperation method	Green belts and buildings	Average shortest path, path steps, average reward, and time	• Effectively solves the coordination problem • Achieves the convergence effect and collision-free path in the least time

Table 3.
Comparative analysis of obstacle avoidance protocols.

Several studies indirectly address the problem of path planning with a collision avoidance perspective for multiple robots. Such as an APF function (APFF), which uses position information, is proposed for multi-USVs during navigation [27]. It improves safety while ensuring collision avoidance and obstacle avoidance. Similarly, the max-min ant colony optimization (MMACO) approach with the Cauchy mutation operator (MMACO-CM) and MMACO approach with differential evolution (MMACO-DE) are suggested in Ref. [23]. Findings show that MMACO-DE reduces distance and saves time by taking lesser turns in a 3D complex environment. Optimized-weighted-speedy Q-learning (OWS Q-learning) algorithm and a collision avoidance cooperation method are suggested for multiple UGVs [28]. **Table 3** summarizes the abovementioned literature in tabular form and provides a comparative analysis of the deployed platforms, applied obstacle avoidance protocols, considered obstacles and evaluation indexes, and finally the resulting performance in each study.

7. Role of artificial intelligence in motion planning

Motion planning is an integral part of robots. AI-based motion planning is revolutionizing the robotics and autonomous system fields. Due to this, these robots and autonomous systems have become capable to navigate themselves autonomously while avoiding collisions and dynamic and uncertain obstacles to perform any task. The flourishing AI techniques: deep learning (DL) algorithms and reinforcement learning (RL) give better performances in handling nonlinear problems with complexity [3]. The complexity includes incompleteness, ambiguity, and the most challenging one uncertainty. As classical ML algorithms do not address time sequential planning problems, they are modified into optimal value RL and policy gradient RL. Optimal value RL applies Q learning and transforms it into deep Q-learning network (DQN), whereas policy gradient RL uses policy gradient and actor-critic algorithm and forms asynchronous advantage actor-critic (A3C), trust region policy optimization (TRPO), and deterministic policy gradient (DPG) [29].

Researchers, technology giants, and institutions are engaged in introducing new motion planning approaches by employing AI techniques or integrating advanced ML techniques with traditional algorithms. These approaches have introduced us to the concepts of autonomous vehicles and unmanned vehicles [30]. Such as Google, Tesla, Baidu, Audi, and Toyota have developed autonomous vehicles (AVs) using AI technologies. These vehicles improve human safety, assist in environmental protection, and reduce emissions, crashes, and congestion.

AI-based motion planning gives reactive or instant responses by considering short-term optimal or suboptimal reactive strategies. Moreover, it achieves long-term optimal planning objectives such as path planning during robots' interaction with the environment [7, 31]. Moreover, it streamlines various processes and replaces hardware with software. This results in improved machine performance, product quality and productivity, and efficiency, with reduced time to market and costs. The automation industry relies on AI-based motion control and planning. Particularly manufacturers trust AI for smart, safe, flexible, and productive decision-making in motion control processes. Particularly, AI reduces the time and costs for setting up the configuration of large plants and for hiring and training new employees and enhances productivity through automation [10].

Besides so many advantages, several limitations and challenges still exist. The promising ongoing research and development will overcome these hurdles and will produce more efficient AI-powered systems in the near future. Additionally, AI-based motion planning will gain more focus for development in the future of automation and robotics [6, 32]. This will enhance the quantity and applications of automation and robotics in the military, agriculture, and various industries, such as healthcare, manufacturing, transportation.

8. Applications of cooperative motion

Groups or swarms of robots are comprised of multiple heterogeneous and homogeneous agents. Cooperative motion enables these agents to achieve their single goal [33]. Earlier the multi-robot cooperative system was applied at a small scale and was relying on simple interactions and rules. Nowadays, such systems are developed for collaborative intelligence, large-scale, and cluster-scale operations. Diverse theories are introduced into these cooperative systems that need to progress in experimental research and theoretical exploration. Considering the example of a multi-vehicle cooperative system, this collaborative system involves cooperative motion planning, collaborative positioning, scheduling and allocation of intelligent tasks, traffic flow optimization, and collaborative planning control. These technologies enable them to analyze and make decisions according to the information achieved from interactions with other vehicles, networks, and roads [34].

Clusters of multi-unmanned systems are used in complex weather, terrain, and electromagnetic environments signifying the efficiency of cooperative motion. Various scientific research institutions have conducted a series of research on cooperative systems in unknown larger areas. For example, the US Office of Naval Research carried out research on clusters of unmanned combat vehicles. Great improvements in their capabilities were realized [35].

Besides military applications, mobile robots and unmanned vehicles play significant roles in the manufacturing industry, agriculture sector, service industry, hospitals, and in hazardous scenarios such as disaster management, firefighting, rescue and search operations, and so on. They are extensively employed to replace humans, reduce injuries, and improve efficiency in a limited time, such as JD Logistics employs the collaborative systems of multi-robots in its warehouse and distribution center for performing collaborative operations to sort express deliveries. In industries, cooperative systems work for assembling and disassembling numerous small parts of a machine [36]. Similarly, groups or swarms of UAVs are widely used for monitoring, disease and pest detection, harvesting, and other purposes. These collaborative systems render their rapid service in catastrophes scenarios, such as monitoring, search, and rescue, and delivering aid and medicine in disasters.

This section signifies that all frameworks of collaborative motion planning possess a high scope of application. However, the mutual obstacle avoidance protocol is susceptible to environmental interference and sensor interference, which leads to catastrophic situations. Therefore, the safety essentials of the multi-vehicle system are unable to take full advantage of cooperative motion and demand further research.

9. Conclusion and future issues

This study has conducted a review of breakthrough results and existing future challenges of motion planning for diverse categories of robots. The main objective is to take an insightful look into the problem of cooperative motion planning with its solution, and a detailed analysis of various path-planning techniques, task-based motion planning techniques, and obstacle avoidance protocols and to present it in one place. A comparison with other review papers shows that most of the papers surveyed different classical methods and advanced methods, while most of them are focused on RL methods, UAVs, and mobile robots. The road map approach, the cell decomposition approach, and the potential field approach are observed to be advantageous solutions for the problem of cooperative motion. The path-planning algorithms allow flexible, efficient, high-quality, more traversable paths and cover more distances and show robustness for UAVs, AUVs, and UGVs in simulations and real-time experiments. Bio-inspired algorithms are widely employed for improving the cooperative motion planning of robots and UVs at the task levels, planning faster and more stable paths, and allowing each agent to accomplish its task. Various obstacle avoidance protocols are discussed that effectively solve the coordination problem, and achieve the convergence effect with safety and collision-free paths for multiple UVs and robots in different terrains. The significant role of AI in motion planning is also elaborated. The research and diverse application areas of these intelligent robots and intelligent vehicles show promising outcomes. In general, the analysis reveals that bio-inspired and RL methods are extensively studied and UAVs are commonly deployed in most of the studies. The studies have conducted these algorithms mostly in simulations. Real-time applications are rarely considered.

Besides advantages, some future issues of the analyzed methods are also evaluated in this section. As the integration of learning-based approaches into planning is essential for planners to work and reason with learned action models rather than human assistance or knowledge, task-based motion planning requires more incorporation of sampling and optimization methods. Further research is essential to plan more curvilinear paths and develop more adaptable obstacle avoidance protocols for complex scenarios with narrow paths and a large number of robots.

Considering the application domain, cooperative motion planning provides solutions for small instances only but at the scalability cost. Further issues that require future considerations include timeout failures, greedy assignments, and the least regard for their effects on the costs of overall solutions. Cooperative motion planning necessities more extension to realistic environments with deformable objects, dynamics, time, etc. These approaches need more capabilities to consider the uncertainty of present and future states with a prime focus on safety and performance.

Abbreviations

3D	three-dimensional
A3C	Asynchronous Advantage Actor-Critic
ACO	Ant Colony Optimization
AI	Artificial Intelligence
ANN	Artificial Neural Networks
APF	Artificial Potential Field

APFF	Artificial Potential Field Function
AUVs	Autonomous Underwater Vehicles
AVs	Autonomous Vehicles
DL	Deep Learning
DPG	Deterministic Policy Gradient
DQN	Deep Q-Learning Network
DRL	Deep Reinforcement Learning
EDA	Estimation of the Distributed Algorithm
FTC	Fault-Tolerant Control
GA	Genetic Algorithm
GWO	Gray Wolf Optimizer
LMP	Local Motion Planning
LNS	Large Neighborhood Search
MCA	Marginal-Cost Algorithm
MEC	Mobile Edge Computing
ML	Machine Learning
MMACO	Max-Min Ant Colony Optimization
MMACO-CM	Max-Min Ant Colony Optimization with Cauchy Mutation Operator
MMADE	Max-Min Ant Colony Optimization with Differential Evolution
NN	Neural Networks
NP-hard	Non-Deterministic Polynomial-Time Hardness
ORPFOA	Optical Reference Point-Fruit Fly Optimization Algorithm
OWS Q-learning	Optimized-Weighted-Speedy Q-Learning
PIO	Pigeon-Inspired Optimization
PPO	Policy Proximal Optimization
PSO	Particle Swarm Optimization
RL	Reinforcement Learning
RMCA	Regret-Based Marginal-Cost Algorithm
SDN	Software-Defined Networking
SDPSO	Switching Delayed Particle Swarm Optimization
SPARCAS	Spot Auction-Based Robotic Collision Avoidance Scheme
TAMP	Task And Motion Planning
TRPO	Trust Region Policy Optimization
UAVs	Unmanned Aerial Vehicles
UGVs	Unmanned Ground Vehicles
UUVs	Unmanned Underwater Vehicles
WMRs	Wheeled Mobile Robots

Author details

Zain Anwar Ali[1]*, Amber Israr[2] and Raza Hasan[3]

1 School of Physics and Electronic Engineering, JiaYing University,
Meizhou, Guangdong, China

2 Electronic Engineering Department, Sir Syed University of Engineering and
Technology, Karachi, Pakistan

3 Department of Science and Engineering, Solent University, Southampton,
United Kingdom

*Address all correspondence to: zainanwar86@hotmail.com

References

[1] Wu Y, Shaobo W, Xinting H. Cooperative path planning of UAVs & UGVs for a persistent surveillance task in urban environments. IEEE Internet of Things Journal. 2020;**8**(6):4906-4919

[2] Sangiovanni B, Incremona GP, Piastra M, Ferrara A. Self-configuring robot path planning with obstacle avoidance via deep reinforcement learning. IEEE Control Systems Letters. 2020;**5**(2):397-402

[3] Zhou C, Huang B, Fränti P. A review of motion planning algorithms for intelligent robots. Journal of Intelligent Manufacturing. 2022;**33**(2):387-424

[4] Hu Y, Yao Y, Ren Q, Zhou X. 3D multi-UAV cooperative velocity-aware motion planning. Future Generation Computer Systems. 2020;**102**:762-774

[5] Iqbal MM, Ali ZA, Khan R, Shafiq M. Motion planning of UAV swarm: Recent challenges and approaches. In: Aeronautics-New Advances. BoD – Books on Demand; 2022

[6] Antonyshyn L, Silveira J, Givigi S, Marshall J. Multiple mobile robot task and motion planning: A survey. ACM Computing Surveys. 2023;**55**(10):1-35

[7] Claussmann L, Revilloud M, Gruyer D, Glaser S. A review of motion planning for highway autonomous driving. IEEE Transactions on Intelligent Transportation Systems. 2019;**21**(5):1826-1848

[8] Sun H, Zhang W, Runxiang Y, Zhang Y. Motion planning for mobile robots—Focusing on deep reinforcement learning: A systematic review. IEEE Access. 2021;**9**:69061-69081

[9] Xiao X, Liu B, Warnell G, Stone P. Motion planning and control for mobile robot navigation using machine learning: A survey. Autonomous Robots. 2022;**46**(5):569-597

[10] Israr A, Ali ZA, Alkhammash EH, Jussila JJ. Optimization methods applied to motion planning of unmanned aerial vehicles: A review. Drones. 2022;**6**(5):126

[11] Dong L, He Z, Song C, Sun C. A review of mobile robot motion planning methods: From classical motion planning workflows to reinforcement learning-based architectures. Journal of Systems Engineering and Electronics. 2023;**34**(2):439-459

[12] Kumar S, Sikander A. Optimum mobile robot path planning using improved artificial bee colony algorithm and evolutionary programming. Arabian Journal for Science and Engineering. 2022;**47**(3):3519-3539

[13] Das PK, Jena PK. Multi-robot path planning using improved particle swarm optimization algorithm through novel evolutionary operators. Applied Soft Computing. 2020;**92**:106312

[14] Lin S, Liu A, Wang J, Kong X. A review of path-planning approaches for multiple Mobile robots. Machines. 2022;**10**(9):773

[15] Lin C, Han G, Jiaxin D, Bi Y, Shu L, Fan K. A path planning scheme for AUV flock-based internet-of-underwater-things systems to enable transparent and smart ocean. IEEE Internet of Things Journal. 2020;**7**(10):9760-9772

[16] Chang H, Chen Y, Zhang B, Doermann D. Multi-UAV mobile edge computing and path planning platform

based on reinforcement learning. IEEE Transactions on Emerging Topics in Computational Intelligence. 2021;**6**(3):489-498

[17] Yu Z, Si Z, Li X, Wang D, Song H. A novel hybrid particle swarm optimization algorithm for path planning of UAVs. IEEE Internet of Things Journal. 2022;**9**(22):22547-22558

[18] Garrett CR, Chitnis R, Holladay R, Kim B, Silver T, Kaelbling LP, et al. Integrated task and motion planning. Annual Review of Control, Robotics, and Autonomous Systems. 2021;**4**:265-293

[19] Zhang K, Lucet E, Sandretto JAD, Kchir S, Filliat D. Task and motion planning methods: Applications and limitations. In: In 19th International Conference on Informatics in Control, Automation and Robotics ICINCO 2022. SCITEPRESS-Science and Technology Publications; 2022. pp. 476-483

[20] Li K, Ge F, Han Y, Wensu X. Path planning of multiple UAVs with online changing tasks by an ORPFOA algorithm. Engineering Applications of Artificial Intelligence. 2020;**94**:103807

[21] Chen Z, Alonso-Mora J, Bai X, Harabor DD, Stuckey PJ. Integrated task assignment and path planning for capacitated multi-agent pickup and delivery. IEEE Robotics and Automation Letters. 2021;**6**(3):5816-5823

[22] Chen B-S, Hung T-W. Integrating local motion planning and robust decentralized fault-tolerant tracking control for search and rescue task of hybrid UAVs and biped robots team system. IEEE Access. 2023;**11**:45888-45909

[23] Anwar AZ, Han Z, Masood RJ. Collective motion and self-organization of a swarm of UAVs: A cluster-based architecture. Sensors. 2021;**21**(11):3820

[24] Jiang C, Yiqun H, He Y, Feng W, Wen X, Li J. Cooperative UUVs detection task scheduling integrated with underwater acoustic environments. In: In 2022 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC). IEEE; 2022. pp. 1-6

[25] Sankar D, Nath S, Saha I. SPARCAS: A decentralized, truthful multi-agent collision-free path finding mechanism. arXiv. 2019

[26] Havenstrøm ST, Rasheed A, San O. Deep reinforcement learning controller for 3D path following and collision avoidance by autonomous underwater vehicles. Frontiers in Robotics and AI. 2021;**7**:211

[27] Xia G, Sun X, Xia X. Multiple task assignment and path planning of a multiple unmanned surface vehicles system based on improved self-organizing mapping and improved genetic algorithm. Journal of Marine Science and Engineering. 2021;**9**(6):556

[28] Cao Y, Fang X. Optimized-weighted-speedy Q-learning algorithm for multi-UGV in static environment path planning under anti-collision cooperation mechanism. Mathematics. 2023;**11**(11):2476

[29] Yang Y, Xiong X, Yan Y. UAV formation trajectory planning algorithms: A review. Drones. 2023;**7**(1):62

[30] Zhang S, Li Y, Ye F, Geng X, Zhou Z, Shi T. A hybrid human-in-the-loop deep reinforcement learning method for UAV motion planning for long trajectories with unpredictable obstacles. Drones. 2023;**7**(5):311

[31] Khan MT, Raza MM, Saad YR, Seo J, Kim D. Aspects of unmanned aerial vehicles path planning: Overview and applications. International Journal of Communication Systems. 2021;**34**(10):e4827

[32] Tamizi MG, Yaghoubi M, Najjaran H. A review of recent trend in motion planning of industrial robots. International Journal of Intelligent Robotics and Applications. 2023;**7**:1-22

[33] Wei H, Yang S. MPC-based motion planning and control enables smarter and safer autonomous marine vehicles: Perspectives and a tutorial survey. IEEE/CAA Journal of Automatica Sinica. 2022;**10**:8-24

[34] Chen J, Ling F, Zhang Y, You T, Liu Y, Xiaoyan D. Coverage path planning of heterogeneous unmanned aerial vehicles based on ant colony system. Swarm and Evolutionary Computation. 2022;**69**:101005

[35] Zhang M, Liu C, Wang P, Yu J, Yuan Q. Uav swarm real-time path planning algorithm based on improved artificial potential field method. In: Proceedings of 2021 International Conference on Autonomous Unmanned Systems (ICAUS 2021). Singapore, Singapore: Springer; 2022. pp. 1933-1945

[36] Prabhu P, Chowdhury AR. Feasibility study of multi autonomous mobile robots (amrs) motion planning in smart warehouse environment. In: 2021 18th International Conference on Ubiquitous Robots (UR). IEEE; 2021. pp. 380-385

Chapter 3

Perspective Chapter: On the Morse Property for the Distance Function of a Robot Arm

Yasuhiko Kamiyama

Abstract

Morse theory plays a central role when we study the configuration space of various mechanical linkages. As an important linkage, we consider the planar robot arm. It is known that the distance function on its configuration space is a Morse function. On the other hand, for a fixed angle θ, we consider the spatial robot arm whose adjacent bond angles are θ. In chemistry, such an arm is used as a model for protein backbones and has been studied extensively. We consider the distance function on its configuration space. The purpose of this chapter is threefold: First, we study whether the distance function is a Morse function. Second, we determine the minimum and maximum values of the function. Consider the case that the arm consists of four bars. Then our third purpose is to study how the distance function is different from the usual Morse function on the torus.

Keywords: robot arm, configuration space, distance function, Morse function, critical manifold

1. Introduction

1.1 Mathematical study of robotics

A motion planning problem studies a sequence of valid configurations of a given mechanical linkage. The problem is interesting because it has many important robotics applications. Some impressive results are obtained in Refs. [1–3]. Mathematicians are also interested in the configuration space of mechanical linkages. Here, the configuration space is defined as the space of all possible shapes of the linkage. Robot arms are quite important examples of mechanical linkages: in molecular biology, the arms describe molecular shapes and the arms play a central role in statistical shape theory.

Morse theory plays a central role when we study the configuration space of robot arms. (See Ref. [4] for the excellent survey with emphasis on the Morse theory.) In differential topology, Morse theory is a powerful tool to analyze spaces. A function $f: M \to \mathbb{R}$ on a manifold M is called a Morse function if every critical point of$f$ is nondegenerate. Morse functions play a principal role in the Morse theory.

The Morse property for the distance function on the configuration space of the robot arm in $\mathbb{R}^2$ is studied well. The purpose of this chapter is to consider the distance function on the configuration space of another robot arm.

1.2 The most famous mechanical linkage: The robot arm in $\mathbb{R}^2$

We consider the robot arm in $\mathbb{R}^2$, which consists of n bars of length $(1, 1, \ldots, 1)$ connected by revolving joints. The initial point of the robot arm is fixed at O. The configuration space of the robot arm is

$$W_n = \left\{(u_1, \ldots, u_n) \in \left(S^1\right)^n\right\}/SO(2). \tag{1}$$

By the $SO(2)$-action, we may normalize u_1 uniquely to be $(1, 0)$. Hence, there is an identification

$$W_n = \left\{(u_1, \ldots, u_n) \in \left(S^1\right)^n \mid u_1 = (1, 0)\right\}. \tag{2}$$

From (2), we have

$$W_n = \left(S^1\right)^{n-1}. \tag{3}$$

Hereafter, we will use (2) as the definition of W_n. (See the following **Figure 1**.)

1.3 The distance function on W_n: Remarkable results

We define the function

$$\mu_n : W_n \to \mathbb{R} \tag{4}$$

by

$$\mu_n(u_1, \ldots, u_n) = \left\| \sum_{i=1}^{n} u_i \right\|^2. \tag{5}$$

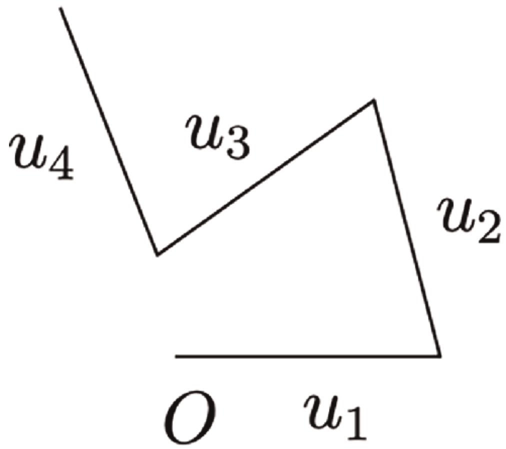


Figure 1.
An element of W_4.

Note that $\mu_n^{-1}(0)$ is the equilateral polygon space, which is a critical manifold of dimension $n-3$. In order to avoid critical manifolds of positive dimension, we consider the following restriction of μ_n:

$$\mu_n|_{W_n \backslash \mu_n^{-1}(0)} : W_n \backslash \mu_n^{-1}(0) \to \mathbb{R}. \tag{6}$$

It is proved in Refs. [5, 6] (see also [[4], Lemma 1.4]) that (6) is a Morse function. More precisely, a critical point corresponds to the case that $u_i = \pm u_1$ for $2 \leq i \leq n$. Moreover, the index of a critical point is determined explicitly. This Morse property has many interesting applications. Among them, the following two applications are particularly important:

- It is proved in Ref. [7] that polygon spaces are obtained from the sphere by successive surgeries.
- The homology groups of planar polygon spaces are determined in Ref. [8].

1.4 A new robot arm: A mathematical model for protein backbones

We fix $\theta \in (0, \pi)$. The robot arm in $\mathbb{R}^3$ with all bond angles θ, permitting "dihedral" spinning about each edge, has been used to model the geometry of protein backbones [9, 10]. Let $X_n(\theta)$ be the configuration space of the arm. That is, $X_n(\theta)$ is defined as the space of all possible shapes of our arm. (See Section 2 for more details.)

1.5 The main problem

Let $f_{n,\theta} : X_n(\theta) \to \mathbb{R}$ be the distance function. In this chapter, we study three problems concerning $f_{n,\theta}$. (See Problem 2.7.) The most important problem is given as follows: Does $f_{n,\theta}$ satisfy the Morse property?

1.6 Motivation for the main problem

We explain the motivation for the above main problem. Recall that we have $W_n = (S^1)^{n-1}$. Similarly, we can prove that $X_n(\theta) = (S^1)^{n-2}$. Moreover, μ_n and $f_{n,\theta}$ are defined to be the distance functions. Now since μ_n satisfies the Morse property, it is natural to ask whether $f_{n,\theta}$ also satisfies the property.

1.7 Previous study on $f_{n,\theta}$

Although some chemists are interested in a special element of $X_n(\theta)$, nobody has studied the space $X_n(\theta)$ nor the function $f_{n,\theta}$. Hence, our results are completely new.

1.8 Summary of the main result

The function $f_{n,\theta}$ satisfies the Morse property if and only if $n = 3$ or 5.

1.9 Organization of this chapter

In Section 2, we first prepare notations. Then we pose three problems in Problem 2.7. Finally, we summarize the known results in Theorems 2.9, 2.10 and 2.11. In Section 3, we state our main theorems. Theorems A and B are answers to the first problem of Problem 2.7. Theorems C and D are answers to the second problem. Theorem E is an answer to the third problem. In Section 4, we construct submanifolds of $X_n(\theta)$, which are used in Section 5. In Section 5, we prove our main theorems. In Section 6, we state the conclusions.

2. Preliminaries

We define the space $X_n(\theta)$ as follows: We fix $\theta \in (0, \pi)$ and define the following space:

$$A_n(\theta) := \left\{ (a_1, \ldots, a_n) \in \left(S^2\right)^n \ \middle| \ \langle a_i, a_{i+1} \rangle = -\cos\theta \ \ (1 \le i \le n-1) \right\}, \tag{7}$$

where $\langle , \rangle$ denotes the standard inner product on $\mathbb{R}^3$. Let $SO(3)$ act on $A_n(\theta)$ diagonally. Then we set

$$X_n(\theta) := A_n(\theta)/SO(3). \tag{8}$$

Using the $SO(3)$-action, we can normalize a_1 and a_2 to be

$$a_1 = (1,0,0) \ \text{ and } \ a_2 = (-\cos\theta, \sin\theta, 0). \tag{9}$$

Then similarly to (2), we use the following identification:

$$X_n(\theta) = \{ (a_1, \ldots, a_n) \in A_n(\theta) \mid a_1 \text{ and } a_2 \text{ are as in } (9) \ \}. \tag{10}$$

Hereafter, we use (10) as the definition of $X_n(\theta)$. (See the following **Figure 2**.)
Lemma 2.1. There is a diffeomorphism

$$T : \left(S^1\right)^{n-2} \to X_n(\theta). \tag{11}$$

Proof: We prove in the same way as given in [[11], Lemma 13] and [[12], Lemma 7]. From an element

$$\left(e^{i\phi_1}, \cdots, e^{i\phi_{n-2}}\right) \in \left(S^1\right)^{n-2}, \tag{12}$$

we construct the element $(a_1, \cdots, a_n) \in X_n(\theta)$ as follows: In the process of constructing a_i, we also construct the elements $n_i \in S^2$ such that $\langle a_i, n_i \rangle = 0$. We set

$$a_{i+2} := -(\cos\theta)a_{i+1} + (\sin\theta\cos\phi_i)n_{i+1} + (\sin\theta\sin\phi_i)a_{i+1} \times n_{i+1} \tag{13}$$

and

$$n_{i+2} := -(\sin\theta)a_{i+1} - (\cos\theta\cos\phi_i)n_{i+1} - (\cos\theta\sin\phi_i)a_{i+1} \times n_{i+1}, \tag{14}$$

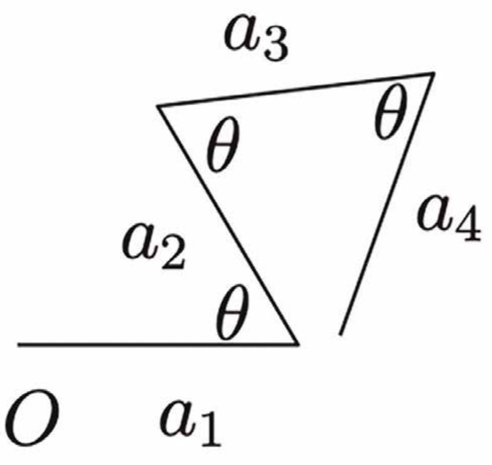


Figure 2.
An element of $X_4(\theta)$.

where $a_{i+1} \times n_{i+1}$ denotes the cross product.

We set

$$\begin{aligned} &a_1 = (1,0,0), \quad n_1 = (0,1,0) \\ &a_2 = (-\cos\theta, \sin\theta, 0) \quad \text{and} \quad n_2 = (-\sin\theta, -\cos\theta, 0). \end{aligned} \tag{15}$$

From (13) and (14) for $i = 1$, we obtain a_3 and n_3. Next from (13) and (14) for $i = 2$, we obtain a_4 and n_4. Repeating this process, we obtain a_i and n_i for $1 \le i \le n$. Now we define T by

$$T\left(e^{i\phi_1}, \cdots, e^{i\phi_{n-2}}\right) = (a_1, \cdots, a_n). \tag{16}$$

From the construction, T is a diffeomorphism.

Notation 2.2. We define the element $[\phi_1, \dots, \phi_{n-2}] \in X_n(\theta)$ by

$$[\phi_1, \dots, \phi_{n-2}] := T\left(e^{i\phi_1}, \cdots, e^{i\phi_{n-2}}\right), \tag{17}$$

where the diffeomorphism T is constructed in Lemma 2.1.

Lemma 2.3. *Assume that $\phi_i \in \{0, \pi\}$ for $1 \le i \le n-2$. We write $[\phi_1, \dots, \phi_{n-2}] = (a_1, \dots, a_n)$. Then the following results hold*:

i. *For $1 \le i \le n-2$, a_i is a vector in $\mathbb{R}^2$.*

ii. *If $\phi_j = \pi$, then we have $a_{j+2} = a_j$. (See the left of the following* **Figure 3**.*) On the other hand, if $\phi_j = 0$, then a_{j+2} is as given in the right of the following* **Figure 3**.

Proof of Lemma 2.3: The lemma is clear from the construction of the diffeomorphism T in Lemma 2.1.

Using Lemma 2.3, we give the following:

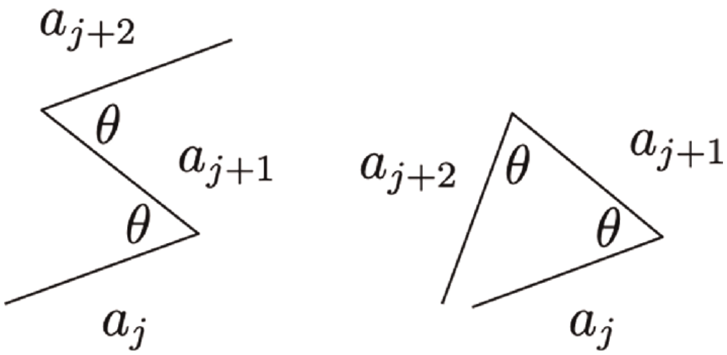


Figure 3.
Left: $\phi_j = \pi$. *Right:* $\phi_j = 0$.

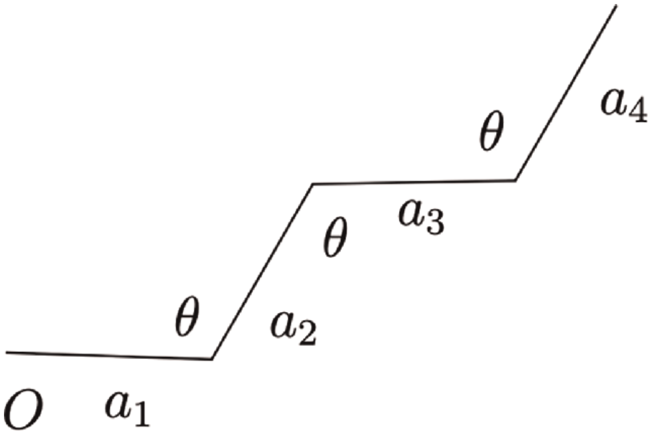


Figure 4.
$P_{4,\theta}$.

Definition 2.4.

i. Consider the point $[\phi_1, \ldots, \phi_{n-2}]$ such that $\phi_i = \pi$ for $1 \leq i \leq n-2$. We denote the unique point by $P_{n,\theta}$. (See the following **Figure 4**.)

ii. Consider the point $[\phi_1, \ldots, \phi_{n-2}]$ such that $\phi_i = 0$ for $1 \leq i \leq n-2$. We denote the unique point by $Q_{n,\theta}$. (See the following **Figure 5**.)

Definition 2.5.

i. We define the function

$$f_{n,\theta} : X_n(\theta) \to \mathbb{R} \tag{18}$$

by

$$f_{n,\theta}(a_1, \ldots, a_n) = \left\| \sum_{i=1}^{n} a_i \right\|^2. \tag{19}$$

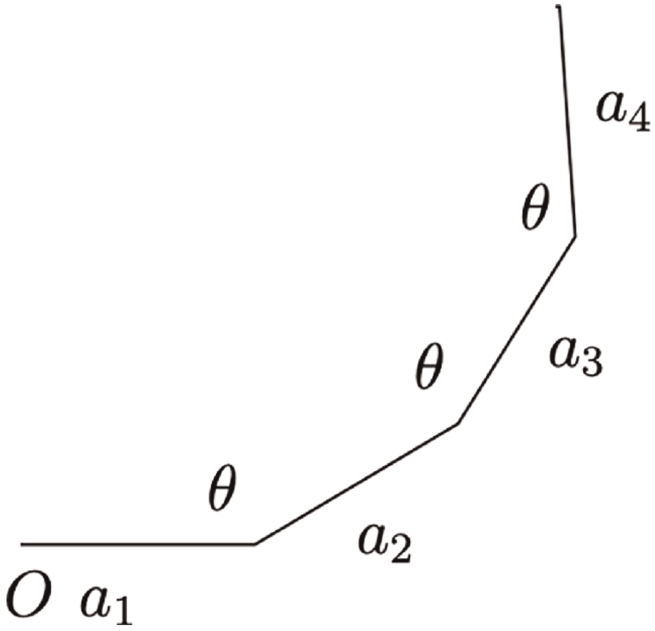


Figure 5.
$Q_{4,\theta}$.

ii. We also define the function

$$d_{n,\theta} : X_n(\theta) \to \mathbb{R} \tag{20}$$

by

$$d_{n,\theta} := \sqrt{f_{n,\theta}}. \tag{21}$$

iii. We also set

$$\ell_{n,\theta} := \min d_{n,\theta}(X_n(\theta)) \quad \text{and} \quad L_{n,\theta} := \max d_{n,\theta}(X_n(\theta)). \tag{22}$$

Remark 2.6. The point $P_{n,\theta}$ in Definition 2.4 (i) is particularly important. In fact, as we see in Proposition 4.1, $P_{n,\theta}$ is a degenerate critical point of$f_{n,\theta}$ when n is even.

The purpose of this chapter is to study the three items in the following:

Problem 2.7. (i) We study whether$f_{n,\theta}$ satisfies the same Morse property as μ_n.

(ii) We obtain explicit formulae for $L_{n,\theta}$ and $\ell_{n,\theta}$.

(iii) We study how the level set $d_{4,\theta}^{-1}(r)$ changes as r moves in $\mathbb{R}$.

Background 2.8. We explain the background for the items in Problem 2.7.

(i) The motivation for Problem 2.7 (i) is explained in Subsection 1.6. More precisely, recall that μ_n in (4) is a function on $\left(S^1\right)^{n-1}$. On the other hand, by Lemma 2.1, we can regard$f_{n,\theta}$ as a function on $\left(S^1\right)^{n-2}$. Then it is natural to ask whether$f_{n,\theta}$ has the same Morse property as μ_n.

(ii) About Problem 2.7 (ii), it is known that a point of $X_n(\theta)$ attains $L_{n,\theta}$ or $\ell_{n,\theta}$. (See Theorem 2.10.) Hence, our task is to obtain explicit formulae for $L_{n,\theta}$ and $\ell_{n,\theta}$.

(iii) About Problem 2.7 (iii), as we stated in Remark 2.6, $P_{4,\theta}$ is a degenerate critical point of $f_{4,\theta}$. Hence, we cannot apply the Morse theory for $f_{4,\theta}$. Instead, we determine the level set $d_{4,\theta}^{-1}(r)$ for each $r \in \mathbb{R}$. In particular, we see from our result that $f_{4,\theta}$ is in fact different from the usual Morse function on $\left(S^1\right)^2$.

We summarize the known results in the following Theorems 2.9, 2.10 and 2.11. First, it is clear that $\mu_n^{-1}(0)$ is always nonempty. On the other hand, we have the following:

Theorem 2.9 [13].

i. *When n is odd, we have*

$$f_{n,\theta}^{-1}(0) = \begin{cases} \varnothing, & \textit{if } 0 < \theta < \dfrac{\pi}{n} \textit{ or } \dfrac{n-2}{n}\pi < \theta < \pi, \\ \{\textit{one point}\}, & \textit{if } \theta = \dfrac{\pi}{n} \textit{ or } \dfrac{n-2}{n}\pi, \\ \textit{a space of dimension } \max\{n-5, 0\}, & \textit{if } \dfrac{\pi}{n} < \theta < \dfrac{n-2}{n}\pi. \end{cases} \tag{23}$$

ii. *When n is even, we have*

$$f_{n,\theta}^{-1}(0) = \begin{cases} \varnothing, & \textit{if } 0 < \theta < \dfrac{n-2}{n}, \\ \{\textit{one point}\}, & \textit{if } \theta = \dfrac{n-2}{n}\pi, \\ \textit{a space of dimension } \max\{n-5, 0\}, & \textit{if } \dfrac{n-2}{n}\pi < \theta < \pi. \end{cases} \tag{24}$$

Proof: Note that $f_{n,\theta}^{-1}(0)$ is the configuration space of equilateral spatial n-gons whose first $(n-1)$ bond angles are θ. In Ref. [13], the space is denoted by $\mathcal{P}_n^{n-1}(\theta)$ and the corresponding results are proved using the notation $\mathcal{P}_n^{n-1}(\theta)$. If we rewrite $\mathcal{P}_n^{n-1}(\theta)$ to $f_{n,\theta}^{-1}(0)$, then we obtain the theorem.

We state our second known result. Recall that $P_{n,\theta}$ and $Q_{n,\theta}$ are defined in Definition 2.4.

Theorem 2.10 [14, 15].

i. The number $L_{n,\theta}$ is attained uniquely by $P_{n,\theta}$.

ii. If $\ell_{n,\theta} > 0$, then $\ell_{n,\theta}$ is attained uniquely by $Q_{n,\theta}$.

Let $\mathcal{M}_4(\theta)$ be the configuration space of equilateral and equiangular squares. In our notation, $\mathcal{M}_4(\theta)$ is defined as

$$\mathcal{M}_4(\theta) = \left\{ (a_1, \dots, a_4) \in X_4(\theta) \;\middle|\; \sum_{i=1}^{4} a_i = (0,0,0) \text{ and } \langle a_4, a_1 \rangle = -\cos\theta \right\}. \tag{25}$$

Then our third known result is the following:

Theorem 2.11 [16]. *We have*

$$\mathcal{M}_4(\theta) = \begin{cases} \varnothing, & \text{if } \frac{\pi}{2} < \theta < \pi, \\ \{\text{one point}\}, & \text{if } \theta = \frac{\pi}{2}, \\ \{\text{two points}\}, & \text{if } 0 < \theta < \frac{\pi}{2}. \end{cases} \tag{26}$$

The following **Figure 6** *illustrates the case* $\theta = \frac{\pi}{2}$. *Note that when* $\theta = \frac{\pi}{2}$ *we have* $\mathcal{M}_4(\theta) = Q_{4,\theta}$, *where* $Q_{4,\theta}$ *is illustrated in* **Figure 5**.

On the other hand, the following **Figure 7** *illustrates the case* $0 < \theta < \frac{\pi}{2}$. *The two figures in* **Figure 7** *are mirror images of each other with respect to the xy-plane.*

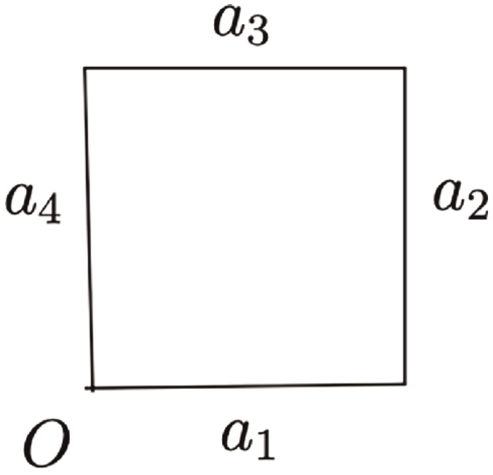


Figure 6.
The element of $\mathcal{M}_4\left(\frac{\pi}{2}\right)$.

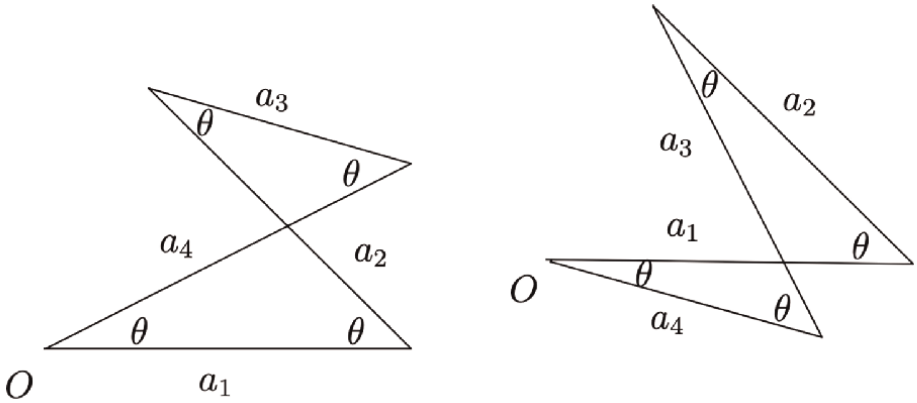


Figure 7.
The two elements of $\mathcal{M}_4(\theta)$ *for* $0 < \theta < \frac{\pi}{2}$.

3. Main results

First, we give an answer to Problem 2.7 (i). (See Example 3.2, Theorems A and B.) Theorem 2.9 tells us that if $n \geq 6$, then $f_{n,\theta}^{-1}(0)$ is a critical manifold of positive dimension. In order to avoid such a manifold, we remove $f_{n,\theta}^{-1}(0)$ from the domain of $f_{n,\theta}$ in the same way as given in (1.3):

Notation 3.1. We abbreviate the following restriction by $g_{n,\theta}$:

$$f_{n,\theta}\Big|_{X_n(\theta)\backslash f_{n,\theta}^{-1}(0)} : X_n(\theta)\backslash f_{n,\theta}^{-1}(0) \to \mathbb{R}. \tag{27}$$

We begin by studying the most elementary case in the following:

Example 3.2. Consider the case $n = 3$. By Theorem 2.9 (i), $f_{3,\theta}^{-1}(0)$ consists of at most finite points. Hence, we may consider $f_{3,\theta}$ for $g_{3,\theta}$. We fix θ and use the diffeomorphism $T : S^1 \to X_3(\theta)$ in Lemma 2.1. Then we can write $f_{3,\theta}\circ T$ as

$$\left(f_{3,\theta}\circ T\right)\left(e^{i\phi_1}\right) = c_1 + c_2 \cos\phi_1 \tag{28}$$

for some $c_1 > 0$ and $c_2 < 0$. Hence for all $\theta, f_{3,\theta}$ is a Morse function with two critical points.

Proof of Example 3.2: First, using (13), we compute a_3. Second, using (19), we compute $\left(f_{3,\theta}\circ T\right)\left(e^{i\phi_1}\right)$. Then we obtain (28). Note that (28) implies that $f_{3,\theta}\circ T$ is essentially the same as the usual Morse function on S^1.

Next, we consider the case $n = 5$. Similarly to Example 3.2, we may consider $f_{5,\theta}$ for $g_{5,\theta}$.

Theorem A.

i. *We set*

$$\alpha{:=}2\ \operatorname{arccot}\left(\sqrt{7}\right)\approx 0.23\pi \quad \textit{and} \quad \beta{:=}2\arctan\left(\sqrt{\frac{3+2\sqrt{5}}{11}}\right)\approx 0.43\pi. \tag{29}$$

Using this, we set

$$\Theta{:=}(0,\pi)\backslash\left\{\frac{\pi}{5}, \alpha, \frac{\pi}{3}, \beta, \frac{3}{5}\pi\right\}. \tag{30}$$

Then $f_{5,\theta}$ is a Morse function if and only if $\theta \in \Theta$.

ii. *For $\theta \in \Theta$, let ν_i be the number of critical points of $f_{5,\theta}$ with index i. Then the four-tuple $(\nu_0, \nu_1, \nu_2, \nu_3)$ is given by the following* **Table 1**.

iii. *In* **Table 1**, *all critical points of index* 0 *attain* $\min f_{5,\theta}(X_5(\theta))$.

Remark 3.3. (i) Consider the case $\frac{\pi}{5} < \theta < \frac{3}{5}\pi$. Then by Theorem 2.9, we have an identification $f_{5,\theta}^{-1}(0) = \{k\,points\}$ for some $k \in \mathbb{N}$. By Theorem A (iii), we have in fact that $k = 4$.

(ii) In **Table 2**, we give a more precise information on $f_{5,\theta}$ for the case θ is the ideal tetrahedral bond angle, i.e., $\theta = \arccos\left(-\frac{1}{3}\right) \approx 109.5°$.

The following theorem is the most crucial result about Problem 2.7 (i).

Theorem B. *Let n be an integer that satisfies $n = 4$ or $n \geq 6$. Then for all θ, $g_{n,\theta}$ is not a Morse function.*

Second, we give an answer to Problem 2.7 (ii). (See Theorems C and D.)

Theorem C.

i. *When n is odd, we have*

$$L_{n,\theta} = \sqrt{\frac{n^2+1}{2} - \frac{n^2-1}{2}\cos\theta}. \tag{31}$$

ii. *When n is even, we have*

$$L_{n,\theta} = n\sin\frac{\theta}{2}. \tag{32}$$

Theorem D.

i. *When $n = 2m + 1$, we have the following results*:

- *When $\frac{n-2}{n}\pi < \theta < \pi$, we have*

θ	$(0, \frac{\pi}{5})$	$(\frac{\pi}{5}, \alpha)$	$(\alpha, \frac{\pi}{3})$	$(\frac{\pi}{3}, \beta)$	$(\beta, \frac{3}{5}\pi)$	$(\frac{3}{5}\pi, \pi)$
$(\nu_0, \nu_1, \nu_2, \nu_3)$	(1,3,3,1)	(4,7,4,1)	(4,9,6,1)	(4,11,8,1)	(4,9,6,1)	(1,5,5,1)

Table 1.
The set $(\nu_0, \nu_1, \nu_2, \nu_3)$.

Critical value	Approximated value	Index	Number of critical points
17	17	3	1
$\frac{121}{9}$	13.44	2	1
$\frac{49+20\sqrt{6}}{9}$	10.88	2	4
$\frac{89}{9}$	9.88	1	3
$\frac{107}{27}$	3.96	1	2
$\frac{1}{81}$	0.01	0	1

Table 2.
The critical points of $f_{5,\theta}$ for $\theta = \arccos\left(-\frac{1}{3}\right)$.

$$\ell_{n,\theta} = -\frac{\cos\left(\frac{\theta}{2} - (m+1)(\theta+\pi)\right)}{\cos\frac{\theta}{2}}. \tag{33}$$

- *When* $\frac{\pi}{n} \leq \theta \leq \frac{n-2}{n}\pi$, *we have*

$$\ell_{n,\theta} = 0. \tag{34}$$

- *When* $0 < \theta < \frac{\pi}{n}$, *we have*

$$\ell_{n,\theta} = (-1)^{m+1} \frac{\cos\left(\frac{\theta}{2} - (m+1)(\theta+\pi)\right)}{\cos\frac{\theta}{2}}. \tag{35}$$

ii. *When* $n = 2m$, *we have the following results:*

- *When* $\frac{n-2}{n}\pi < \theta < \pi$, *we have*

$$\ell_{n,\theta} = -\frac{\sin(m(\theta-\pi))}{\cos\frac{\theta}{2}}. \tag{36}$$

iii. *When* $0 < \theta \leq \frac{n-2}{n}\pi$, *we have*

$$\ell_{n,\theta} = 0. \tag{37}$$

Third, we give an answer to Problem 2.7 (iii). (See Theorem E.) We define the functions $\alpha(\theta)$, $\beta(\theta)$ and $\gamma(\theta)$ as follows:

$$\alpha(\theta) := 4\sin\frac{\theta}{2} \qquad \text{for } 0 < \theta < \pi, \tag{38}$$

$$\beta(\theta) := 2 - 2\cos\theta \qquad \text{for } 0 < \theta < \pi, \tag{39}$$

and

$$\gamma(\theta) := \begin{cases} 2\left(\sin\frac{\theta}{2} - \sin\frac{3\theta}{2}\right), & \text{for } \frac{\pi}{2} < \theta < \pi, \\ -2\left(\sin\frac{\theta}{2} - \sin\frac{3\theta}{2}\right), & \text{for } 0 < \theta \leq \frac{\pi}{2}. \end{cases} \tag{40}$$

Lemma 3.4.

i. The graphs of $\alpha(\theta)$, $\beta(\theta)$ and $\gamma(\theta)$ are given by the following **Figure 8**.

ii. *For* $0 < \theta < \pi$, *we have* $L_{4,\theta} = \alpha(\theta)$.

iii. *We have*

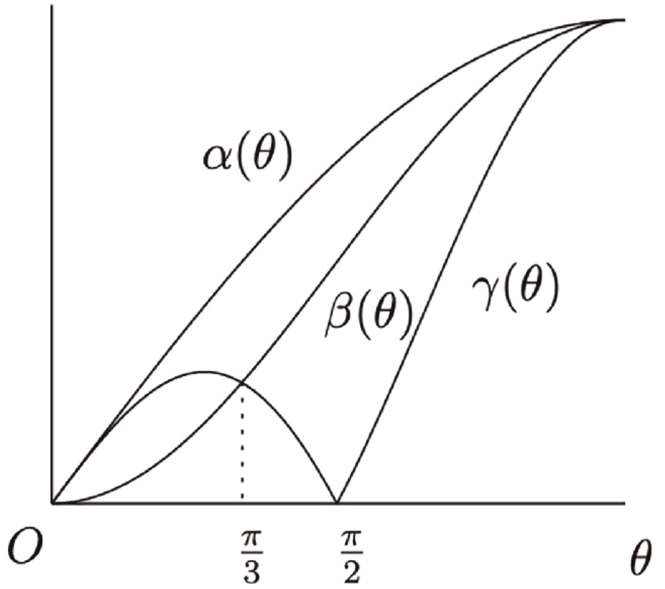


Figure 8.
The graphs of $\alpha(\theta)$, $\beta(\theta)$ and $\gamma(\theta)$.

$$\ell_{4,\theta} = \begin{cases} \gamma(\theta), & \textit{if } \frac{\pi}{2} < \pi < \theta, \\ 0, & \textit{if } 0 < \theta \le \frac{\pi}{2}. \end{cases} \tag{41}$$

Proof of Lemma 3.4: The item (i) is clear from (38), (39) and (40). The items (ii) and (iii) follow immediately from Theorem C (ii) and Theorem D (ii), respectively.

We also define the spaces U, V and W by the following **Figure 9**.

Recall that $P_{4,\theta}$ and $Q_{4,\theta}$ are illustrated in **Figures 4** and **5**, respectively. On the other hand, $\mathcal{M}_4(\theta)$ are illustrated in **Figures 6** and **7**. Then the following Theorem E is the main result on Problem 2.7 (iii).

Theorem E. *The level set $d_{4,\theta}^{-1}(r)$ is given by the following tables.*

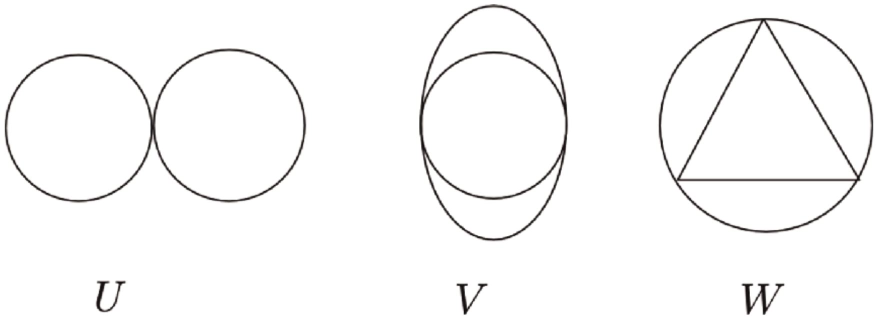


Figure 9.
The spaces U, V and W.

i. *When* $\frac{\pi}{2} \leq \theta < \pi$, *we have* (**Table 3**).

ii. *When* $\frac{\pi}{3} < \theta < \frac{\pi}{2}$, *we have* (**Table 4**)

iii. *When* $\theta = \frac{\pi}{3}$, *we have* (**Table 5**)

iv. *When* $0 < \theta < \frac{\pi}{3}$, *we have* (**Table 6**)

Remark 3.5.

i. For $0 < \theta \leq \frac{\pi}{2}$, it is clear from (25) that $\mathcal{M}_4(\theta) \subset d_{4,\theta}^{-1}(0)$. But **Tables 4–6** imply that the equation $\mathcal{M}_4(\theta) = d_{4,\theta}^{-1}(0)$ in fact holds.

ii. In Theorem E, only the following two points are degenerate critical points of $d_{4,\theta}$: One is $P_{4,\theta}$, which is illustrated in **Figure 4**. The other is $Q_{4,\theta}$ for $\theta = \frac{\pi}{2}$, which is illustrated in **Figure 6**.

iii. When θ approaches $\frac{\pi}{2}$ from below, the three terms $\mathcal{M}_4(\theta)$, $S^1 \coprod S^1$ and U in **Table 4** collapse to $Q_{4,\theta}$ for $\theta = \frac{\pi}{2}$.

r	$\gamma(\theta)$	$(\gamma(\theta), \beta(\theta))$	$\beta(\theta)$	$(\beta(\theta), \alpha(\theta))$	$\alpha(\theta)$
$d_{4,\theta}^{-1}(r)$	$\{Q_{4,\theta}\}$	S^1	V	S^1	$\{P_{4,\theta}\}$

Table 3.
$d_{4,\theta}^{-1}(r)$ for the case $\frac{\pi}{2} \leq \theta < \pi$.

r	0	$(0, \gamma(\theta))$	$\gamma(\theta)$	$(\gamma(\theta), \beta(\theta))$	$\beta(\theta)$	$(\beta(\theta), \alpha(\theta))$	$\alpha(\theta)$
$d_{4,\theta}^{-1}(r)$	$\mathcal{M}_4(\theta)$	$S^1 \coprod S^1$	U	S^1	V	S^1	$\{P_{4,\theta}\}$

Table 4.
$d_{4,\theta}^{-1}(r)$ for the case $\frac{\pi}{3} < \theta < \frac{\pi}{2}$.

r	0	$(0, \beta(\theta))$	$\beta(\theta)$	$(\beta(\theta), \alpha(\theta))$	$\alpha(\theta)$
$d_{4,\theta}^{-1}(r)$	$\mathcal{M}_4(\theta)$	$S^1 \coprod S^1$	W	S^1	$\{P_{4,\theta}\}$

Table 5.
$d_{4,\theta}^{-1}(r)$ for the case $\theta = \frac{\pi}{3}$.

r	0	$(0, \beta(\theta))$	$\beta(\theta)$	$(\beta(\theta), \gamma(\theta))$	$\gamma(\theta)$	$(\gamma(\theta), \alpha(\theta))$	$\alpha(\theta)$
$d_{4,\theta}^{-1}(r)$	$\mathcal{M}_4(\theta)$	$S^1 \coprod S^1$	V	$S^1 \coprod S^1$	U	S^1	$\{P_{4,\theta}\}$

Table 6.
$d_{4,\theta}^{-1}(r)$ for the case $0 < \theta < \frac{\pi}{3}$.

4. Submanifolds of $X_n(\theta)$

In order to prove Theorem B, we consider the following three submanifolds of $X_n(\theta)$:

- The first submanifold is the point $P_{n,\theta}$. (About $P_{n,\theta}$, see Definition 2.4 (i) for the definition and Proposition 4.1 for the property.)

- The second submanifold is $V_n(\theta)$ for odd n. (About $V_n(\theta)$, see Definition 4.3 for the definition, and Lemma 4.4 and Proposition 4.5 for the property.)

- The third submanifold is Σ_n, which is a point of $X_n(\theta)$ for the case $n = 2m + 1$, $n \equiv 3 (\mathrm{mod} 4)$ and $\theta = \arccos\left(\frac{m}{m+1}\right)$. (About Σ_n, see Definition 4.6 for the definition, and Lemma 4.7 and Proposition 4.8 for the property.)

Proposition 4.1. *Let n be an even number that is greater than or equal to* 4*. Then* $P_{n,\theta}$ *is a degenerate critical point of* $g_{n,\theta}$*, where* $P_{n,\theta}$ *is defined in Definition 2.4 (i).*

Remark 4.2. When n is an odd number, $P_{n,\theta}$ is a nondegenerate critical point.

Proofs of Proposition 4.1 and Remark 4.2: We prove by computing $f_{n,\theta} \circ T$, where T is constructed in Lemma 2.1. Recall from Theorem 2.10 that the global maximum of $f_{n,\theta} \circ T$ is attained by $\left(e^{i\phi_1}, \dots, e^{i\phi_{n-2}}\right)$, where $\phi_i = \pi$ for $1 \le i \le n-2$. Let $H\left(f_{n,\theta} \circ T\right)\left(e^{i\pi}, e^{i\pi}, \dots, e^{i\pi}\right)$ be the Hessian matrix of $f_{n,\theta} \circ T$ at the point. We set

$$K_n := \det\left(H\left(f_{n,\theta} \circ T\right)\left(e^{i\pi}, e^{i\pi}, \dots, e^{i\pi}\right)\right). \tag{42}$$

Then it is easy to see that

$$K_n = \begin{cases} -2^{n-2} \cdot \left(\left(\frac{n-3}{2}\right)!\right)^2 \cdot (\sin\theta)^{2n-4}, & \text{if } n \text{ is odd,} \\ 0, & \text{if } n \text{ is even.} \end{cases} \tag{43}$$

Proposition 4.1 and Remark 4.2 follow from (43).

Definition 4.3. Let $n = 2m + 1$ be an odd number which is greater than or equal to 7. We define the subspace $V_n(\theta)$ of $X_n(\theta)$ as follows.

(i) If $n \equiv 1 (\mathrm{mod} 4)$, then we set

$$V_n(\theta) := \{[\phi_1, \dots, \phi_{n-2}] \mid \text{the following conditions hold}\}, \tag{44}$$

where.

- $\phi_1 = \pi$, $\phi_2 = 0$ and $\phi_3 = \pi$.

- ϕ_4 is a variable in $[0, 2\pi]$.

- $\phi_{m+2} = 0$.

- $\phi_i = \pi$ for $5 \le i \le m+1$ or $m+3 \le i \le n-2$.

(ii) If $n \equiv 3 \pmod 4$, then we set

$$V_n(\theta) := \{[\phi_1, \dots, \phi_{n-2}] \mid \text{the following conditions hold}\}, \tag{45}$$

where

- $\phi_1 = 0$ and $\phi_2 = \pi$.
- ϕ_3 is a variable in $[0, 2\pi]$.
- $\phi_{m+2} = 0$.
- $\phi_i = \pi$ for $4 \leq i \leq m+1$ or $m+3 \leq i \leq n-2$.

Lemma 4.4. *Let $n = 2m+1$ be an odd number which is greater than or equal to 7. Then the following items hold.*

i. For all θ, $V_n(\theta)$ is a critical manifold of $f_{n,\theta}$. Moreover, $V_n(\theta)$ is diffeomorphic to S^1.

ii. *The following equation holds.*

$$f_{n,\theta}(V_n(\theta)) = \begin{cases} ((m+1) - m\cos\theta)^2, & \text{if } m \text{ is even,} \\ \Big((m - (m+1)\cos\theta)^2, & \text{if } m \text{ is odd.} \end{cases} \tag{46}$$

iii.

a. *If $n \equiv 1 \pmod 4$, then for all θ, $V_n(\theta)$ is a subspace of $X_n(\theta) \backslash f^{-1}_{n,\theta}(0)$.*

b. *Assume that n satisfies $n \equiv 3 \pmod 4$. Then except for the case $\theta = \arccos\left(\frac{m}{m+1}\right)$, $V_n(\theta)$ is a subspace of $X_n(\theta) \backslash f^{-1}_{n,\theta}(0)$.*

Proof of Lemma 4.4: First, we prove the item (i). By direct computations, it is easy to prove that $V_n(\theta)$ is a critical manifold of $f_{n,\theta}$. Moreover, in Definition 4.3 (i), only ϕ_4 is the parameter for $V_n(\theta)$. Hence, $V_n(\theta)$ is diffeomorphic to S^1. Similarly, in Definition 4.3 (ii), only ϕ_3 is the parameter for $V_n(\theta)$. Hence, $V_n(\theta)$ is also diffeomorphic to S^1.

We can prove the item (ii) from direct computations. The item (iii) follows immediately from the item (ii).

As a corollary of Lemma 4.4, we have the following:

Proposition 4.5. *Let $n = 2m+1$ be an odd number which is greater than or equal to 7. Then except the case that $n \equiv 3 \pmod 4$ and $\theta = \arccos\left(\frac{m}{m+1}\right)$, $g_{n,\theta}$ is not a Morse function.*

Proof: By Lemma 4.4 (i) and (iii), $V_n(\theta)$ is a critical manifold of $g_{n,\theta}$ of positive dimension. Since nondegenerate critical points are isolated, the proposition follows.

In order to consider the remaining case other than Proposition 4.5, we give the following:

Definition 4.6. Let $n = 2m + 1$ be an odd number which is greater than or equal to 7 and $n \equiv 3 (\mathrm{mod}\, 4)$. We set

$$\theta_0 := \arccos\left(\frac{m}{m+1}\right). \tag{47}$$

We define the point Σ_n of $X_n(\theta_0)$ as follows.

$$\Sigma_n := \{[\phi_1, \dots, \phi_{n-2}] \mid \text{the following conditions hold}\}, \tag{48}$$

where

- $\phi_i = \pi$ for $1 \le i \le m - 1$.
- $\phi_m = 0$.
- $\phi_i = \pi$ for $m + 1 \le i \le n - 4$.
- $\phi_{n-3} = \arccos\left(\frac{1}{n}\right)$.
- $\phi_{n-2} = -\arccos\left(\frac{1}{n}\right)$.

(See the following **Figure 10**.)

Lemma 4.7. *We have*

$$f_{n,\theta_0}\left(\Sigma_n\right) = \frac{16}{n+1}. \tag{49}$$

Proof of Lemma 4.7: Similarly to the proof of Lemma 4.4 (ii), we can prove the lemma from direct computations.

Note that Lemma 4.7 tells us that Σ_n is a point of $X_n(\theta_0) \setminus f^{-1}_{n,\theta_0}(0)$.

Proposition 4.8. *Let n and θ_0 be as given in Definition 4.6. Then the point Σ_n is a degenerate critical point of g_{n,θ_0}.*

Proof of Proposition 4.8: We can prove the proposition from easy computations in calculus.

Remark 4.9. We write Σ_n in (48) as $\Sigma_n = (a_1, \dots, a_n)$. Then the pair

$$(\phi_{n-3}, \phi_{n-2}) = \left(\arccos\left(\frac{1}{n}\right), -\arccos\left(\frac{1}{n}\right)\right) \tag{50}$$

is defined to satisfy

$$\sum_{i=n-3}^{n} a_n = 0. \tag{51}$$

But if we generalize Σ_n with respect to general θ so as to satisfy (51), the generalized point is not a critical point of $g_{n,\theta}$.

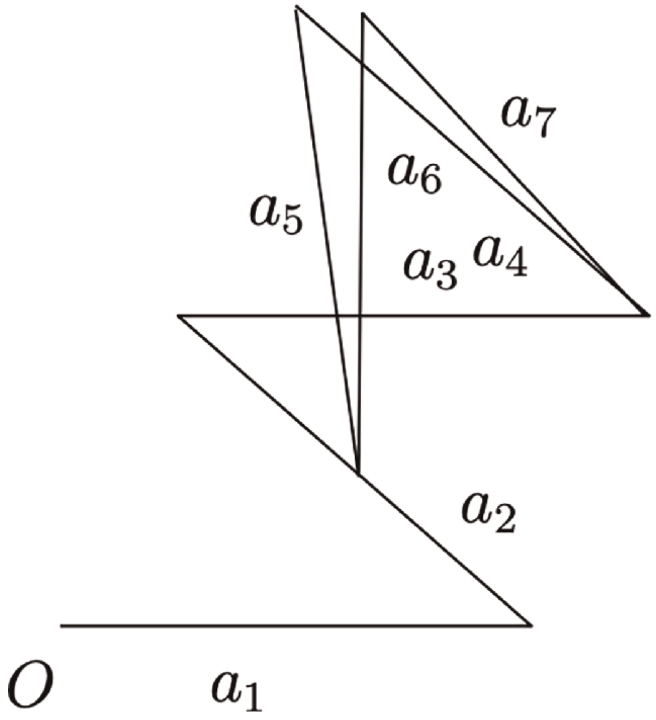


Figure 10.
The point Σ_7, where only a_i $(1 \leq i \leq 5)$ are planar vectors.

5. Proofs of the main theorems

Proof of Theorem A: First we prove the item (i). As in the proof of Proposition 4.1, we prove by computing$f_{5,\theta} \circ T$. By direct computations, we see that a degenerate critical point of$f_{5,\theta} \circ T$ has the form $\left(e^{i\phi_1}, e^{i\phi_2}, e^{i\phi_3}\right)$ for some

$$(\phi_1, \phi_2, \phi_3) \in \{0, \pi\} \times \{0, \pi\} \times \{0, \pi\}. \tag{52}$$

We compute the eigenvalues of the Hessian matrix of$f_{5,\theta} \circ T$ at $\left(e^{i\phi_1}, e^{i\phi_2}, e^{i\phi_3}\right)$, where (ϕ_1, ϕ_2, ϕ_3) is as given in (52). An eigenvalue is a function on θ and we determine θ for which an eigenvalue is 0. Then we obtain (i).

The items (ii) and (iii) are proved in the process of proving the above item (i).

Proof of Theorem B: The theorem follows immediately from Propositions 4.1, 4.5 and 4.8.

Proof of Theorem C: By Theorem 2.10 (i), $L_{n,\theta}$ is attained uniquely by $P_{n,\theta}$. Let a_1 and a_2 be as given in (9). If $n = 2m + 1$, then

$$L_{n,\theta} = \|(m+1)a_1 + ma_2\|. \tag{53}$$

Simplifying the right-hand side, we obtain Theorem C (i). Theorem C (ii) can be proved similarly.

Proof of Theorem D: By Theorem 2.9, the range of θ for which $\ell_{n,\theta} = 0$ is known. Hence, we need to obtain an explicit formula of $\ell_{n,\theta}$ when $\ell_{n,\theta} > 0$. Recall from Theorem 2.10 (ii) that when $\ell_{n,\theta} > 0$, it is attained uniquely by $Q_{n,\theta}$. We set

$$A{:=}\begin{pmatrix} -\cos\theta & -\sin\theta \\ \sin\theta & -\cos\theta \end{pmatrix} \tag{54}$$

Then similarly to (53), we have the following formula:

$$\ell_{n,\theta} = \left\| \sum_{i=0}^{n-1} A^i \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right\| \tag{55}$$

Simplifying the right-hand side, we obtain Theorem D.

Proof of Theorem E: Since $d_{4,\theta}^{-1}(r)$ is a curve, it is easy to draw its shape. Then Theorem E follows.

6. Conclusions

Consider the case $n = 5$. In Ref. [11], $d_{5,\theta}^{-1}(1)$ is denoted by $C_6(\theta)$ and its topological type was determined as per the following **Table 7**, where following the Schläfli symbol, $\{6\}$ denotes the regular hexagon.

In Ref. [11], **Table 7** was proved as follows: Let $R : C_6(\theta) \to S^1$ be a certain map. For each $\xi \in S^1$, we determine the topological type the level set $R^{-1}(\xi)$. Then we obtain **Table 7**.

Below, we indicate that there is a chance to reprove **Table 7** alternatively: For example, we consider the case $\theta = \arccos\left(-\frac{1}{3}\right)$. Then applying the Morse lemma to **Table 2**, we obtain the diffeomorphism

$$C_6\left(\arccos\left(-\frac{1}{3}\right)\right) \cong S^2. \tag{56}$$

If we construct a similar table to **Table 2** for various θ, then we can reprove **Table 7**. This is our further problem.

θ	$\left(0, \frac{\pi}{3}\right)$	$\left(\frac{\pi}{3}, \frac{\pi}{2}\right)$	$\left(\frac{\pi}{2}, \frac{2}{3}\pi\right)$	$\frac{2}{3}\pi$	$\left(\frac{2}{3}\pi, \pi\right)$
Topological type	$\underset{3}{\#}(S^1 \times S^1)$	$\underset{3}{\#}(S^1 \times S^1)$	S^2	$\{6\}$	$\varnothing$

Table 7.
The topological type of $C_6(\theta)$.

Author details

Yasuhiko Kamiyama
Faculty of Science, Department of Mathematical Sciences, University of the Ryukyus,
Nishihara-Cho, Okinawa, Japan

*Address all correspondence to: kamiyama@sci.u-ryukyu.ac.jp

References

[1] Ali ZA, Han Z. Maneuvering control of hexrotor UAV equipped with a cable-driven gripper. IEEE Access. 2021;**9**: 65308-65318. DOI: 10.1109/ ACCESS.2021.3076129. Available from: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9417166 [Accessed: 1 September 2023]

[2] Ali ZA, Li X. Modeling and controlling of quadrotor aerial vehicle equipped with a gripper. Measurement and Control. 2019;**52**:577-587. DOI: 10.1177/0020294019834040. Available from: https://journals.sagepub.com/doi/epub/10.1177/ 0020294019834040 [Accessed: 1 September 2023]

[3] Ali ZA, Li X. Modeling and controlling the dynamic behavior of an aerial manipulator. Fluctuation and Noise Letters. 2021;**20**:2150044. DOI: 10.1142/S0219477521500449. Available from: https://www.worldscientific.com/doi/abs/10.1142/ S0219477521500449?journalCode=fnl [Accessed: 1 September 2023]

[4] Farber M. Invitation to topological robotics. In: Zurich Lectures in Advanced Mathematics. Zurich: European Mathematical Society; 2008. p. 143. DOI: 10.4171/054

[5] Hausmann J-C. Sur la topologie des bras articulés. In: Algebraic topology Poznań 1989. Springer Lecture Notes in Mathematics. 1989;**1474**:146-159. DOI: 10.1007/BFB0084743

[6] Walker K. Configuration Spaces of Linkages. NJ: Princeton; 1985. Available from: https://canyon23.net/math/1985thesis.pdf [Accessed: 1 September 2023]

[7] Kapovich M, Millson J. On the moduli space of polygons in the Euclidean plane. Journal of Differential Geometry. 1995; **42**:430-464. DOI: 10.4310/jdg/ 1214457237. Available from: https:// projecteuclid.org/journals/journal-of-differential-geometry/volume-42/ issue-2/On-the-moduli-space-of-polygons-in-the-Euclidean-plane/ 10.4310/jdg/1214457237.full [Accessed: 1 September 2023]

[8] Farber M, Schütz D. Homology of planar polygon spaces. Geometriae Dedicata. 2007;**125**:75-92. DOI: 10.1007/ s10711-007-9139-7. Available from: https://link.springer.com/content/pdf/ 10.1007/s10711-007-9139-7.pdf [Accessed: 1 September 2023]

[9] Demaine E, Langerman S, O'Rourke J. Geometric restrictions on producible polygonal protein chains. Algorithmica. 2006;**44**:167-181. DOI: 10.1007/ s00453-005-1205-7. Available from: https://link.springer.com/article/ 10.1007/s00453-005-1205-7 [Accessed: 1 September 2023]

[10] Soss M, Toussaint G. Geometric and computational aspects of polymer reconfiguration. Journal of Mathemtical Chemistry. 2000;**27**:303-318. DOI: 10.1023/A:1018823806289. Available from: https://link.springer.com/article/10.1023/A:1018823806289 [Accessed: 1 September 2023]

[11] Kamiyama Y. The Topology of the Configuration Space of a Mathematical Model for Cycloalkenes: Advanced Topics of Topology. London, UK: IntechOpen; 2022. DOI: 10.5772/ intechopen.100723. Available from: https://www.intechopen.com/chapters/ 79299 [Accessed: 1 September 2023]

[12] Kamiyama Y. The differential structure on the configuration space of a mathematical model for cycloalkenes. JP

Journal of Geometry and Topology. 2022;**27**:11-31. DOI: 110.17654/0972415X22002. Available from: http://www.pphmj.com/abstract/14276.htm [Accessed: 1 September 2023]

[13] Kamiyama Y. A filtration of the configuration space of spatial polygons. Advanced Applied Discrete Mathematics. 2019;**22**:67-74. DOI: 10.17654/DM022010067. Available from: https://www.researchgate.net/publication/336117647_a_filtration_of_the_configuration_space_of_spatial_polygons [Accessed: 1 September 2023]

[14] Benbernou N. Fixed-Angle Polygonal Chains: Locked Chains and the Maximum Span. MA: Smith College; 2006. Available from: http://people.csail.mit.edu/nbenbern/thesis.pdf [Accessed: 1 September 2023]

[15] Benbernou N, O'Rourke J. On the maximum span of fixed-angle chains. In: 18th Canadian Conference on Computational Geometry. Kingston; 2006. pp. 14-16. Available from: http://people.csail.mit.edu/nbenbern/MaxSpan.CCCG.pdf [Accessed: 1 September 2023]

[16] Crippen G. Exploring the conformation space of cycloalkanes by linearized embedding. Journal of Computational Chemistry. 1992;**13**:351-361. DOI: 10.1002/jcc.540130308. Available from: https://www.researchgate.net/publication/30839993_Exploring_the_conformation_space_of_cycloalkanes_by_linearized_embedding [Accessed: 1 September 2023]

Chapter 4

Multi-Agent Robot Motion Planning for Rendezvous Applications in a Mixed Environment with a Broadcast Event-Triggered Consensus Controller

Nohaidda Sariff, Zool Hilmi Ismail,
Ahmad Shah Hizam Md Yasir, Denesh Sooriamoorthy and
Puteri Nor Aznie Fahsyar Syed Mahadzir

Abstract

Finding consensus is one of the most important tasks in multi-agent robot motion coordination research, especially in a communication environment. This justification underlies the use of event-triggered controller in current multi-agent consensus research. However, the communication issue has not been adequately addressed in a broadcast communication environment for rendezvous applications. Therefore, the broadcast event-triggered (BET) controller with a new formulation was designed using the Simultaneous Perturbation Stochastic Algorithm (SPSA). Theorems and relevant proofs were presented. Agent performances with the BET controller were evaluated and compared with the conventional broadcast time-triggered (BTT) controller. The results showed an effective motion generated by a multi-agent robot to reach the rendezvous point based on the Bernoulli distribution and gradient approximation of the agent local controller. The BET controller has proven to work more efficiently than the BTT controller when it reaches convergence in less than 40.42% of time and 21.00% of iterations on average. The utilization of communication channels is slightly reduced for BET, which is 71.09% usage instead of fully utilized by BTT. The threshold value of the event-triggered function (ETF) and SPSA parameters affected agent performances. Future research may consider using an effective and efficient BET controller in a complex communication environment with many variations of graph topology networks.

Keywords: multi-agent robot system, motion coordination, motion planning, broadcast event-triggered, rendezvous

1. Introduction

1.1 Motivation of the research

The effectiveness and robustness of multi-agent robot systems (MARS) [1–4] in carrying out tasks as compared to single agent [5–7] has led to an expansion in the MARS research. The advantages of having MARS that may increase the flexibility and scalability as well as reduce load among agents, makes cooperative MARS research still active until present day. Example of applications with MARS such as medical robots known as the nano [8] and magnetic robot [9] have been used to send medicine directly to the human organ, inspection robots to inspect and clean pipelines in oil and gas industries [10, 11], planetary rovers [12, 13], unmanned aerial vehicles [14–18], and swarm robots [19, 20]. Due to the importance of MARS in assisting human activities, several researchers have been actively reviewing and discussed a variety of issues related to cooperative MARS [2, 21, 22] such as formation [23–27], consensus [28–30], containment [31–33], tracking [34], and rendezvous [29, 35–37].

One of the issues that researchers pay a lot of attention to in motion coordination of MARS is consensus subject to the difficulties of agent to find an agreement between agents to a certain degree [22, 38, 39]. Various researchers proposed to develop an effective communication and control system to be used practically in MARS consensus applications. Issues related to limited communication resources [40, 41], time delay [42–44], and disturbances [45–47] are among the communication problems which might affect an agent to reach consensus.

Finding strategies to limit the usage of agent resources such as communication [48–52] and energy [16, 53–56] has been emphasized in consensus research. This is to ensure that the consensus controller is feasible and practical to be used. Therefore, minimizing communication usage and energy resources is not an option in consensus, especially when the agent is embedded with constrained controller board resources. As a result, an event-based system [35, 47, 57–59] has been used to reduce communication in terms of the number of communication channels utilised for transmission while also preserving bandwidth coverage. Other examples of research concentrating on decreasing energy resources from trajectory [52] and actuator [53] are energy-awareness or energy-efficiency [16, 60–62]. The lifespan of a MARS can be increased by conserving resources, which also increases the significance of this research for MARS in practical applications.

In addition, a hybrid controller was developed to provide a robust system that can ensure that the consensus task may be accomplished effectively. Sliding mode controller [46], fuzzy logic techniques [63–69], model predictive control [70], distributed control [49, 63, 71–74], and dynamic role assignment [75] are a few examples of hybrid controllers that have been used for multi-agent robot formation, rendezvous, and path planning [76–81] applications. In order to identify the consensus system efficiently and conserve the agent's communication and resources, these controllers have been coupled with an event-based system [45, 70, 71, 82, 83].

In relation to wireless network technology, there is a need for a consensus study since the network topologies exist between the agent and its neighbours [29, 73, 84, 85]. For instance, the communication between the UAV ground mobile robot via the network on the air and the ground [86], the communication between the heterogeneous UAV and the satellite and ground station [87], and the communication between the cluster multi-agent and the network topology [88]. Since the agents are connected in a network topology of information flow either directed [89, 90] or undirected [91, 92], one way or two-way communications, one-to-one or broadcast [25, 31, 58, 93–98], finding consensus can therefore be considered challenging due to the complexity of the network.

The importance of MARS motion coordination consensus research was proven from the above discussion. When communication problems arise in the broadcast and communication contexts, the agent's performances such as utilization of agent resources and energy levels may be impacted. This served as the impetus for putting up a remedy and plan to create a successful and effective consensus control and communication system. Although some studies on event-triggered have been proposed for consensus research studies such as average consensus [58, 59, 73, 85, 99–102], leader follower consensus [103], and other consensus or rendezvous control systems [29, 35, 36, 46, 63, 70, 71, 82, 88, 103–112], research on resolving the communication problem for the application of rendezvous in a broadcast communication environment has yet to be explored.

1.2 Related work

The validity of the MARS motion coordination research was proven when several researchers kept improvising the control and communication systems, especially for consensus problems. For better understanding, this section will discuss the two main important findings found in the literature, which have been divided into two categories: the development of MARS consensus control and event-based consensus.

1.2.1 Multi-agent robot system consensus

The consensus of multi-agents is among the issues that have attracted many researchers' attention in the past few years [22, 38]. An agreement between agents can be achieved via an appropriate communication and control system to reach certain quantities of interest. Therefore, various control and communication issues related to multi-agent consensus have been discussed to show the recent issues in multi-agent consensus. The development of consensus control depends on the issues solved for every single application. Some focus on velocity and positional consensus issues. The consensus on velocity leads to a collective behaviour where the agents can move with a common velocity, while the consensus on position drives all agents to a common position. Formation [27], tracking [113], and rendezvous [114, 115] were among the consensus applications being focused on by researchers.

Since communication will determine the realization of consensus, a recent development of multi-agent consensus research focusing on communication issues such as time delay [42, 112], disturbances [45, 71], and limited resources [64, 70] has been evident. Therefore, an alternative to broadcast [113–117] and agent-to-agent [102, 118] communication has been introduced, as well as sampling methods such as time-triggered, periodic, aperiodic, or event-triggered. However, the MARS consensus for rendezvous applications was extended in this research. Applied to a group of leaderless agents with a double integrator system, the issue of communication among agents in a broadcast communication environment has been proposed to determine the realization of multi-agent consensus for practical rendezvous applications. The main objective is to find positional consensus among the multi-agents with zero velocities and acceleration when the rendezvous point is achieved.

1.2.2 Event-based consensus

The idea of event-based sampling was proposed by Astrom et al. [119] in early 1999, while many researchers applied the multi-agent control issue [38, 81, 106, 107]. The objective of having ET is to save communication resources and computation, where the sample data will be sent to the agent and the control update will only occur when it meets the event-triggered function (ETF).

The evolution of event-based control shows that it has been applied initially with a single integrator or order [110], followed by a second order [118], and more recently in a fourth order control system [83] for MARS. Based on positive research outcomes, the ET has thus been recently extended with other intelligent control systems such as Sliding Mode Control [70], adaptive fuzzy [64], and Model Predictive Control [69]. These controllers were integrated with an event-based system, which guaranteed the efficiency of the tracking, formation, and positioning systems in terms of resource utilization.

1.3 Contributions

The broadcast controller has been combined with the event-triggered controller using simultaneous perturbation stochastic algorithm (SPSA) for the rendezvous application for MARS which is the main contribution of this paper [113, 114]. The agents should be able to agree on the rendezvous point while maintaining their performance in terms of convergence time and iteration, trajectory pattern, and number of communication channels. The proposed theorems and proofs are well presented and discussed. Besides that, the broadcast time-triggered (BTT) system, known as a traditional sample technique [95, 97, 98], has been used in this case study as a comparison to the broadcast event-triggered (BET) control system. The effectiveness and efficiency of a new BET control system are evaluated and proven based on the results obtained. Lastly, the effects of SPSA parameter and ET controller changes on agent performances were observed and analyzed. This will be yet another contribution that illustrates the relationship between these hybrid BET controller parameters and agent outputs within the context of the broadcast and communication environment.

1.4 Organization of the chapter

The sections of this paper are structured as follows: the first section and second section presents the introduction and formulation of the problem, the third section focuses on the design of a new BET controller with discussion on related theorems and proof, the fourth section explains the effectiveness and efficiency evaluation of multi-agent robot performances with BET and BTT, the fifth section explains the SPSA and ETF effects on agent performances, and the final section is the conclusion with some recommendations for MARS motion coordination research direction.

2. Problem formulation

Assume a dynamic linear group of agents i connected with neighbours j in an undirected topology network located within the broadcast communication

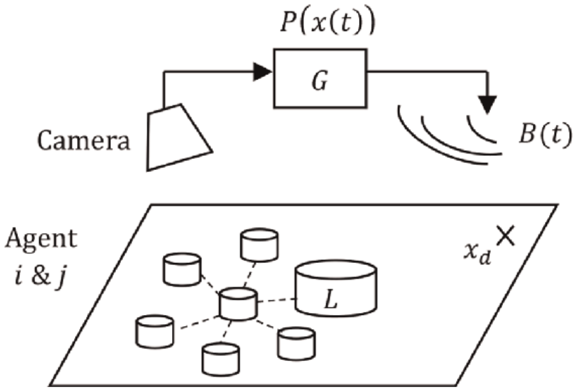


Figure 1.
Multi-agent robot working environment.

environment as shown in **Figure 1**. The agent has to communicate among themselves to reach consensus to rendezvous point x_d while continuously receiving global feedback $B(t)$ from global controller . This is to satisfy the motion coordination task in a such a way that the value of $P(x(t)) = 0$ which shows the convergence achieved as derived from Eqs. (1) and (2).

Referring to this environment, the agents have limited knowledge of its environment whereby the system does not provide any global coordinates and the agents will rely more on their relative position during communication. Due to this situation, it may cause high utilization of communication resources such as channel and bandwidth required when the agent must communicate continuously to reach consensus to the desired rendezvous point. This might get worse if the number of agents increased and the location of the target is quite far from the agent. The communication issue must be considered as it might affect the effectiveness and the efficiency of the agent, especially when the agent is supplied with limited power from the microcontroller board. The agent's performances such as time and iteration are taken as well as trajectory will get effected until convergence is achieved. While various problems had been solved in the scope of agent in broadcast and communication environment such as quantization [97], collision avoidance [115], instability [95], consensus [98, 116], and event-triggered [35, 64, 117, 118], the communication problem within the range of broadcast and communication environment still remains unsolved.

$$P(x) = \sum_{i=1}^{N} ||x_i - x_d|| \tag{1}$$

$$\lim_{t\to\infty} P(x(t)) = 0 \tag{2}$$

3. Broadcast event-triggered controller design

Notation: Denote R is the real number, R_+ is a set of positive real numbers, n is sample size, and N is the population size. The state position of agent in Euclidean coordinates is represented by vector while x is expressed by $x_i(t) = |x_1, x_2, x_3 \ldots .x_n|^T \in R^n$. The function represented by$f(x)$, β SPSA random and deterministic movement, α state function, and γ ETF. Assume e is the measurement error, L_i local controller, G

global controller, B broadcast signal $\in R, A_i$ agent number, $P(x(t))$ is performances index measurement, $t \in N$ is discrete time, $(t+1) \in N$ is next discrete time, and $u_{i,j,R,D1,D2}$ is control input. The agent i is connected with its own neighbor set $j \in N_i$ in undirected graph $G = (V, E)$ where V is vertex represent the agent and $E$is the edge of each vertex represent the connection between i and j.

The overall design of this controller is based on the SPSA where the unknown gradient, which is the state position, will be generated using simultaneous perturbation approximation as shown in **Figure 2**. Measurement of two objective functions is required in this process instead of the overall dimension of agent state position. Firstly, the digital camera will capture the image of the agent's position and update $x_i(t)$ in the system. The global controller G compute the performances index $P(x(t))$ of the scalar value which indicates the distance between the agent's and rendezvous point to send $B(t)$ to the local controller L_i. The local controller then computes agent control input $u_i(t)$ based on random Bernoulli distribution error and gradient approximation with event-triggered error as shown on the right side of **Figure 2**. Lastly, each agent A_i will update the next position $x_i(t+1)$ based on the latest value of control input. This process continuously iterates until all agents reach consensus at the rendezvous point. When the system reaches convergence, agent performances will be evaluated and recorded such as convergence time and iteration, number of channels (NOCs), as well as agent trajectory to determine the effectiveness and efficiency of the system. A detailed description of agent state position, global controller, local controller, ETF, and agent motion are explained in the section below.

Agent state position, x_i

The agent state position of $x_i(0) = [x_1(0), x_2(0), x_3(0) \ldots] \in R^n$ depends on the image from the digital camera. A collective agent position in Cartesian coordinates can be represented as the Euclidean norm of the vector $x_i(t) = |x_1(t), x_2(t), x_3(t)| \in R^n$. These positions update will be continuously sent to the global controller until the agent meets the rendezvous point.

Global controller, G

The performances of agent were measured based on the degree of agent reaching consensus indicated by the distance between the agent with the rendezvous point as presented by Eqs. (1) and (3). $B(t) \in R$ represents the output of global controller, $x(t) \in R^{nN}$ is the collective state position of agents, and $P(x) \in R$ is the objective function of rendezvous.

$$G : B(t) = P(x(t)) \tag{3}$$

Local controller, L_i

The local controller or known as distributed controller will determine the agent movement $A_i$at every even and odd t time based on the input from broadcast signal $B(t) \in R$. The variables of the control system are represented by a column vector as shown in Eq. (4) where $\delta_{i1}(t)$ is the state position, $\delta_{i2}(t)$ is the broadcast signal, $\delta_{i3}(t)$ is the even movement and $\delta_{i4}(t)$ is the odd movement. Three main functions which might affect the output of $u_i(t) \in R$ (Eq. (5)) are $\alpha : R^v \times R \to R^v$ is the state function, $\beta : R^v X\ R \to R^v$ is the random and deterministic function of SPSA, and $\gamma : R^v \times R \to R^v$ is the standard consensus protocol with ET as stated in Eqs. (6)–(8). As a result, the random error of Bernoulli represented by β even and the deterministic error of SPSA approximation with communication error of ET represented by β odd will determine the agent's movement as shown in **Figures 3** and **4**.

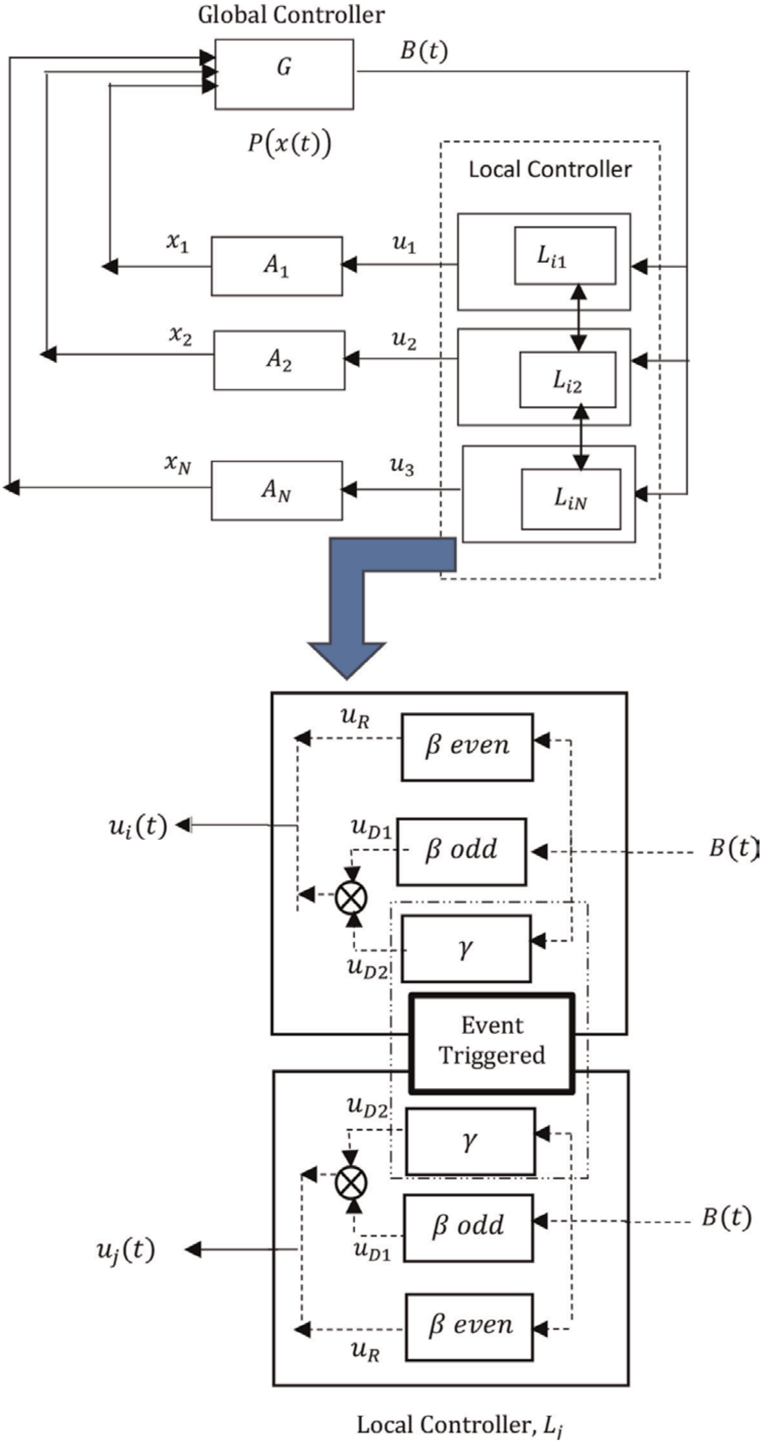


Figure 2.
BET controller design.

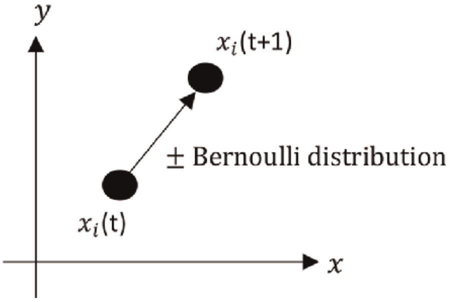


Figure 3.
Even movement (random).

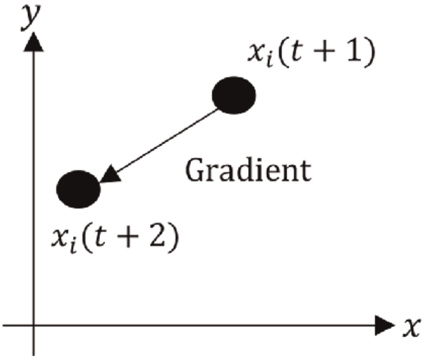


Figure 4.
Odd movement (deterministic).

$$\delta_i(t) = \begin{bmatrix} \delta_{i1}(t) \\ \delta_{i2}(t) \\ \delta_{i3}(t) \\ \delta_{i4}(t) \end{bmatrix} \in R^n \times R \times R \times R \tag{4}$$

$$L_i : \begin{cases} \delta_i(t+1) = \alpha(\delta_i(t), B(t)) \\ u_i(t) = \beta(\delta_i(t), B(t)) + \gamma(\delta_i(t), B(t)) \end{cases} \tag{5}$$

$$\alpha(\delta_i(t), B(t)) = \begin{bmatrix} \Delta_i(t) \\ B\ (t) \\ \delta_i(t+1) \\ x_j(t) \end{bmatrix} \tag{6}$$

$$\beta(\delta_i(t), B(t)) = \begin{cases} c(\delta_{i3}(t))\Delta_i \ \mathit{if} \ \delta_{i3}(t) \in \{0, 2, 4, \ \ldots.\} \\ -c(\delta_{i4}(t))\delta_{i1}(t) - a(\delta_{i4}(t))\dfrac{B(t) - \delta_{i2}(t)}{c(\delta_{i4}(t))} * \delta_{i1}^{-1}(t) \ \mathit{if} \ \delta_{i4}(t) \in \{1, 3, 5, \ \ldots.\} \end{cases} \tag{7}$$

$$\gamma(\delta_i(t), B(t)) = k \sum_{j \in N_i} a_{ij} |x_i \delta_{i4}(t) - x_j \delta_{i4}(t)| \text{when } \delta_{i4}(t) \in \{1, 3, 5\} \tag{8}$$

Event-triggered function

The control input of u_{D2} of each agent is determine by the event-triggered process as shown in **Figure 5**. The purpose of having an event-triggered in the local controller is to reduce communication among agents where the communication among agent i and its neighbour $j \in N_i$ will only happen if the event detected and satisfy the ETF as shown Eq. (9) Eq. (10) indicates the state measurement error between the agent position during the instant event t^i_{k+1} and the previous event t^i_k while Eq. (11) is a threshold value or state dependent value which depends on the agent's state position and neighbour's state position.

Initially, the state of $x_i(t)$ is sample by a sampler at every odd time and become $x_i(nt)$. The event detector will monitor the event based on the given sample. The new update sample state of agent i, $x_i(n_{k+1}t)$ will be sent to the neighbour j if the event detector detects an event. Right after the neighbours j received the state sent by agent i, it will update the agent i state information and store the newly received state values of agent $i, x_i(nt)$. This state will then be used by the controller and event detector of agent j until the next event is triggered from agent i. When there is no event, the sample of agent position $x_i(n_k t)$ will directly be sent to the controller by means of no transmission and a control update will be required at this time. The control signal is held constant until the next event is controlled by having a zero-order holed in the system.

$$f(e_i(t), x_i(t)) \geq \sigma_i(z_i(t)) \tag{9}$$

$$e_i(t) = x_i\left(t^i_k\right) - x_i(t) \tag{10}$$

$$z_i(t) = \sum_{j \in N_i} a_{ij}\left(x_i\ (t) - x_j\ (t)\right) \tag{11}$$

Agent motion, A_i

The state position of agent $A_i(t)$ at $(t+1)$ is based on the latest control input $u_i(t)$obtained from the local controller L_i as Eq. (12)

$$A_i : x_i(t+1) = x_i(t) + u_i(t) \tag{12}$$

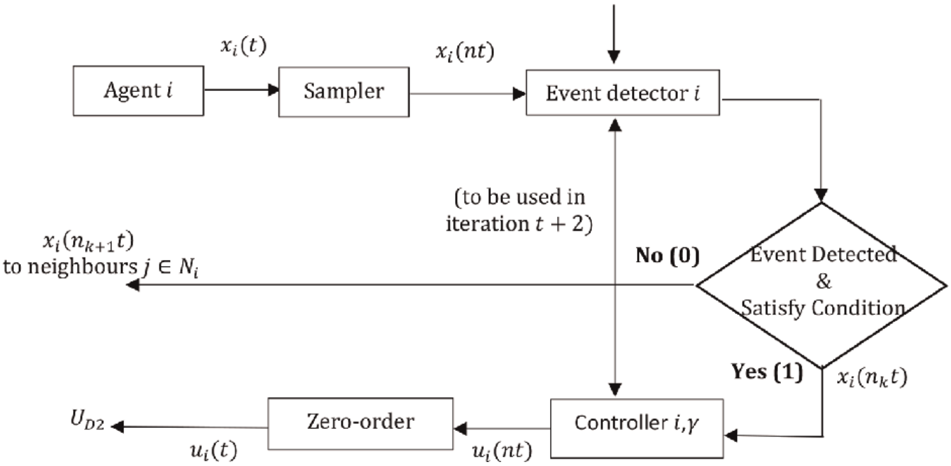


Figure 5.
Agent i event-triggered controller, γ.

3.1 Proposed theorems

Two relevant theorems were proposed which are convergence of the average consensus of distributed controller as Theorem 1 and convergence of BET consensus controller as Theorem 2. The theorems provided with proof derived from theoretical studies as well as proven in the real experiment.

Theorem 1: Control input from communication between the agent and neighbours will affect the agent linear discrete system dynamics. When the system violates the ETF function, the measurement error will be sent by the agent to the neighbours to update the control input whereas if the system satisfies the ETF function, the information will not be sent. The average consensus among the agents is marginally stabilize at $t \to \infty$and reach a steady state error as proven in Gershgorin circle when the Perron matrix eigenvalue is equal to λ and Lyapunov stability that indicates the definite negative when $V < 0$.

Proof 1: Eigenvalues of Perron matrix and Gershgorin circle unit

The dynamic of the system is represented in Eq. (13) by substituting the state measurement error,

$$\begin{aligned} x_i(t+1) &= x_i(t) + u_i(t) \\ &= \left(\widehat{x_i}(t) - e_i(t)\right) + u_i(t) \\ &= \left(\widehat{x_i}(t) - e_i(t)\right) + \left(\sigma\, a_{ij}\left(\widehat{x_j}(t) - \widehat{x_i}(t)\right)\right. \\ &= -e_i(t) + \sigma\, a_{ij}\widehat{x_j}(t) + \left(1 - \sigma\, a_{ij}\right)\widehat{x_i}(t)) \end{aligned} \tag{13}$$

The Perron matrix can be used to represent the agent's next position in discrete time by using a normalized Laplacian matrix as shown in Eq. (14),

$$\begin{aligned} x_i(t+1) &= x_i(t) + u_i(t) \\ &= x_i(t) + \frac{1}{1+d_i} a_{ij} \sum_{j \in N_i} \left(x_j(t) - x_i(t)\right) \end{aligned}$$

since $a_{ij} \sum\limits_{j \in N_i} \left(x_j(t) - x_i(t)\right)$ is equivalent to –$Lx(t)$, thus

$$\begin{aligned} &= x_i(t) + \frac{1}{1+d_i}(-L\,x(t)) \\ &= x_i(t) + \left(-(I+D)^{-1}\right)L\,x(t)) \\ &= x_i(t) - (I+D)^{-1})L\,x(t)) \\ &= I - (I+D)^{-1}(I+A)x(t) \\ &= (I+D)^{-1}(I+A)x(t) \\ &= P_e x(t) \end{aligned} \tag{14}$$

By substituting agent measurement error into Eq. (14), the latest agent state $\mathrm{x_i}(\mathrm{t}_\mathrm{k}^\mathrm{i}) = \widehat{x_i(t)}$ is equivalent to

$$x(t+1) = P_e(e_i(\mathrm{t}) + x_i(t)) \tag{15}$$

The new state position of Eq. (15) will be sent to the connected neighbors for the control update when the ETF is violated, event occurs and the error $e_i(t)$ will simultaneously counted and updated. When ETF is satisfied, there is no event and communication happening between the agent and neighbors which might cause the value of error equal to 0. At this point, the agent state value will remain the same when the sampled data is not updated.

Even though the agent's next state value $x(t+1)$ as shown in Eq. (15) above affected from the state measurement error, the average consensus among connected agents was guaranteed to be achieved when the eigenvalues of square matrix P is strictly contained in the Gershgorin circle criterion. The values of eigen in Gershgorin circle unit was expected to be $\lambda_{10} = 1$ and the eigenvalues of Perron matrix for a topology of 10 agents strongly connected with an undirected graph are $\lambda_1 = 0, \lambda_2 = 0.5 \pm 0.5i$and $\lambda_{10} = 1$.

Proof 2: Lyapunov stability

Eq. (16) of Lyapunov when $Lx \triangleq y = [y_1, y_2, \dots y_N]^T$, represents as Eq. (17)

$$Vx(t) = \frac{1}{2}x(t)^T Lx(t) \tag{16}$$

$$\begin{aligned} \dot{V} = \frac{\partial V}{\partial x} \cdot \frac{dx}{dt} &= x^T L\dot{x} - x^T L(Lx + Le) \\ &= -LLxx^T - LLex^T \\ &= -y^T y - y^T Le \end{aligned} \tag{17}$$

Eq. (17) will be Eq. (18) with Laplacian matrix,

$$\begin{aligned} \dot{V} &= \sum_i y_i^2 - \sum_i \sum_{j \in N_i} y_i (e_i - e_j) \\ &= \sum_i y_i^2 - \sum_i |N_i| + \sum_i \sum_{j \in N_i} y_i e_j \end{aligned} \tag{18}$$

For $a > 0$, $\dot{V}$ can bound by using inequality$|xy| \le \frac{a}{2}x^2 + \frac{1}{2a}y^2$, thus

$$\dot{V} \le -\sum_i y_i^2 + \sum_i a|N_i|y_i^2 + \sum_i \frac{1}{2a}|N_i|y_i^2 + \sum_i \sum_{j \in N_i} \frac{1}{2a} y_j^2 \tag{19}$$

Since the communication graph is undirected,

$$\sum_i \sum_{j \in N_i} \frac{1}{2a} e_j^2 = \sum_i \sum_{j \in N_i} \frac{1}{2a} e_i^2 = \sum_i \frac{1}{2a} |N_i| \, e_i^2 \tag{20}$$

Therefore, the final Lyapunov derivation will be negative definite for $0 < \sigma_i < 1$ which shows that the system was stable.

$$\begin{aligned} \dot{V} &\le -\sum_i (1 - a|N_i|) y_i^2 + \sum_i \frac{1}{a} |N_i| e_i^2 \\ \dot{V} &\le \sum_i (\sigma_i - 1)(1 - a\,|N_i|) y_i^2 \end{aligned} \tag{21}$$

Theorem 2: Given the objective function of BET consensus controller as stated in Eq. (1), the vector of $x(\mathrm{t}) \in R^{nN}$ which indicates the movement of agent will keep changing until satisfying $\nabla P(x(t)) = 0$. Let L_i, $G$and ETF be given by Eqs. (1)–(11). If BET fulfil Theorem 1 and conditions below,

(B1) $\Delta_{i1}(t), \Delta_{i2}(t), \ldots \ldots .\Delta_{iN}(t)$ are Bernoulli random probability distribution.

(B2) $a(t) = a(t+1)$ and $c(t) = c(t+1)$ for every
$t \in \{0, 2, 4, \ldots), \sum_{t=0}^{\infty} a(t) = \infty$, and $\sum_{t=0}^{\infty} c(t) = \infty$

(B3) $x(t)$is stable under gradient system approximation of $x(t) = -\dot{\nabla} P(x\ (t))$ where $x(t) \in R^{nN}$ and the stability is in the Lyapunov sense.

(B4) $E[P(x(t) + c(t)\Delta(t)]$is bounded for all $t \in N$

(B5) $\sup_{t \in N} ||x(t)|| < \infty w.p.1$

(B6) $P(x)$, the performance index value change at t time

(B7) $0 < k < 1/N$, then $\lim_{t \to \infty} x_i(t) - x_j(t) = 0$

(B8) $f(e_i(t), x_i(t)) \leq \sigma_{\mathrm{i}}(z_i(t))$, then $e_i(t) = 0$, else $e_i(t) = \widehat{x_{i(t)}}\text{-}x_{i(t)}$

then,

$\lim_{t \to \infty} x(t) = x_d$ with probability 1 is achieved.

Proof 3: Consensus among agent with BET

1. The agent's movement is subject to the global (Eq. (1)), local (Eq. (5), and event-triggered (Eq. (9)) functions from $(t = 0)$ until $(t \to \infty)$ and achieved convergence with probability 1 under the above B1–B8 conditions.

2. The next agent state position at even time $t \in \{0, 2, 4, \ldots ..\}$ and odd time $\{1, 3, 5, \ldots ..\}$ will depend on Eqs. (21) and (22).

$$B(t) = P(x(t)) \tag{22}$$

$$B(t+1) = P(x(t)) + (c(t) + \Delta_i(t)) \tag{23}$$

3. For $t \in \{0, 2, 4, \ldots ..\}$, the agent movement with BET is represented by Eq. (24).

$$x(t+2) = x(t) - \left[a(t)\frac{B(t+1) - B(t)}{c(t)} * \Delta_i^{-1}(t)\right] - k\sum_{j \in N_i} a_{ij}|x_i - x_j| \tag{24}$$

The Eq. (25) had been simplified to Eq. (26) after substituting Eqs. (22) and (23) into Eq. (25)

$$\boldsymbol{x}(t+2) = x(t) - a(t)\frac{(P(x(t) + c(t)\Delta_i(t)) - P(x(t))}{c(t)}\Delta_i^{-1}(t) - k\, a_{ij}\sum_{j \in N_i}(x_i - x_j) \tag{25}$$

$$x(t+2) = x(t) - a(t)\, d(x(t), \Delta_i(t), c(t)) - k\, a_{ij}\sum_{j \in N_i}(x_i - x_j) \tag{26}$$

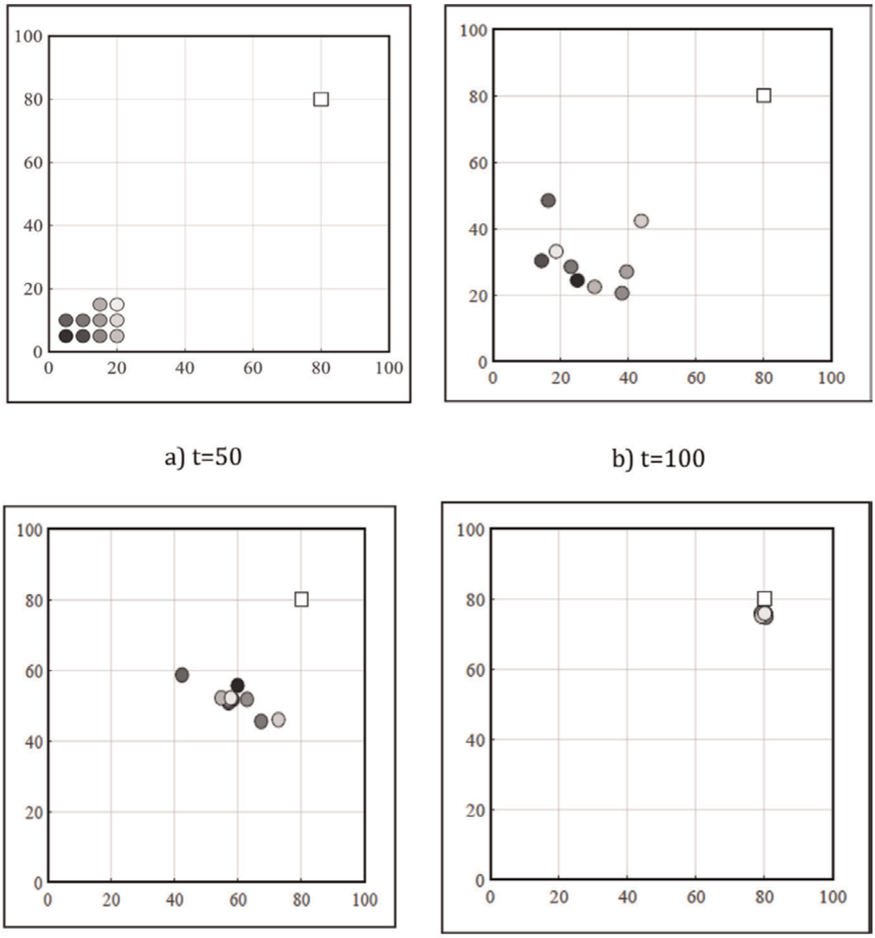
4. The collective of agent dynamic movement will reach convergence when all agents meet rendezvous point of $x_d$after executing Theorem 1 and Theorem 2.

5. If the condition of broadcast with SPSA convergence satisfy (B1–B6) and conditions of ET satisfy (B7 and B8), it shows that all agents reached the rendezvous desired target x_d. The broadcast settings of B1 and B2 as well as B7 and B8 are imposed tuning for $L_i$and G as defined in Eqs. (1)–(11).

4. Agent performances

4.1 Broadcast event-triggered controller

The 10 homogeneous agents were located at 10 initial positions within the workspace area with 1 target point in the 2D Cartesian coordinates as shown in **Figure 6a**. The BET controller was designed completely with the SPSA algorithm with local and global controllers. Optimal settings of the SPSA based on investigation conducted [120, 121] was used as a reference and the final value applied for this case study which are $a = 0.2, A_1 = 30, \alpha_1 = 0.6, c = 1, \gamma_1 = 0.06, a_k = a/(t+1+A)^{\alpha_1}$ and $c_k = c/(t+1)^{\gamma_1}$. With the local controller, the agent will determine the next position based on the stochastic and deterministic rule during even and odd times. Since the agent will communicate among them, the error in communication will also affect the agent's next position. The global controller at the same time will broadcast the signal continuously to the agent to update on the agent's location with the rendezvous point.



a) t=50

b) t=100

c) t=400

d) t=900

Figure 6.
Agent motion and location when t = 50, 100, 400, and 900. a. t = 50. b. t = 100. c. t = 400. d. t = 900.

Time (sec)	20.4	40.8	61.2	81.6	102	122.4	142.8	163.2	183.6	204	225.2
Iteration	100	200	300	400	500	600	700	800	900	1000	1104
NOC	248	181	177	255	436	500	500	500	500	500	520

Table 1.
NOCs per 100 iterations.

From this, it was proven that the controller works effectively by creating a motion of agents to reach consensus among them until they reach the rendezvous point in average of 84.24 s and 676.6 iteration in 10 times run. The movement was recorded in a one-time run as shown in **Figure 6**. From **Table 1**, the NOC was proven to be reduced by 21.79% with a total of 5520 channels used based on the below calculation,

$$\text{Total of iteration} = 1104/2 = 552 \text{ iterations.}$$

$$\text{Time per iteration} = 225.42/1104 = 0.204 \text{ s.}$$

Each iteration is equivalent to 10 channels which carried a total of 552*10 = 5520 channels.

In order to prove the first theorem, which consensus achieved among the agents, the results shown in **Figure 7** were proven. The agents were proven to reach the average local consensus successfully when it met at the average point at t=6 sec (Theorem 1). The eigenvalues of square matrix of Perron illustrated in **Figure 8** showed that the value was in the range of the Gershgorin circle (Proof 1) which indicated that the system was marginally stable, and it reached steady state values (Proof 2). Thus, it can be clearly understood that the average consensus among agents was achieved via communication among agents as shown in Eqs. (27) and (28).

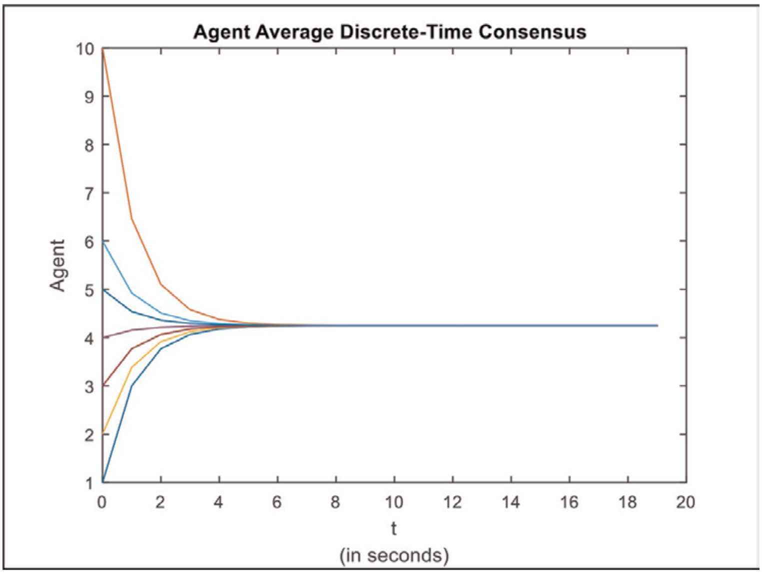


Figure 7.
Agent state trajectories in reaching average consensus.

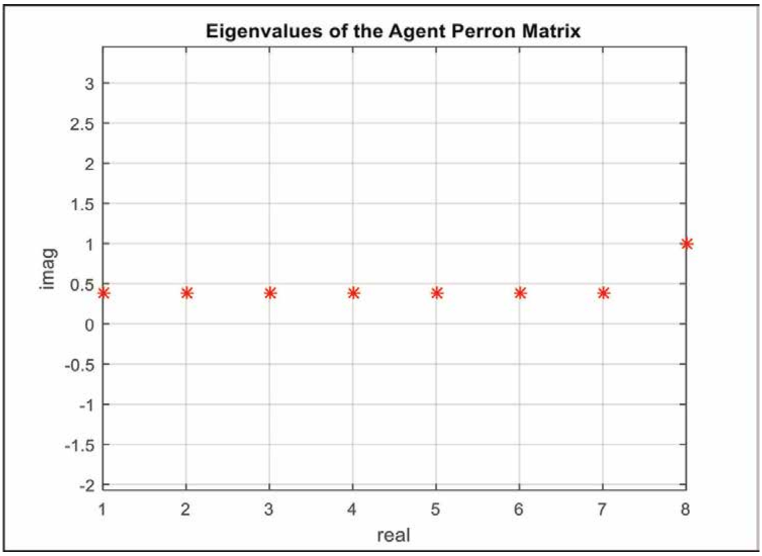


Figure 8.
Eigenvalues of Perron matrix for undirected graph.

$$x_i(t) = \sum_{j\in N_i} a_{ij}\left(x_j(t) - x_i(t)\right) \tag{27}$$

$$\alpha_c = \frac{1}{n}\sum_i x_i(0) \tag{28}$$

4.2 Broadcast event-triggered and broadcast time-triggered performances comparison

The BET and BTT controllers were proven effective to find consensus to rendezvous target point. The performances of convergence time and iteration, trajectory, the NOC utilization were among the evaluated to compare the effectiveness of both the controllers. Compared with the two different sampling methods that has been integrated into the broadcast communication controller, BET showed faster convergence and less number of iterations than BTT. BET was proven to be able to simplify the convergence process to meet rendezvous, leading 57.15 s with 177 number of iterations against BTT based on average value found in 10 times test run. These readings depended on controller performances with a different sampling system, TT and ET produced inconsistent results in ten times run as illustated in **Figures 9** and **10**. The stochastic error and deterministic error will determine the next agent's random movement and next agent's deterministic movement until rendezvous was achieved.

The agent trajectory was observed at every 50 interval of iterations to see the pattern of movement and the measurement error. For BET, it was clearly seen that agent movement was scattered at the first few iterations and gathered towards 60% of the overall process while the BTT move closer and systematically at the beginning until the end of the process as shown in **Figures 11** and **12**. This showed that the BTT communication gave effect to the agent consensus since the agent continuously communicate with the neighbour at every time whereas the BET communication only gave

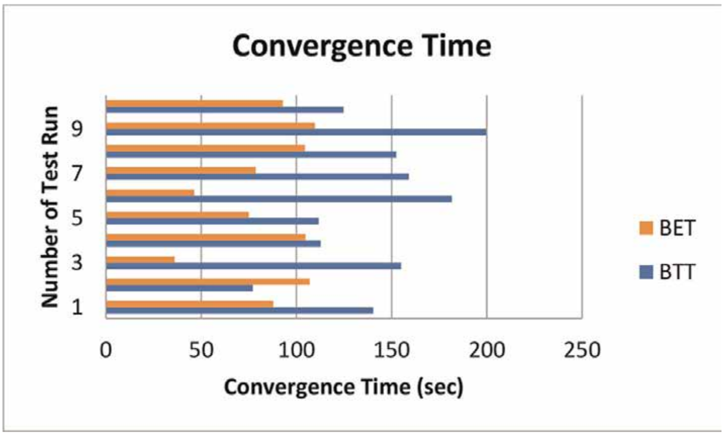


Figure 9.
Convergence time.

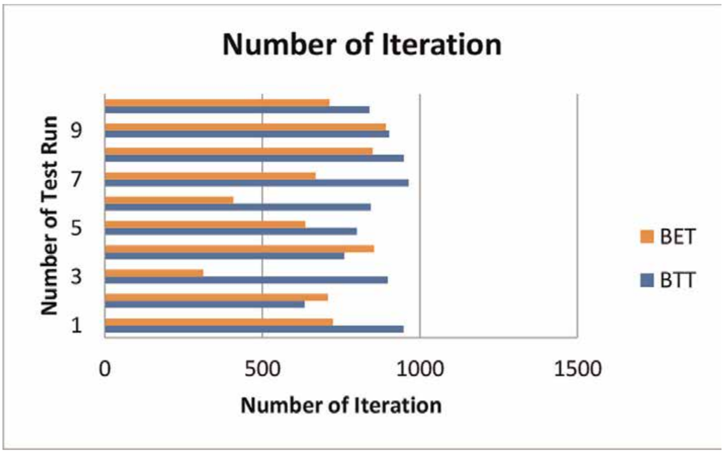


Figure 10.
Number of iteration.

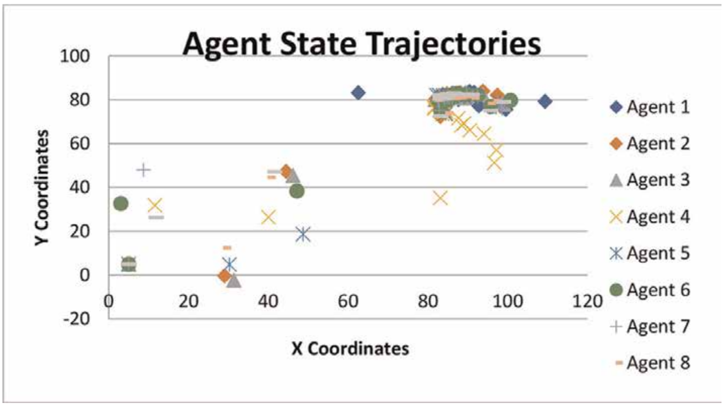


Figure 11.
Agent state trajectories (BET).

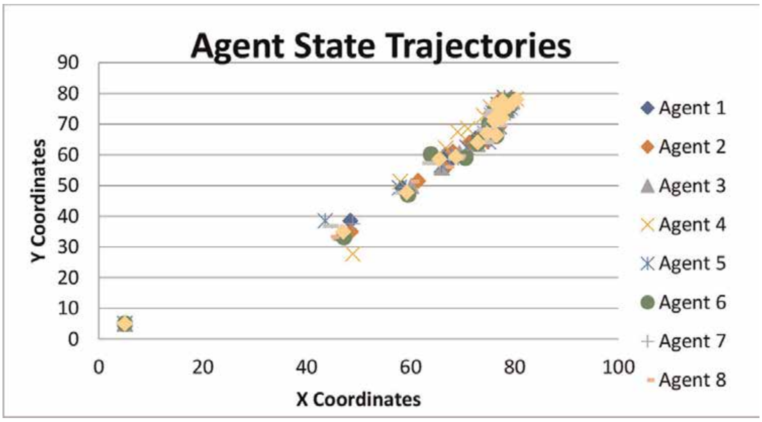


Figure 12.
Agent state trajectories (BTT).

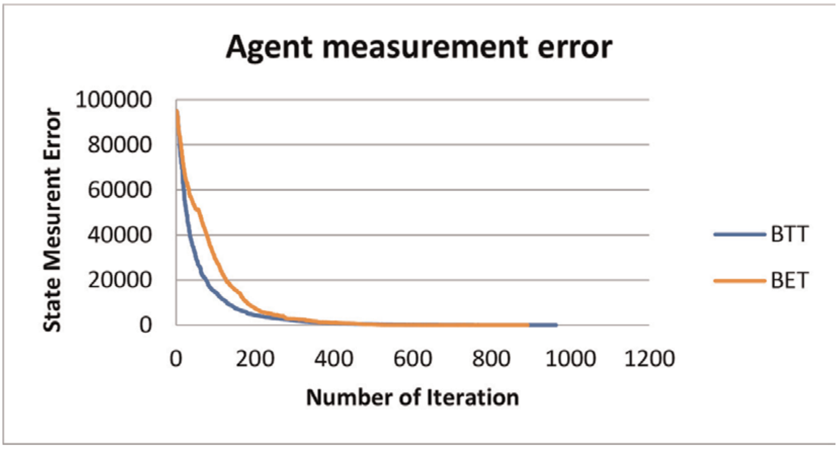


Figure 13.
Agent measurement error (BTT and BET).

significant effects when the ETF condition is violated which results to the pattern of measurement error being lesser with BTT as compared to BET as shown in **Figure 13**. Thus, in terms of movement, the BTT was more accurate compared to the BET.

The NOCs witin 100 iterations was recorded. For BET, the NOCs can be reduced and save up to 29% utilization as compared to BTT as shown in **Table 2**. This was because the BET controller will only allow the information exchange between the agent and neighbours to occur when it violated the ETF but for BTT, the information exchange happened at every time-triggered.

Iteration	100	200	300	400	500	600	700	800
NOC (BET)	248	181	177	255	436	500	500	500
NOC (BTT)	500	500	500	500	500	500	500	500

Table 2.
NOCs of BET and BTT.

Type of sampling	Time (sec)	Number of iteration	No channel	Trajectory	Efficiency
ET	84.24	677	70%	Scattered	High
TT	141.4	854	100%	Systematic	Low

Table 3.
Agent performances with BET and BTT in 10 times run.

In overall comparison, the efficiency of BET was more than BTT sine the time taken and number of iterations were lesser. In terms of trajectory, there were better accuracy for TT since it was closer to the target trajectory as compared to ET which was less systematic from the effects of the communication error. Therefore, the implementation of ET into the agent broadcast has an effect towards the overall agent performances in terms of agent time and iteration, utilization of channel as well as agent trajectory as summarized in **Table 3**. The significant affect was thus proven. It was shown that the second theorem was proven not only in theory but also through the practical experiements.

5. Broadcast event-triggered parameter effects

5.1 Event-triggered function threshold value, σ_i

The threshold value (Eq. (9)) has been varied to observe the effects on agent performance in terms of NOC utilization. The optimal threshold value for this case study was within the range of 10 and 30. The results show that the convergence and NOC gets affected when the value of the threshold is less or more than the optimal range, as shown in **Table 4**.

5.2 Simultaneous perturbation stochastic algorithm parameters

Based on the BET's ideal parameters, which were covered in Section 4.1, the gain a_k ranged from 0.025 to 0.003 and the c_k ranged from 0.662 to 1 for iterations 1–194. The gain, which also has an impact on agent performance in terms of convergence time, iterations, and trajectory movements of the agent to reach rendezvous point, was shown to be impacted by changes in each broadcast parameter coefficient. **Table 5** shows the example of SPSA parameter settings applied for a different cases, which are basic investigation of SPSA by James Spall [104], as well as broadcast and broadcast event-triggering for MARS motion coordination applied in this research project.

Threshold, σ_i of ET	Rendezvous status	Percentage NOC usage
$\sigma_i < 10$	Meet rendezvous with full communication	96.07%
$10 \leq \sigma_i \leq 30$	Meet rendezvous with less communication	93.73%
$\sigma_i > 30$	Does not meet rendezvous	86.93%

Table 4.
Communication and rendezvous status with three different σ_i values.

SPSA variables	By James Spall	Broadcast	Broadcast + ET
a	0.027	0.21	0.2
A_1	20	30	30
α_1	0.602	0.6	0.6
c	1	1	1
γ_1	0.101	0.06	0.06
σ	—	—	30

Table 5.
Optimal settings of SPSA for three different case studies.

6. Conclusions

The BET consensus controller was proven to work for multi-agent robots to reach rendezvous points in a broadcast communication environment. The agents will only communicate among themselves when they satisfy the event-triggered condition to produce an effective and efficient way of motion coordination to reach the desired rendezvous point. The proposed controller system reached convergence with minimum time and iteration as well as reduced channel utilization, which can claim better performances as compared with a conventional BTT system. However, the trajectory pattern was not as systematic as the BTT system. A thorough analysis was conducted to highlight the effect of parameters on agent performances. In the future, the artificial intelligent controller and advanced controller can be embedded with event-triggered systems to produce a more robust and optimal controller for a practical multi-agent motion coordination application. Besides that, it is recommended to apply it to large numbers of agents, such as swarm robots, and to different types of graph topologies for various MARS applications. This controller can be further explored and expanded for many more multi-agent motion coordination systems, such as static or dynamic communication environments.

Acknowledgements

Thanks to UTM for providing their available software and robotic platform. This work was supported by Rabdan Academy, Abu Dhabi, United Arab Emirates.

Conflict of interest

The authors declare no conflict of interest.

Author details

Nohaidda Sariff[1]*, Zool Hilmi Ismail[2], Ahmad Shah Hizam Md Yasir[3],
Denesh Sooriamoorthy[4] and Puteri Nor Aznie Fahsyar Syed Mahadzir[1]

1 School of Engineering, Faculty of Innovation and Technology, Taylor's University, Subang Jaya, Selangor, Malaysia

2 Centre for Artificial Intelligence and Robotics (CAIRO), Malaysia-Japan International Institute of Technology (MJIT), Universiti Teknologi Malaysia, Jalan Sultan Yahya Petra, Kuala Lumpur, Malaysia

3 Faculty of Resilience, Rabdan Academy, Abu Dhabi, United Arab Emirates

4 School of Engineering, Asia Pacific University, Kuala Lumpur, Malaysia

*Address all correspondence to: nohaiddasariff@yahoo.com;
Nohaidda.Sariff@taylors.edu.my

IntechOpen

References

[1] Dorri A, Kanhere SS, Jurdak R. Multi-agent systems: a survey. IEEE Access. 2018;**6**:28573-28593. DOI: 10.1109/ACCESS.2018.2831228

[2] Doriya R, Mishra S, Gupta S. A brief survey and analysis of multi-robot communication and coordination. In: International Conference on Computing, Communication & Automation, India; 15-16 May 2015. pp. 1014-1021. DOI: 10.1109/CCAA.2015.7148524

[3] Yan Z, Jouandeau N, Cherif AA. A survey and analysis of multi robot coordination. International Journal of Advanced Robotics Systems. 2013; **10**(399):1-18

[4] Cao YU, Fukunagu AS, Kahng AB. Cooperative mobile robotics: antecedents and directions. Journal of Autonomous Robots. 1997;**4**:1-23

[5] Sariff N, Xing BTS. A wheeled mobile robot obstacles avoidance for navigation control in a static and dynamic environments. Journal of Physics: Conference Series. 2023;**2523**:1-10. DOI: 10.1088/1742-6596/2523/1/012028

[6] Sariff N, Elyana N. Mobile robot obstacles avoidance by using braitenberg approach. In: 2nd International Conference on Emerging Trends in Scientific Research (ICETSR); November 2014; Kuala Lumpur, Malaysia pp. 1-6

[7] Sariff N, Raihan U. Line following mobile robot by using a fuzzy logic technique. In: 2nd International Conference on Emerging Trends in Scientific Research (ICETSR); Kuala Lumpur, Malaysia. 2014, pp. 1-6

[8] Mertz L. Tiny conveyance: micro- and nanorobots prepare to advance medicine. IEEE Pulse. 2018;**9**(1):19-23. DOI: 10.1109/MPUL.2017.2772118

[9] Baek I, Jeon G, Yu C, Kim S. Hybrid control of magnetic micro-robot using three-axis Helm-holtz coil. In: 2018 IEEE International Magnetics Conference (INTERMAG), Singapore; 23-27 April 2018. pp. 1-10. DOI: 10.1109/INTMAG.2018.8508120

[10] Archila JF, Becker M. Study of robots to pipelines, mathematical models and simulation. In: 2013 Latin American Robotics Symposium and Competition; 21-27 October 2013. pp. 18-23. DOI: 10.1109/LARS.2013.51

[11] Zhong H, Ling Z, Miao C, Guo W, Tang P. A new robot-based system for in-pipe ultrasonic inspection of pressure pipelines. In: 2017 Far East NDT New Technology & Application Forum (FENDT); 22-24 June 2017. pp. 246-250. DOI: 10.1109/FENDT.2017.8584579

[12] Bramante L, Deffacis M, Bussi D, Barrera M, Picco C, Franceschetti P. The mars terrain simulator: A high level measurement facility in support to the ExoMars mission. In: 2019 IEEE 5th International Workshop on Metrology for AeroSpace (MetroAeroSpace); 19-21 June 2019. pp. 303-308. DOI: 10.1109/MetroAeroSpace.2019.8869653

[13] Martinez G. Improving the robustness of a direct visual odometry algorithm for planetary rovers. In: 2018 15th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), Mexico City, Mexico; 5-7 September 2018. pp. 1-6. DOI: 10.1109/ICEEE.2018.8534000

[14] Kenny Chour J-PR, Dotterweich J, Childers M, Humann J, Rathinam S, Darbha S. An agent-based modeling framework for the multi-UAV rendezvous recharging problem. Robotics and Autonomous Systems. 2023;**166**:0921-8890. DOI: 10.1016/j.robot.2023.104442

[15] Sabitri Poudel SM. Priority-aware task assignment and path planning for efficient and load-balanced multi-UAV operation. Vehicular Communications. 2023;**42**:2214-2096. DOI: 10.1016/j.vehcom.2023.100633

[16] Wei Y, Bai Z, Zhu Y. An energy efficient cooperation design for multi-UAVs enabled wireless powered communication networks. In: 2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall), Honolulu, HI, USA; 22-25 September 2019. pp. 1-5. DOI: 10.1109/VTCFall.2019.8890984

[17] Muslimov TZ, Munasypov RA. Consensus-based cooperative circular formation control strategy for multi-UAV system. In: 2019 International Russian Automation Conference (RusAutoCon), Sochi, Russia; 8-14 September 2019. pp. 1-8. DOI: 10.1109/RUSAUTCON.2019.8867733

[18] Ali ZA, Zhangang H, Hang WB. Cooperative path planning of multiple UAVs by using max–min ant colony optimization along with cauchy mutant operator[J]. Fluctuation and Noise Letters. 2020;**20**(1):2150002-2152391. DOI: 10.1142/S0219477521500024

[19] Maria Mannone VS, Chella A. Modeling and designing a robotic swarm: a quantum computing approach. Swarm and Evolutionary Computation. 2023;**79**: 2210-6502. DOI: 10.1016/j.swevo.2023.101297

[20] Wenyu Cai ZL, Zhang M, Wang C. Cooperative artificial intelligence for underwater robotic swarm. Robotics and Autonomous Systems. 2023;**164**: 0921-8890. DOI: 10.1016/j.robot.2023.104410

[21] Ismail ZH, Sariff N. A survey and analysis of cooperative multi-agent robot systems: challenges and directions. In: Mobile Robots. Vol. 1. London, UK: Intech Open Access; 2018, pp. 1-22

[22] Gulzar MM, Rizvi STH, Javed MY, Munir U, Asif H. Multi-agent cooperative control consensus: a comparative review. Electronics. 2018; **7**(2):1-20

[23] Oh K-K, Park M-C, Ahn H-S. A survey of multi-agent formation control. Automatica. 2015;**53**:424-440. DOI: 10.1016/j.automatica.2014.10.022

[24] Balch T, Arkin RC. Behavior-based formation control for multirobot teams. IEEE Transactions on Robotics and Automation. 1998;**14**(6):926-939. DOI: 10.1109/70.736776

[25] Das K, Ghose D. Broadcast control mechanism for positional consensus in multiagent systems. IEEE Transactions on Control Systems Technology. 2015; **23**(5):1807-1826. DOI: 10.1109/TCST.2015.2388732

[26] Gholamreza Khodamipour SK, Farshad M. Adaptive formation control of leader–follower mobile robots using reinforcement learning and the Fourier series expansion. ISA Transactions. 2023; **138**:63-73. DOI: 10.1016/j.isatra.2023.03.009

[27] Wang Y, Cheng Z, Xiao M. UAVs' formation keeping control based on multi–agent system consensus. IEEE Access. 2020;**8**:49000-49012. DOI: 10.1109/ACCESS.2020.2979996

[28] Wang C, Liu C, Liu F. Fixed-time consensus tracking of heterogeneous multi-agent systems. In: 2019 Chinese Automation Congress (CAC); 22-24 November 2019. pp. 984-989. DOI: 10.1109/CAC48633.2019.8997210

[29] Peng X, Geng Z. Distributed rendezvous and consensus control of multiple unicycle-type vehicles under directed graphs. In: 2019 IEEE International Conference on Industrial Technology (ICIT), Melbourne, VIC, Australia; 13-15 February 2019. pp. 1436-1441. DOI: 10.1109/ICIT.2019.8755110

[30] Hanzhen Xiao CLPC, Lai G, Dengxiu Y, Zhang Y. Integrated nonholonomic multi-robot consensus tracking formation using neural-network-optimized distributed model predictive control strategy. Neurocomputing. 2022;**138**:282-293. DOI: 10.1016/j.neucom.2022.11.007

[31] Ma Z, Chen H, Shi L, Shao J. Analysis of containment control for multi-agent systems based on broadcast gossip algorithm. In: 2019 Chinese Automation Congress (CAC); 22-24 November 2019. pp. 5153-5157. DOI: 10.1109/CAC48633.2019.8996305

[32] Haghshenas H, Badamchizadeh MA, Baradarannia M. Containment control of heterogeneneous linear multi agent systems. Automatica. 2015;**54**:210-216

[33] Younan Zhao FZ, Xu D. Self-triggered bipartite formation-containment control for heterogeneous multi-agent systems with disturbances. Neurocomputing. 2023;**548**:0925-2312. DOI: 10.1016/j.neucom.2023.126382

[34] Ilyas M, Ali ME, Rehman N. Design, development & evaluation of a prototype tracked mobile robot for difficult terrain. Sir Syed Research Journal of Engineering & Technology. 2013;**3**(1):7. DOI: 10.1142/S0219477521500024

[35] Mu B, Zhang K, Xiao F, Shi Y. Event-based rendezvous control for a group of robots with asynchronous periodic detection and communication time delays. IEEE Transactions on Cybernetics. 2019;**49**(7):2642-2651. DOI: 10.1109/TCYB.2018.2831684

[36] Zhang Y, Fan Y, Song C, Dong C, Wang L. Multi-agent rendezvous control based on event-triggered mechanism. In: 2017 32nd Youth Academic Annual Conference of Chinese Association of Automation (YAC), Hefei, China; 19-21 May 2017. pp. 780-784. DOI: 10.1109/YAC.2017.7967515

[37] Xavier Défago AH, Tixeuil S, Wada K. Using model checking to formally verify rendezvous algorithms for robots with lights in Euclidean space. Robotics and Autonomous Systems. 2023;**163**:0921-8890. DOI: 10.1016/j.robot.2023.104378

[38] Qin J, Ma Q, Shi Y, Wang L. Recent advances in consensus of multi-agent systems: a brief survey. IEEE Transactions on Industrial Electronics. 2017;**64**(6):4972-4983. DOI: 10.1109/TIE.2016.2636810

[39] Olfati-Saber R, Fax JA, Murray RM. Consensus and cooperation in networked multi-agent systems. Proceedings of the IEEE. 2007;**95**(1):215-233. DOI: 10.1109/JPROC.2006.887293

[40] Toyota R, Namerikawa T. Event-triggered formation control of a generalized multi-agent system. In: 2018 57th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), Nara, Japan; 11-14 September 2018. pp. 940-945. DOI: 10.23919/SICE.2018.8492671

[41] Nambo K, Katsura S. Event-triggered formation control of leader-follower multi-agent system for reducing the number of information transmission. In: IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society, Beijing, China. 2017. pp. 7269-7274. DOI: 10.1109/IECON.2017.8217273

[42] Xing M, Deng F, Hu Z. Sampled-data consensus for multiagent systems with time delays and packet losses. IEEE Transactions on Systems, Man, and Cybernetics: Systems. 2020;**50**(1): 203-210. DOI: 10.1109/TSMC.2018.2815616

[43] Zhang Y, Mu C, Zhao Q, Wang K. Nearly optimal consensus control of discrete time multiagent systems with time delays. In: 2019 IEEE Symposium Series on Computational Intelligence (SSCI); 6-9 December 2019. pp. 72-77. DOI: 10.1109/SSCI44817.2019.9003144

[44] Zhou D, Zhang A, Yang P, Yang M. Finite-time consensus of second-order multi-agent systems with time-delay and connectivity preservation. In: 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), Bari, Italy; 6-9 October 2019. pp. 1782-1787. DOI: 10.1109/SMC.2019.8913935

[45] Zhang Y, Sun J, Liang H, Li H. Event-triggered adaptive tracking control for multiagent systems with unknown disturbances. IEEE Transactions on Cybernetics. 2020; **50**(3):890-901. DOI: 10.1109/TCYB.2018.2869084

[46] Nair RR, Behera L, Kumar S. Event-triggered finite-time integral sliding mode controller for consensus-based formation of multirobot systems with disturbances. IEEE Transactions on Control Systems Technology. 2019; **27**(1):39-47. DOI: 10.1109/TCST.2017.2757448

[47] Teixeira PV, Dimarogonas DV, Johansson KH, Sousa J. Event-based motion coordination of multiple underwater vehicles under disturbances. In: OCEANS'10 IEEE SYDNEY; 24-27 May 2010. pp. 1-6. DOI: 10.1109/OCEANSSYD.2010.5603980

[48] Liu CH, Chen Z, Tang J, Xu J, Piao C. Energy-efficient UAV control for effective and fair communication coverage: A deep reinforcement learning approach. IEEE Journal on Selected Areas in Communications. 2018;**36**(9): 2059-2070. DOI: 10.1109/JSAC.2018.2864373

[49] Xie D, Xu S, Zhang B, Li Y, Chu Y. Consensus for multi agent systems with distributed adaptive control and an event-triggered communication strategy. IET Control Theory & Applications. 2016;**10**:1547-1555

[50] Nowzari C, Cortés J. Zeno-free, distributed event-triggered communication and control for multi-agent average consensus. In: 2014 American Control Conference, Portland, OR, USA; 4-6 June 2014. pp. 2148-2153. DOI: 10.1109/ACC.2014.6859495

[51] Guinaldo M, Farias G, Dormido-Canto S, Chaos D, Sánchez J, Dormido SA. A mobile robots experimental environment with event-based wireless communication. Sensors. 2013;**13**(7):9396-9413

[52] Licea DB, Bonilla M, Ghogho M, Lasaulce S, Varma VS. Communication-aware energy efficient trajectory planning with limited channel knowledge. IEEE Transactions on Robotics. 2020;**36**(2):431-442. DOI: 10.1109/TRO.2019.2948801

[53] Tallamraju R, Verma P, Sripada V, Agrawal S, Karlapalem K. Energy conscious over-actuated multi-agent payload transport robot: Simulations and preliminary physical validation. In: 2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), New Delhi, India; 14-18 October 2019. pp. 1-7. DOI: 10.1109/RO-MAN46459.2019.8956442

[54] Cabreira TM, Kappel K, Brisolara LBD, Ferreira PR. An energy-aware real-time search approach for cooperative patrolling missions with multi-UAVs. In: 2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE); 6-10 November 2018. pp. 254-259. DOI: 10.1109/LARS/SBR/WRE.2018.00054

[55] Oosterhuis B. The impact of event-triggered control on the energy consumption of a legged robot [master of Sciences. Delft Center for Systems and Control (DCSC), Electrical Engineering, Mathematics and Computer Science, Delft University of Technology; 2016

[56] Wu Y, Zhang B, Yang S, Yi X, Yang X. Energy-efficient joint communication-motion planning for relay-assisted wireless robot surveillance. In: IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, Atlanta, GA, USA; 1-4 May 2017. pp. 1-9. DOI: 10.1109/INFOCOM.2017.8057072

[57] Socas R, Dormido S, Dormido R, Fabregas E. Event-based control strategy for mobile robots in wireless environments (in eng). Sensors (Basel). 2015;**15**(12): 30076-30092. DOI: 10.3390/s151229796

[58] Seyboth GS, Dimarogonas DV, Johansson KH. Event-based broadcasting for multi-agent average consensus. Automatica. 2013;**49**(1):245-252. DOI: 10.1016/j.automatica.2012.08.042

[59] Fan Y, Wang S, Qiu J. Event-based control for average consensus of multi-agent systems. In: 2014 International Conference on Mechatronics and Control (ICMC), Jinzhou, China; 3-5 July 2014. pp. 832-836. DOI: 10.1109/ICMC.2014.7231670

[60] Yang Z, Xu W, Shikh-Bahaei M. Energy efficient UAV communication with energy harvesting. IEEE Transactions on Vehicular Technology. 2020;**69**(2):1913-1927. DOI: 10.1109/TVT.2019.2961993

[61] Zebrowski P, Litus Y, Vaughan RT. Energy efficient robot rendezvous. In: Fourth Canadian Conference on Computer and Robot Vision (CRV '07), Montreal, QC, Canada; 28-30 May 2007. pp. 139-148. DOI: 10.1109/CRV.2007.27

[62] Cheng LW, Hii MLHAQ, Murali R, Sooriamoorthy D. Purpose-driven design of a burger assembly machine with a 3 degrees of freedom robot arm. Advanced Robotics and Unmanned Systems. 2022;**1**(2):1-6

[63] Li Y, Yang G, Tong S. Fuzzy adaptive distributed event-triggered consensus control of uncertain nonlinear multiagent systems. IEEE Transactions on Systems, Man, and Cybernetics: Systems. 2019;**49**(9):1777-1786. DOI: 10.1109/TSMC.2018.2812216

[64] Liang H, Guo X, Pan Y, Huang T. Event-triggered fuzzy bipartite tracking control for network systems based on distributed reduced-order observers. IEEE Transactions on Fuzzy Systems. 2020;**29**(6):1601-1614. DOI: 10.1109/TFUZZ.2020.2982618

[65] Adam YM, Sariff N, Al-Geelani NA. E-puck mobile robot obstacles avoidance using fuzzy logic controller. In: 2nd International Conference on Smart Computing and Electronic Enterprise (ICSCEE 2021), Al-Madinah International University; Kuala Lumpur, Malaysia, 2021. pp. 1-6

[66] Sariff N, Nadihah NH. Automatic mobile robot obstacles avoidances in a static environment using hybrid approaches (fuzzy logic and artificial neural network). In: 2014 International Conference Artificial Intelligence System Technology (ICAIST); Kota Kinabalu, Sabah; December 2014. pp. 137-142

[67] Jeffril MA, Sariff N. The integration of fuzzy logic and artificial neural network method for mobile robot obstacles avoidance in a static environment. In: 2013 IEEE 3rd International Conferences on System Engineering and Technology (ICSET); Shah Alam, Malaysia; August 2013. pp. 326-330

[68] Hajar Ashikin S, Akmal Jeffril M, Sariff N. Mobile robot obstacles avoidances by using fuzzy logic techniques. In: 2013 IEEE 3rd International Conferences on System Engineering and Technology (ICSET); Shah Alam, Malaysia. 2013. pp. 332-335

[69] Mohamad MF, Sariff N, Buniyamin N. Mobile Robot Obstacle Avoidance in Various Type of Static Environments Using Fuzzy Logic Approach. In: 2014 International Conference on Electrical, Electronics and System Engineering (ICEESE2014), Kuala Lumpur, Malaysia; December 2014. pp. 83-89. DOI: 10.1109/ICEESE.2014.7154600

[70] Zhao H, Dai X, Zhang Q, Ding J. Robust event-triggered model predictive control for multiple high-speed trains with switching topologies. IEEE Transactions on Vehicular Technology. 2020;**69**(5):4700-4710. DOI: 10.1109/TVT.2020.2974979

[71] Yao D, Li H, Lu R, Shi Y. Distributed sliding-mode tracking control of second-order nonlinear multiagent systems: An event-triggered approach. IEEE Transactions on Cybernetics. 2020;**50**(9):3892-3902. DOI: 10.1109/TCYB.2019.2963087

[72] Chen F, Chen J. Minimum-energy distributed consensus control of multiagent systems: a network approximation approach. IEEE Transactions on Automatic Control. 2020;**65**(3):1144-1159. DOI: 10.1109/TAC.2019.2917279

[73] Shi X, Song S, Wang T, Yan G. Distributed average consensus with event-triggered in multi-agent systems under general directed topology. In: 2017 2nd International Conference on Cybernetics, Robotics and Control (CRC), Chengdu, China; 21-23 July 2017. pp. 116-120. DOI: 10.1109/CRC.2017.25

[74] Wang X, Zeng Z, Cong Y. Multi-agent distributed coordination control: Developments and directions via graph viewpoint. Neurocomputing. 2016;**199**:204-218. DOI: 10.1016/j.neucom.2016.03.021

[75] Wu S, Xia Y, Luo Y, Lin M. Event-triggered cooperative formation control for multi-agent system with dynamic role assignment. In: 2019 Chinese Control Conference (CCC), Guangzhou, China; 27-30 July 2019. pp. 6130-6135. DOI: 10.23919/ChiCC.2019.8865189

[76] Buniyamin N, Sariff N, Ngah WAJW, Mohamad Z. Robot global path planning overview and a variation of ant colony system algorithm. International Journal of Mathematics

and Computers in Simulation (IMACS 2011). 2011;**5**(1):9-16

[77] Sariff N, Buniyamin N. Genetic algorithm versus ant colony optimization algorithm: comparison of performances in robot path planning application. In: 7th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2010); Madeira, Portugal. 2010. pp. 125-132

[78] Sariff N, Buniyamin N. Comparative study of genetic algorithm and ant colony optimization algorithm in global static environment of different complexities. In: 2009 IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA 2009), Daejeon, Korea. 2009. pp. 132-137

[79] Sariff N, Buniyamin N. An overview of autonomous robot path planning algorithms. In: 4th Student Conference on Research and Development (SCORED 2006); Shah Alam, Malaysia. 2006. pp. 184-188

[80] Buniyamin N, Sariff N, Wan Ngah WAJ, Mohamad Z. A simple local path planning algorithm for autonomous mobile robots. International Journal of Systems Applications, Engineering & Development (ISAED 2011). 2011;**5**(2): 151-159

[81] Sariff N, Buniyamin N. Evaluation of robot path planning algorithms in global static environments: Genetic algorithm VS ant colony optimization algorithm. International Journal of Electrical and Electronic Systems Research (IEESR 2010). 2010;**3**:1-12

[82] Li W, Liu Y, Sun H. A survey of event-based consensus for multi-agent systems. In: 2017 Chinese Automation Congress (CAC); 20-22 October 2017. pp. 6606-6611. DOI: 10.1109/CAC.2017.8243967

[83] Peng C, Li F. A survey on recent advances in event-triggered communication and control. Information Sciences. 2018;**457-458**:113-125. DOI: 10.1016/j.ins.2018.04.055

[84] Yu Z, Zhao Y, Zhang W. Study on consensus of the forth-order discrete-time multiagent system in directed networks. IEEE Access. 2020;**8**: 11658-11668. DOI: 10.1109/ACCESS.2020.2965556

[85] Meng X, Xie L, Soh YC, Nowzari C, Pappas GJ. Periodic event-triggered average consensus over directed graphs. In: 2015 54th IEEE Conference on Decision and Control (CDC), Osaka, Japan; 15-18 December 2015. pp. 4151-4156. DOI: 10.1109/CDC.2015.7402866

[86] Xu Z, Huo J, Wang Y, Yuan J, Shan X, Feng Z. Analyzing two connectivities in UAV-ground mobile ad hoc networks. 2011. DOI: 10.1109/CSAE.2011.5952445

[87] Si P, Yu F, Yang R, Zhang Y. Dynamic spectrum management for heterogeneous UAV networks with navigation data assistance. 2015. pp. 1078-1083

[88] Xu B, He W. Event-triggered cluster consensus of leader-following linear multi-agent systems. Journal of Artificial Intelligence and Soft Computing Research. 2018;**8**:293-302. DOI: 10.1515/jaiscr-2018-0019

[89] Zuo R, Li Y, Lv M, Liu Z. Distributed asynchronous consensus control of nonlinear multi-agent systems under directed switching topologies. Automatica. 2023;**152**. DOI: 10.1016/j.automatica.2023.110952

[90] Wentuo Fang ZC, Zamani M. Structural consensus in networks with directed topologies and its cryptographic implementation. ISA Transactions. 2023; **132**:598-606. DOI: 10.1016/j.isatra. 2022.11.003

[91] Xiaofeng Zong TL, Zhang J-F. Consensus control of second-order stochastic delayed multi-agent systems with intrinsic dynamics and undirected topologies. IFAC-PapersOnLine. 2017; **50**(1):2421-2426. DOI: 10.1016/j. ifacol.2017.08.438

[92] Juan Zhang HZ, Cai Y, Lu Y. Distributed cooperative output regulation of heterogeneous linear multi-agent systems based on event- and self-triggered control with undirected topology. ISA Transactions. 2020;**99**: 191-198. DOI: 10.1016/j.isatra.2019. 08.064

[93] Yang J, Lee H, Han S. Synchronization by modified broadcast gossip algorithm in multi-agent system. In: 2018 SICE International Symposium on Control Systems (SICE ISCS); 9-11 March 2018. pp. 6-11. DOI: 10.23919/ SICEISCS.2018.8330149

[94] Das K, Ghose D. Positional consensus in multi-agent systems using a broadcast control mechanism. In: 2009 American Control Conference, St. Louis, MO, USA, 10-12 June 2009. pp. 5731-5736. DOI: 10.1109/ACC.2009.5160384

[95] Nor MHM, Ismail ZH, Ahmad MA. Broadcast control of multi-agent systems with instability phenomenon. In: 2016 IEEE International Conference on Underwater System Technology: Theory and Applications (USYS); 13-14 December 2016. pp. 7-12. DOI: 10.1109/ USYS.2016.7893945.

[96] Azuma SI, Baba I, Sugie T. Broadcast control of markovian multi-agent systems. SICE Journal of Control, Measurement, and System Integration. 2016;**9**(2):103-112. DOI: 10.9746/ jcmsi.9.103

[97] Tanaka Y, Azuma S-I, Sugie T. Broadcast control of multi-agent systems with quantized measurements. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences. 2014;**E97.A**: 830-839. DOI: 10.1587/transfun.E97. A.830

[98] Azuma S, Yoshimura R, Sugie T. Multi-agent consensus under a communication broadcast mixed environment. International Journal of Control. 2014;**87**(6):1103-1116

[99] Yang Y, Yang Y, Fan Y. Hybrid triggering control for average consensus of multi-agent systems. In: 2016 35th Chinese Control Conference (CCC), Chengdu, China; 27-29 July 2016. pp. 7634-7639. DOI: 10.1109/ ChiCC.2016.7554567

[100] Noorbakhsh SM Ghaisari J. Distributed event-triggered average consensus protocol for multi-agent systems. In: 2015 23rd Iranian Conference on Electrical Engineering, Tehran, Iran; 10-14 May 2015. pp. 840-845. DOI: 10.1109/ IraanCEE.2015.7146329

[101] Chen X, Hao F. Event-triggered average consensus control for discrete-time multi-agent systems. IET Control Theory & Applications. 2012;**6**(16): 2493-2498. DOI: 10.1049/iet-cta.2011.0535

[102] Liu Z, Chen Z. Event-triggered average-consensus for multi-agent systems. In: Proceedings of the 29th Chinese Control Conference, Beijing, China; 29-31 July 2010. pp. 4506-4511

[103] Liu D, Yang G. A dynamic event-triggered control approach to leader-following consensus for linear multiagent systems. IEEE Transactions on Systems, Man, and Cybernetics: Systems. 2020;**51**(10):1-9. DOI: 10.1109/TSMC.2019.2960062

[104] Cheng Z, Zhang H, Fan M. Consensus and rendezvous predictive control for multi-agent systems with input constraints. In: Proceedings of the 33rd Chinese Control Conference, Nanjing, China; 28-30 July 2014. pp.61438-1443. DOI: 10.1109/ChiCC.2014.6896840

[105] Park J, Yoo JH, Kim HJ. Two distributed guidance approaches for rendezvous of multiple agents. In: ICCAS 2010, 27-30 October 2010. pp. 2128-2132. DOI: 10.1109/ICCAS.2010.5670200

[106] Li K, Zheng B, Park JH. Event based robust consensus for multi-agent systems via sliding-mode control. In: 2018 Chinese Control and Decision Conference (CCDC) Shenyang, China; 9-11 June 2018. pp. 4483-4488. DOI: 10.1109/CCDC.2018.8407906

[107] Ding L, Han Q, Ge X, Zhang X. An Overview of Recent Advances in Event-Triggered Consensus of Multiagent Systems. IEEE Transactions on Cybernetics. 2018;**48**(4):1110-1123. DOI: 10.1109/TCYB.2017.2771560

[108] Nowzari C, Garcia E, Cortés J. Event-triggered communication and control of networked systems for multi-agent consensus. Automatica. 2019;**105**: 1-27. DOI: 10.1016/j.automatica.2019.03.009

[109] Amini A, Asif A, Mohammadi AA. A performance guaranteed sampled-data event-triggered consensus approach for liner multi-agent systems. Journal of Information Sciences. 2019;**484**:338-349

[110] Zhu W, Tian Z. Event-based consensus of first-order discrete time multi-agent systems. In: 2016 12th World Congress on Intelligent Control and Automation (WCICA); 12-15 June 2016. pp. 1692-1696. DOI: 10.1109/WCICA.2016.7578796

[111] Wang A. Event-based consensus control for single-integrator networks with communication time delays. Neurocomputing. 2016;**173**:1715-1719. DOI: 10.1016/j.neucom.2015.09.044

[112] Heemels WPMH, Johansson KH, Tabuada P. Event-triggered and self-triggered control. In: Baillieul J, Samad T, editors. Encyclopedia of Systems and Control. London: Springer London; 2013. pp. 1-10

[113] Sariff N, Ismail ZH. Broadcast and event triggered distributed consensus controller for multi agent motion coordination systems. In: 2019 12th Asian Control Conference (ASCC), Kitakyushu, Japan; 9-12 June 2019. pp. 260-265

[114] Sariff N, Ismail ZH. Broadcast event-triggered control scheme for multi-agent rendezvous problem in a mixed communication environment. Applied Sciences. 2021;**11**(9):1-21. DOI: 10.3390/app11093785

[115] Nor MH, Ismail Z, Ahmad MA. Broadcast control of multi-agent systems for assembling at unspecified point with collision avoidance. 2016;**8**:75-79

[116] Azuma S, Tanaka Y, Sugie T. Multi-agent consensus under communication-broadcast mixed environment. In: 2012 IEEE 51st IEEE Conference on Decision and Control (CDC), Maui, HI, USA; 10-

13 December 2012. pp. 94-99. DOI: 10.1109/CDC.2012.6426830

[117] Socas R, Dormido S, Dormido R. Optimal Threshold Setting for Event-Based Control Strategies. IEEE Access. 2017;**5**:2880-2893. DOI: 10.1109/ACCESS.2017.2671419

[118] Socas R, Dormido S, Dormido R, Fabregas E. Event-Based Control Strategy for Mobile Robots in Wireless Environment. Journal of Sensors. 2017: 30076-30092

[119] Åström JK, Bernhardsson B. Comparison of periodic and event-based sampling for first-order stochastic systems. IFAC Proceedings Volumes. 1999;**32**(2):5006-5011

[120] Sariff N, Ismail ZH. Simultaneous perturbation stochastic algorithm parameters effect towards multi agent robot broadcast controller. In: 2017 IEEE 7th International Conference on Underwater System Technology: Theory and Applications, Kuala Lumpur, Malaysia. 2017. pp. 1-6. DOI: 10.1109/USYS.2017.8309456

[121] Spall JC. Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control. New Jersey: John Wiley & Sons;, 2003. pp. 1-583

Chapter 5

Path Planning Algorithms for Mobile Robots: A Survey

Zaharuddeen Haruna, Muhammed Bashir Mu’azu, Abubakar Umar and Glory Okpowodu Ufuoma

Abstract

Mobile robots have applications in military (for reconnaissance, search and rescue operations, bomb detection, surveillance), transportation (for cargo and packet delivery), data acquisition, etc. For the mobile robots to be able to execute these tasks with minimum or no human intervention, they need to be autonomous and intelligent. Path planning (PP) is one of the most critical areas of concern in the field of autonomous mobile robots. It is about obtaining a collision-free motion optimal path based on either time, distance, energy or cost in a static or dynamic environment containing obstacles. However, power limitation hinders the mobile robots to accomplish their task of reaching the target location as there are several paths they can follow. Each of these paths has its own path length, cost (i.e., time to reach destination), and energy constraint, thus, the need to plan for an optimal path according to a certain performance criterion. Significant research has been conducted in recent years to address the PP problem. Hence, this chapter is aimed at presenting the different approaches for PP of mobile robots with respect to different optimality criteria (time, distance, energy and cost), challenges and making recommendations on possible areas of future research.

Keywords: mobile robots, autonomous systems, path planning, energy constraint, optimality criterion

1. Introduction

Mobile robots play a crucial role in diverse applications, including search and rescue, cargo and packet delivery, environmental monitoring, and remote sensing, among others [1–3]. These robots operate in various environments and are classified according to their deployment technology into ground mobile robots, aerial mobile robots, water surface mobile robots, and underwater mobile robots. Ground mobile robots utilize mobile platforms like wheels, tracks, legs, or even biomimicry inspired by animals such as snakes. Aerial mobile robots include drones and helicopters, used for tasks like remote sensing, surveillance, and packet delivery [4, 5]. Water surface mobile robots operate on water for data acquisition, environmental monitoring, and surveillance. Underwater mobile robots, on the other hand, undertake missions in the ocean depths, ranging from underwater construction to ocean exploration and military applications.

Achieving autonomy in these challenging environments with minimal or no human intervention requires robust path-planning techniques. Path planning refers to the process by which mobile robots determine the optimal or feasible path from their current location to a desired destination while intelligently avoiding obstacles and adhering to various constraints [6]. In this review paper, we explore the path-planning techniques employed by ground mobile robots in different environments, addressing the unique challenges faced in ground scenarios. By enabling efficient and safe navigation, path planning plays a pivotal role in unlocking the full potential of mobile robots for accomplishing a wide range of tasks in real-world applications.

The robot path-planning problem is a fundamental challenge falling under the broader category of scheduling and routing problems, known for its NP-complete nature [7]. When seeking its target destination, a mobile robot typically has multiple paths to choose from. However, determining the optimal path involves evaluating various criteria such as path length, time, energy consumption, and cost [8]. Among these criteria, path length and time are the most commonly used measures for optimizing robot trajectories.

Mobile robot path planning can be broadly classified into two classes based on the nature of the environment in which the robot operates [8, 9]. These classes are crucial in guiding the path-planning approach and algorithms used to navigate the robot effectively. This is shown in **Figure 1**:

a. Path planning of robot in static environment: this is the class of path planning that contains only static obstacles in the environment;

b. Path planning in dynamic environment: this is the class of path planning that contains both static and moving obstacles in the environment.

The dynamic path planning can be further sub divided into global path planning and local path planning [8–10]. Global path planning, also known as off-line path

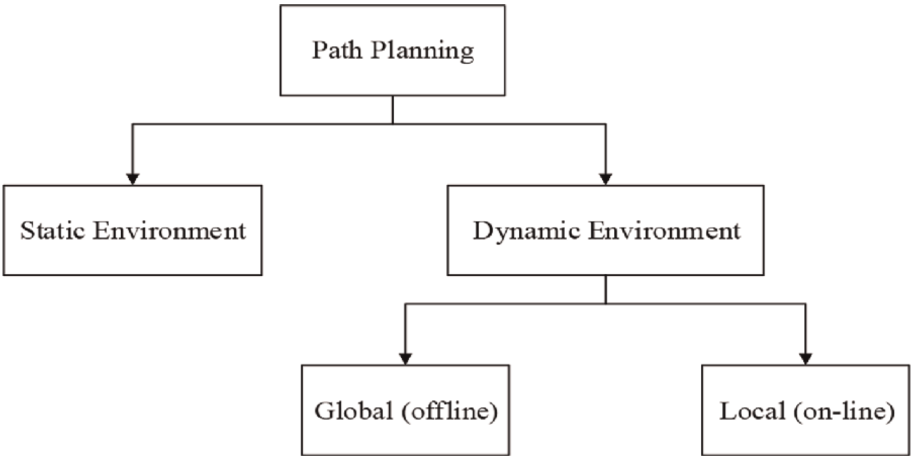


Figure 1.
Classification of mobile robot path planning [10].

planning, refers to the process in which the robot possesses complete knowledge of the environment and the positions of obstacles before it starts moving. During global path planning, the robot calculates the optimal path just once at the beginning and then follows this path until it reaches the target destination. This approach assumes that the environment remains static and does not change during the robot's movement [8, 11].

On the other hand, online or local path planning involves situations where the robot lacks prior information about the environment and the positions of obstacles before it starts moving. In this scenario, the mobile robot relies on its sensors to perceive the environment in real-time. It uses the sensor data to build a map that accurately represents the structure of the environment. Based on this dynamic map, the robot then plans its path on-the-fly, continuously adapting to changing conditions and obstacles as it navigates toward its goal [8, 10, 12].

In the domain of mobile robot path planning, researchers have explored a variety of approaches in the literature to address this challenging problem. These approaches can be broadly classified into three categories: classical (conventional), heuristics, and metaheuristics [12–14].

2. Path-planning approaches

Figure 2 presents the classification of path-planning approaches:

2.1 Classical (conventional) approaches

These methods typically involve using algorithms based on traditional problem-solving techniques. Examples include road map, cell decomposition, potential field, and sub-goal network approaches. These methods are well-suited for simple environments with relatively small search spaces, where they can effectively find optimal or near-optimal paths for mobile robots.

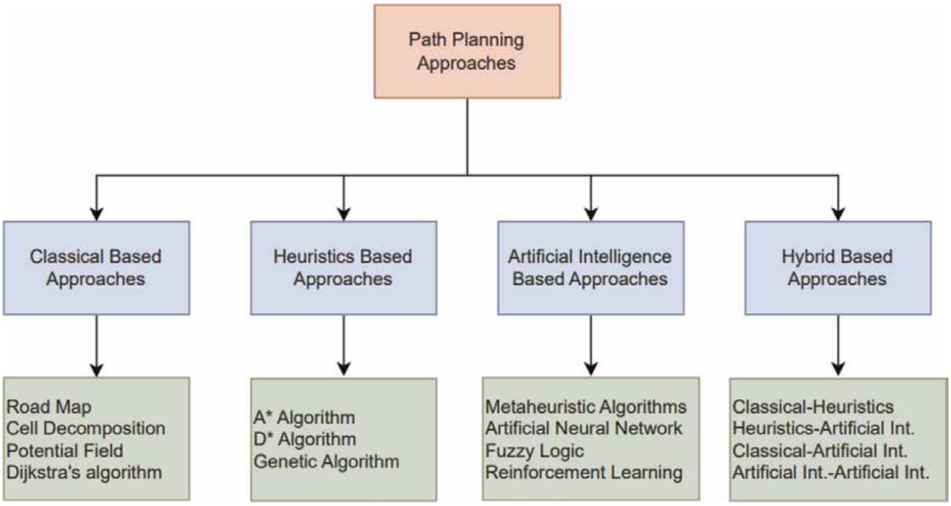


Figure 2.
Path-planning approaches.

However, in more complex scenarios with larger and intricate environments, these approaches may encounter challenges due to their exhaustive nature. Local minima convergence and high computational cost are some of the limitations associated with these methods. Local minima convergence occurs when the algorithms get trapped in suboptimal paths, preventing them from finding the best solution. Additionally, the high computational cost can become a significant hindrance, especially in real-time applications or scenarios with limited computational resources. The inherent unpredictability and uncertainty of real-world applications further compound these limitations. The dynamic and changing nature of real-world environments can make it difficult for traditional methods to adapt and find optimal paths in such uncertain conditions. Also, by providing insights into the most effective path planning algorithms, the paper can contribute to the advancement of mobile robot technologies, leading to improved efficiency, safety, and cost-effectiveness in multiple industries and critical scenarios.

2.1.1 Cell decomposition

In this method, a set of simple cells are obtained by decomposing the free configuration space. The relationships among the adjacent cells are then computed. The cells housing the start and the goal locations of the mobile robot are identified so as to generate a collision-free optimal path by using a sequence of connected cells to connect them [15, 16]. **Figure 3** presents an example of cell decomposition.

2.1.2 Potential field

The concept of this method was first introduced in [17]. A robot in this method is considered as a point in a configuration space that is subjected under the action of an artificial potential function (U) whose variations are seen as the structure of the free space. The potential function is defined as the summation of the attractive and

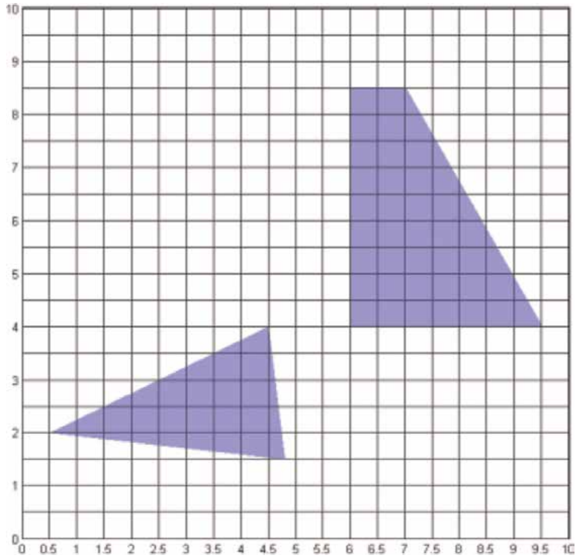


Figure 3.
Cell decomposition [16].

repulsive potentials that respectively pull the robot toward the goal configuration and push the robot away from the obstacles. The results of the potential function lead the robot to the goal destination [18].

$$U = F_a(q) + F_r(q) \tag{1}$$

Where F_a is the attractive force that pulls the robot to the goal destination, while F_r is the repulsive force that pushes the robot from the obstacles [19].

$$F_a(q) = \frac{1}{2} k_a \left(q - q_f\right)^2 \tag{2}$$

Where q is the current state of the robot, k_a is the attractive proportional gain, and q_f is the coordinate of the goal location.

$$F_r(q) = \begin{cases} \frac{1}{2} k_a \left(\frac{1}{p} - \frac{1}{p_0}\right)^2 & if\ p \leq p_0 \\ 0 & if\ p > p_0 \end{cases} \tag{3}$$

Where p is the distance of the robot to the nearest obstacle, p_0 is the distance limit of the repulsion, and k_r is the repulsive proportional gain.

The application of this method and modified versions are found in motion controller for obstacle avoidance of robots [20].

2.1.3 Road map

This method is also referred to as a retraction approach. In this approach, a set of possible motions are retracted and mapped onto a one-dimensional network of lines. This method is a graph searching since the solution is only restricted to the network. The famous roadmaps are discussed below.

Visibility graph: this involves collection of lines that connect a feature of an object to another object in the configuration space. These features in their principal form are vertices of obstacles (polygonal shapes). In this approach, the number of edges is defined by $O(n^2)$. This can be constructed in a 2D configuration space and in $O(n^2)$, where n denotes the number of features. The visibility graph is used for motion planning of mobile robots [21]. **Figure 4** presents the visibility graph:

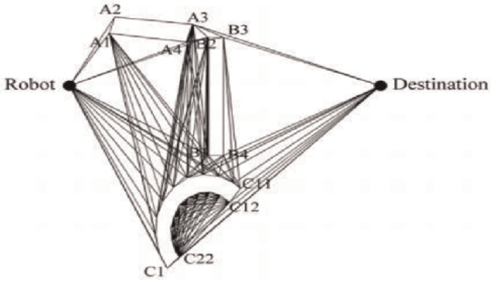


Figure 4.
Visibility graph [15].

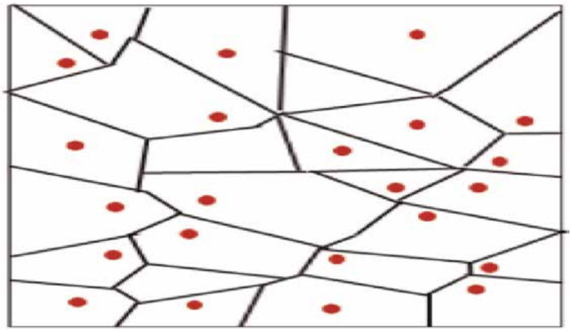


Figure 5.
Voronoi diagram [22].

Voronoi diagram: in this method, as shown in **Figure 5**, cells are obtained by partitioning the configuration space. Each cell consists points that are closer to a particular object in the space than any other. Voronoi diagram has been applied for robot motion planning [23].

Silhouette method: In Refs. [24, 25], this method was developed in arbitrary dimensions to construct a roadmap by projecting an object from a higher to lower dimensional space. The projection boundary curves are then traced out.

2.1.4 Sub-goal network

In this method, a list of reachable formations from start to the goal destination are maintained. The motion planning is only solved when the goal destination is reachable. A local motion planning algorithm (local operator) is used to determine the reachability of one formation from another by moving the robot to the target goal in a straight line [26]. **Figure 6** presents sub-goal graphs on a grid:

2.1.5 Dijkstra's algorithm

Dijkstra's algorithm is a classic and widely used pathfinding algorithm for finding the shortest path between two points in a graph. It is suitable for mobile robot path planning in scenarios where the environment can be represented as a graph, such as grid-based or road networks [28, 29].

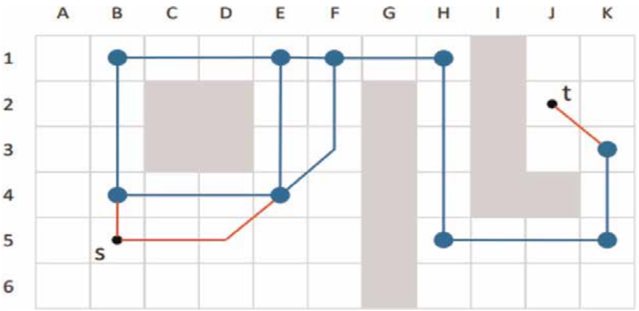


Figure 6.
Sub-goal graphs on grid [27].

However, it has heavy memory because of the computation of all possible outcomes for the determination of the shortest path especially in large and complex environments [29].

2.2 Heuristic approaches

Heuristic techniques leverage domain-specific knowledge or rules to guide the search for a feasible path efficiently. They sacrifice completeness and optimality to gain computational speed, making them suitable for real-time applications [30]. Examples include Dijkstra's algorithm, A* (A-star), D* (D-star), and Genetic algorithm.

2.2.1 A-star algorithm

The A* algorithm is a popular and widely used pathfinding algorithm for finding the shortest path between a starting point and a goal in a graph or grid-based environment. It efficiently combines elements of Dijkstra's algorithm (breadth-first search) and Greedy Best-First Search to achieve optimal pathfinding with reduced computational cost. Its operation is such that it works based on its lowest cost path tree from the start point to the final point [31]. The f-value of a node is defined as the sum of the cost function (g), and the heuristic function (h) for that node is given as:

$$f(n) = g(n) + h(n) \tag{4}$$

Where $g(n)$ is the cost function to track the actual cost of reaching a node from the starting point, and $h(n)$ is the heuristic function that estimates the cost from the current node to the goal node.

A* algorithm has many variants, and they are computationally timesaving based on the accuracy of their heuristic function. Its efficiency and optimality make it a popular choice for mobile robot path planning, especially in grid-based environments and scenarios where finding the shortest path is crucial.

2.2.2 D algorithm*

The D* algorithm is a popular path planning algorithm designed for dynamic environments, where the cost of traversing a particular path can change over time due to the presence of moving obstacles or changes in the environment. It is an improved version of the A* algorithm, specifically tailored to address the challenges of re-planning paths in such dynamic scenarios [32].

$$f(n) = h(n) \tag{5}$$

The D* algorithm's ability to handle dynamic environments and its efficiency in updating paths in real-time make it well-suited for mobile robot path planning in scenarios where the environment is subject to frequent changes or where re-planning is essential to ensure safe and efficient navigation [33].

2.2.3 Genetic algorithm

Genetic algorithm (GA) was introduced in [34] as an evolutionary technique that utilized the advantage operators like natural selection, crossover, and mutation in its search process [35]. Selection operator is utilized in ranking chromosomes on the basis of fitness (i.e., distances of chromosomes from goal) and selecting chromosomes with the minimum path length to generate new solutions. Crossover operator is then utilized to generate a new solution from the selected chromosomes. The crossover operator splits the selected chromosomes and generates new solutions by exchanging their parts. Mutation operator is utilized to randomize the chosen chromosome(s). This operator is typically used whenever the solution converges toward local minima or whenever a certain generation needs performance improvement as a result of poor genetic diversity [36, 37].

2.3 Artificial intelligence approaches

The classical approaches have disadvantages like high computation cost, applicability to only 2D, and trapping in local minima [8, 10, 38]. Also, path planning is a non-deterministic polynomial time hard problem [10]. Thus, metaheuristic search techniques are better techniques for solving this type of problem.

2.3.1 Particle swarm optimization algorithm

Particle swarm optimization algorithm was developed by Eberhart and Kennedy in [35]. The algorithm is a nature-inspired search algorithm that mimics the social behavior of a group of birds or fish schooling [39, 40]. This algorithm belongs to the class of population-based stochastic search technique and is applied by researchers in solving global optimization problems. During the search process of the algorithm, the swarm of particles achieves the objective of obtaining the optimal solution by sharing common information across each member in the swarm. This enables them to globally move together, knowing the position of each at every step of the particles [1, 30, 41].

2.3.2 Ant colony optimization algorithm

Proposed by Marco Dorigo in 1992 as ant system, ant colony optimization is a population-based stochastic search approach used to solve combinatorial optimization problems. It is a nature-inspired algorithm that mimics ants' foraging behavior and their inherent ability of obtaining minimum path length while searching for food sources [6, 42, 43]. Naturally, ants are blind, and they tend to move randomly when searching for food. Ants return to their nest by dropping a chemical pheromone trail that helps them generate the shortest path to their nest from the food source, thereby increasing the probability of that path being followed by other ants [44].

2.3.3 Cuckoo search algorithm

Yang and Deb [45] developed this metaheuristic search algorithm by mimicking the characteristics of brood parasitism of some cuckoo species. Three idealized rules were developed for modeling the algorithm. The first rule deals with the process initialization in which each cuckoo's egg is laid and dumped in a randomly selected nest. The second rule deals with the search process. In this rule, the fittest nest will be passed over to the next generations. The number of host nests is fixed. The final rule deals with the probability of discovering a cuckoo's egg. The probability of discovering a cuckoo's egg should always be between 0 and 1 [45, 46].

2.3.4 Bat algorithm

Yang in 2010 developed this search algorithm as a nature-inspired algorithm. This algorithm mimics the echolocation behavior of micro bats [47]. Micro bats emit ultrasonic while searching for food to detect their preys and the presence of obstacles along their search path. They also use it in dark places to detect cracks in their nests. The ultrasonic sound is usually emitted at the rate of 10 to 200 times per second. The reflected sound after striking the chase or obstacles is called echo. Micro bats navigate the environment by using the time delay to receive the echo to detect an obstacle-free motion path, the distance to the obstacle, and the type of chase. They also use the variations of Doppler effect to differentiate target insects [38, 47, 48].

2.3.5 Artificial neural network

This is a learning algorithm that mimics the human brain's working mechanism. It is designed to emulate the learning process of computational elements called neurons. It consists of input, hidden, and output layers. These layers are interconnected through links known as synaptic weights. Synaptic weights are a set of parameters that are adapted so that the artificial neural network can vary its behavior with respect to the problem to be addressed. The artificial neural network is trained with a supervised learning technique that utilizes back propagation algorithm. The network outputs a model that can be applied for controlling the movement of mobile robots from one location to another while avoiding obstacles in an autonomous manner [49–51].

2.3.6 Fuzzy logic controller

This type of controller has been successfully applied to address various industrial problems. It has three basic components: Fuzzification (input), fuzzy inference (set of linguistic IF – THEN rules), and defuzzification (output). For a path-planning application, the set of linguistic rules that form the fuzzy inference are modeled based on the behavior of the robot. The controller can be used for the navigation of the autonomous robot especially when navigating from a source location to a target destination in an environment with obstacles [52–54].

In summary, the choice of the path-planning approach depends on the specific characteristics of the environment, the computational resources available, and the desired level of optimality. Researchers continue to explore and develop novel techniques to improve the efficiency and effectiveness of mobile robot path planning in various practical scenarios.

3. Performance metrics

Performance metrics of path-planning approaches of mobile robots in solving path-planning problems can be evaluated quantitatively. **Figure** 7 is the representation of a path-planning problem.

Figure 6 is the graphical representation of a path-planning problem where the red square is the start location and the green diamond is the goal location. The obstacles were represented with blue rectangles of different sizes, randomly generated and positioned in the environment. The task of the mobile robot is to reach the goal

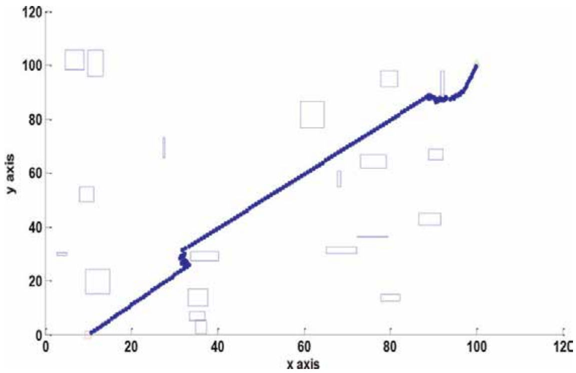


Figure 7.
Path planning problem [12].

location via a collision-free optimal path using one or a combination of the following performance metrics in evaluating the paths:

3.1 Time

This is the time taken by a robot to move from the start to the goal location. For simulation MATLAB, it is recorded by the difference between the value of the MATLAB command tic and toc, which respectively record the start and end time of a running program. The time taken is then displayed in the command line interface.

3.2 Obstacle avoidance efficiency

This deals with the efficiency of the robot in avoiding obstacle(s) in an environment.

3.3 Path length

Path Length (PL) is the total distance covered by the robot from the start location (x_s, y_s) to the goal location $\left(x_g, y_g\right)$. The PL of the trajectory in an x-y plane is computed as:

$$PL = \sum_{i=1}^{n-1} \sqrt{\left(x^{i+1} - x^i\right)^2 + \left(y^{i+1} - y^i\right)^2} \tag{6}$$

3.4 Energy consumption

This is the amount of energy used by the robot in moving from the start location to the goal location. The size of the environment, onboard equipment, weight of the robot, air drag, nature of the road surface, and motion of the actuators are some of the factors that determine the amount of consumed energy.

3.5 Smoothness

This is the measure of time and energy requirement for the motion of a robot. The smoothness of UGV's movement is computed as a function of curvature (k) using bending energy (BE). The curvature of curves at any point, say (x_i, y_i), across a trajectory in an x-y plane is computed as

$$k(x_i, y_i) = \frac{f''(x_i)}{\left(1 + (f'y_i)^2\right)^{\frac{3}{2}}} \tag{7}$$

The BE is computed mathematically as presented as:

$$BE = \frac{1}{n} \sum_{i-1}^{n} k^2(x_i, y_i) \tag{8}$$

The distribution of the performance metrics based on some selected literatures is presented in **Table 1**.

The summary of the performance metrics as reported in **Table 1** is shown in **Table 2**:

Author	What was done	Performance metrics
[1]	Developed an obstacle avoidance scheme-based elite opposition bat algorithm for unmanned ground vehicles	Path length and time
[2]	Developed optimal path-planning technique using elite opposition-based bat algorithm for mobile robots	Elapsed time
[3]	A directed artificial bee colony to solve the problem of path planning by using the present direction of the fittest bee to move other bees toward the optimal position	Shortest distance
[6]	Developed a dynamic path planning technique for autonomous navigation of mobile robot in an unknown static environment	Elapsed time
[7]	Developed an evolutionary approach for mobile robot path planning in a complex environment	Success rate
[8]	A new ant–based path planning approach that considers unmanned ground vehicle energy consumption in its planning strategy	Energy, time, travel length, and computational time
[10]	A path-planning algorithm of mobile robots using bacteria foraging optimization technique	Minimum time
[11]	Developed a graphical user interface for the path planning of mobile robots using a modified bat algorithm in a 2D environment containing rectangular static obstacles	Minimum time
[12]	Developed a dynamic planning algorithm-based modified bat algorithm for mobile robots in an unknown static environment	Elapsed time
[13]	A global path planning using modified firefly algorithm	Shortest distance
[14]	A global real-time optimal path planning of mobile robots in an environment with a moving target based on artificial immune approach	Shortest distance and minimum time
[15]	A smooth local path planning algorithm based on modified visibility graph	Path length, time, planning time
[16]	A safe path planning using cell decomposition approximation	Safety
[18]	Developed an optimal path planning for unmanned ground vehicles using potential field method and optimal control method	Time

Author	What was done	Performance metrics
[21]	Developed a visibility-polygon search and Euclidean shortest paths	Time
[22]	A fast-global flight path-planning algorithm based on space circumscription and sparse visibility graph for unmanned aerial vehicle	Path length and time
[27]	Feasibility study: Sub-goal graphs on state lattices	Time
[28]	Application of the Dijkstra algorithm in robot path planning	Path length
[29]	Global path planning for mobile robots based on artificial bee colony and Dijkstra's algorithms	Path length
[30]	Comparison of BAT with PSO for path-planning problems	Path length and Time
[31]	Mobile robot path-planning algorithm based on improved A star	Path length and smooth path
[32]	Path-planning algorithm using D* heuristic method based on PSO in a dynamic environment	Path length
[36]	An improved genetic algorithm-based path-planning approach for autonomous robots	Number of turns and minimum time
[37]	A path-planning technique using a modified genetic algorithm for a mobile robot	Shortest distance and safety
[55]	A global mobile robot path planning in a dynamic environment with obstacles of different shapes and sizes	Shortest distance and minimum time
[56]	An ant colony optimization path planning of an autonomous robot in a dynamic environment with rectangular obstacles	Minimum time
[57]	A metaheuristic optimization-based algorithm to solve path-planning problems of mobile robots in an unknown environment	Minimum time
[58]	Compared bat algorithm with cuckoo search for mobile robot path planning in an environment with static obstacles	Shortest distance and minimum time
[59]	Applied cuckoo search algorithm for solving path planning of mobile robots in different environments with static obstacles.	Distance and time
[60]	A modified particle swarm optimization for time optimal path planning of unmanned ground vehicle with solar power schedules	Minimum time
[61]	A lane-level path-guiding method for unmanned ground vehicles in a structured road environment	Traffic rules
[62]	An image processing and reinforcement learning-based path planning technique for mobile robots	Shortest distance
[63]	A path-planning technique for mobile robots using a set of surrounding points and path improvement	Shortest and smoothest path
[64]	A novel path-planning technique of mobile robots using fuzzy image processing in a 2D map containing obstacles	computational time
[65]	A modified ant colony optimization algorithm-based path-planning software tool for 2D environments containing obstacles	Minimum time
[66]	Developed a path-planning technique for mobile robots' obstacle avoidance	Shortest distance
[67]	Developed a hybrid planning algorithm by fusing A* algorithm with Voronoi diagram for non-holonomic autonomous vehicles	Time

Author	What was done	Performance metrics
[68]	Developed a hybrid algorithm for path planning of mobile robots in static and dynamic environments.	smoothness, path length and safety
[69]	Developed a hybrid optimization path-planning algorithm for autonomous mobile robots in static and dynamic environments.	Traveling distance
[70]	A dynamic path-planning technique for autonomous driving with the mission of avoiding obstacles on various roads	Shortest distance
[71]	Comparison between two meta-heuristic algorithms for path planning in robotics	Time

Table 1.
Distribution of performance metrics.

S/N	Performance metric	Paper(s)
1	Time	[2, 6, 8, 10–12, 14, 15, 18, 21, 22, 27, 30, 36, 55–60, 64, 65, 67, 71]
2	Distance	[1, 3, 8, 13–15, 22, 28–32, 37, 55, 58, 59, 62, 63, 66, 68–70]
3	Energy	[8]
4	Safety	[16, 37, 68]
5	Smooth path	[31, 63, 68]

Table 2.
Summary of the performance metrics.

Figure 8 presents a pictorial representation of the highest- and lowest-used performance metrics for different path planning approaches in the reviewed literature. The performance metrics depicted in the figure encompass various aspects, such as path length, computation time, energy consumption, and other relevant criteria.

It can be seen from **Figure 8** that distance (path length) and time are the most commonly used performance evaluation criteria for path-planning approaches, while energy consumption is the least-used criterion.

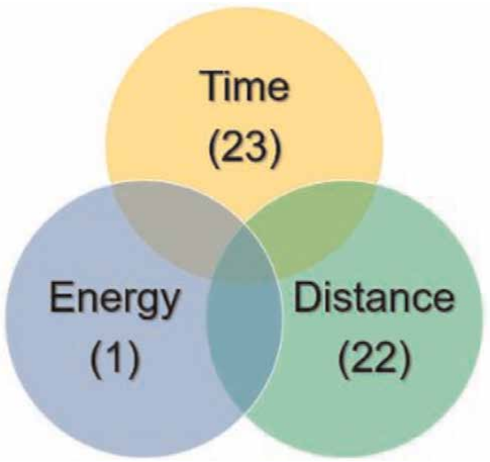


Figure 8.
Performance metrics distribution.

4. Literature review

The review in this section is focused on the use path-planning approaches on mobile robots. Different research has been investigated and reported as follows:

In Ref. [55], a global mobile robot path planning in a dynamic environment with obstacles of different shapes and sizes. The path-planning algorithm was developed using the direction concept, which is capable of handling static obstacles, and the waiting time concept, which is capable of handling moving obstacles.

In Ref. [56], an ant colony optimization path planning of an autonomous robot in a dynamic environment with 20 rectangular obstacles randomly generated. In the experiment, a robot of asterisk structure was assumed, where the source was of a square shape and the target was of a diamond shape in a moving space of 160*160 cm^2.

In Ref. [14], a global real-time optimal path planning of mobile robot in an environment with a moving target based on artificial immune approach. A MAKLINK simulation environment was developed, and Dijkstra is then applied to determine the suboptimal path of the mobile robot.

In Ref. [57], a metaheuristic optimization-based algorithm to solve path planning problem of mobile in an unknown environment consisting of 20 randomly generated obstacles. The path-planning algorithm was based on ant colony optimization, and its performance was compared with that of a particle swarm optimization algorithm.

In Ref. [3], a directed artificial bee colony to solve the problem of path planning by using the present direction of fittest bee to move other bees toward the optimal position.

In Ref. [10], a path-planning algorithm of mobile robots using the bacteria foraging optimization (BFO) technique. The developed technique was used to determine an obstacle-free motion path for a robot moving toward the target in a dynamic environment.

In Ref. [58], the bat algorithm was compared with cuckoo search for mobile robot path planning an environment with 20 static obstacles. Simulation was done using MATLAB; the algorithms were compared by increasing the number of populations.

In Ref. [30] showed the growth of path planning by comparing two algorithms of initial and recent phases, in which one is the particle swarm optimization (PSO) algorithm, while the other is the bat algorithm.

In Ref. [59] applied the cuckoo search algorithm for solving path planning of mobile robots in different environments with static obstacles. Based on the behavior of robots with respect to obstacle avoidance and goal seeking, a new objective function was formulated.

In Ref. [71] compared the performance of two path-planning algorithm-based ABC and PSO in a static environment containing obstacles. The performance of the planning algorithms was evaluated by varying the number of populations, and the time taken by the robot to reach the target is recorded.

In Ref. [60], a modified particle swarm optimization for time optimal path planning. The path planning algorithm was applied to an unmanned ground vehicle with solar power schedules. The electrical component of the unmanned ground vehicle was driven by its power, while the solar energy harvested by the unmanned ground vehicle was strictly used for the mission's energy constraints.

In Ref. [37] presented a path-planning technique using a modified genetic algorithm for a mobile robot. The square map simulation environment was modeled so as to reflect a real environment.

In Ref. [61] presented a lane-level path-guiding method for unmanned ground vehicles (UGV) in a structured road environment. A lane-level digital map was built to present lanes and segments of a road with respect to traffic rules.

In Ref. [62] presented an image processing and reinforcement learning-based path-planning technique for mobile robots. The algorithm and image processing were implemented in MATLAB and transferred to the mobile robot.

In Ref. [63] presented a path-planning technique for mobile robots using a set of surrounding points and path improvements. The developed technique was tested on 10 maps with different starting points, target points, and number of obstacles.

In Ref. [8] proposed a new ant-based path-planning approach that considers unmanned ground vehicle energy consumption in its planning strategy. In the work, the energy prediction model was incorporated into an ant-based algorithm to reach the target goal by providing the collision-free shortest path with low power consumption.

In Ref. [64] presented a novel path planning technique of mobile robot using fuzzy image processing in a 2D map containing obstacles. The fuzzy rules and membership functions were optimized using bacterial. A modified genetic algorithm with enhanced A* algorithm was used to obtain energy consumption, security, minimum traveling time, and safety for the mobile robot.

In Ref. [36] presented an improved genetic algorithm (GA)-based path-planning approach for autonomous robots. The developed path-planning approach was implemented on a 2D grid simulation environment.

In Ref. [65] presented a modified ant colony optimization (ACO) algorithm-based path-planning software tool for 2D environments containing obstacles.

In Ref. [64] presented a dynamic path-planning technique for autonomous driving with the mission of avoiding obstacles on various roads. The developed technique also obtains appropriate vehicle speed and acceleration to enable maneuvering around obstacles.

In Ref. [2] developed an optimal path-planning algorithm using elite opposition-based bat algorithm (EOBA) for mobile robots. The EOBA was developed by diversifying the exploration phase of the bat algorithm so as to increase the diversity of the solutions in the search space. The inertia weight was introduced at the local search phase to balance the exploration and exploitation.

In Ref. [66] developed a path-planning technique for mobile robots' obstacle avoidance. The technique was developed by modifying the artificial potential field algorithm so as to avoid been trapped in local minima. The modified algorithm can guide the robot to the target location through an optimal path without colliding with obstacles.

In Ref. [6] developed a dynamic path-planning technique for autonomous navigation of mobile robots in an unknown static environment. An objective function was modeled in the form of distance function using the coordinate of the start and goal locations. A dynamic path-planning algorithm was then applied to solve the optimization problem, which generates an optimal collision-free motion path for autonomous navigation of the mobile robot.

In Ref. [67] fused A* algorithm with Voronoi diagram for solving path planning of non-holonomic autonomous vehicles. The Voronoi diagram generates guiding waypoints that were optimized by the A* algorithm to generate the shortest path for the autonomous vehicles regardless of the constraints.

In Ref. [68] developed a hybrid algorithm for path planning of mobile robots. Membrane-inspired algorithm, pseudo bacteria genetic algorithm, and artificial potential field were hybridized to form a membrane pseudo bacterial potential field algorithm.

In Ref. [11] developed a graphical user interface for path planning of mobile robots using a modified bat algorithm in a 2D environment containing rectangular static obstacles. The modified bat algorithm was utilized as the navigational guidance controller that guided the mobile robot based on the echolocation behavior of bats. The development of the GUI software makes it easier to implement path planning of mobile robots as the user is provided with the opportunity to define environment variables without having any knowledge of programming.

In Ref. [12] developed a dynamic planning algorithm-based modified bat algorithm for mobile robots. An objective function was modeled based on the distance of the robot to the goal location and to the nearest obstacles. The dynamic planning algorithm optimized the paths of the mobile robots based on this function in others to generate a collision-free optimal path.

Classification	Path planning approach	Reference
Classical	Cell decomposition	[16]
	Visibility graph	[15, 21, 22]
	Dijkstra algorithm	[28, 29]
	Artificial Potential Field	[18, 66]
	Subgoal graphs	[27]
	set of surrounding point and path improvement algorithm	[63]
Heuristics	Genetic algorithm	[7, 36, 37]
	A* algorithm	[31]
	D* algorithm	[32]
Artificial intelligence	Bat algorithm	[1, 2, 6, 11, 12, 30, 58]
	Particle swarm optimization algorithm	[30, 32, 57, 60, 71]
	Ant colony optimization	[8, 56, 57, 65]
	Artificial bee colony	[3, 29, 71]
	Firefly algorithm	[13]
	Bacteria foraging	[10]
	Cuckoo search algorithm	[58, 59]
	Artificial immune algorithm	[14]
	Fuzzy logic controller	[64]
	Reinforcement learning	[62]
Hybrid methods	Hybrid A* algorithm with Voronoi diagram	[67]
	Hybrid pseudo-bacterial potential field algorithm	[68]
	Hybridized particle swarm optimization-modified frequency bat algorithm	[69]

Table 3.
Distribution of path planning approaches.

In Ref. [69] developed hybrid optimization path-planning algorithm for a mobile robot in static and dynamic environments. The hybrid algorithm is a combination of PSO and modified frequency BA. A multi-objective optimization problem was formulated based on the shortest distance and path smoothness, which was optimized by the developed path planning algorithm to generate an optimal collision path.

In Ref. [48] developed an obstacle avoidance scheme-based elite opposition bat algorithm for unmanned ground vehicles. The obstacle avoidance system comprises a simulation map, a perception system for obstacle detection, and implementation of EOBA for generating an optimal collision-free path that led the UGV to the goal location. Three distance thresholds of 0.1 m, 0.2 m, and 0.3 m were used in the obstacle detection stage, while 0.3 m was determined as the optimal distance threshold for obstacle avoidance.

Table 3 presents the summary of the path-planning approaches based on the literatures reviewed.

Figure 9 presents a clear distribution of the path-planning approaches based on the reviewed literature.

It can be seen from **Figure 9** that the artificial intelligence-based path planning approach emerges as the most frequently used technique, guiding mobile robots to their target destinations while effectively avoiding collisions with obstacles.

This prominence of artificial intelligence-based approaches can be attributed to their versatility and robustness in addressing various path-planning challenges. Leveraging learning capabilities and intelligent decision-making, these techniques enable mobile robots to adapt to dynamic environments, learn from past experiences, and efficiently navigate through complex terrains.

Out of the 44 reviewed literatures, 26 (59%) studies employed artificial intelligence-based path planning, showcasing its widespread acceptance and effectiveness in the field.

The dominance of artificial intelligence-based approaches underscores their potential to revolutionize mobile robot navigation, driving advancements in autonomous robotics and expanding the scope of practical applications in industries such as logistics, transportation, search and rescue, surveillance, and service robotics.

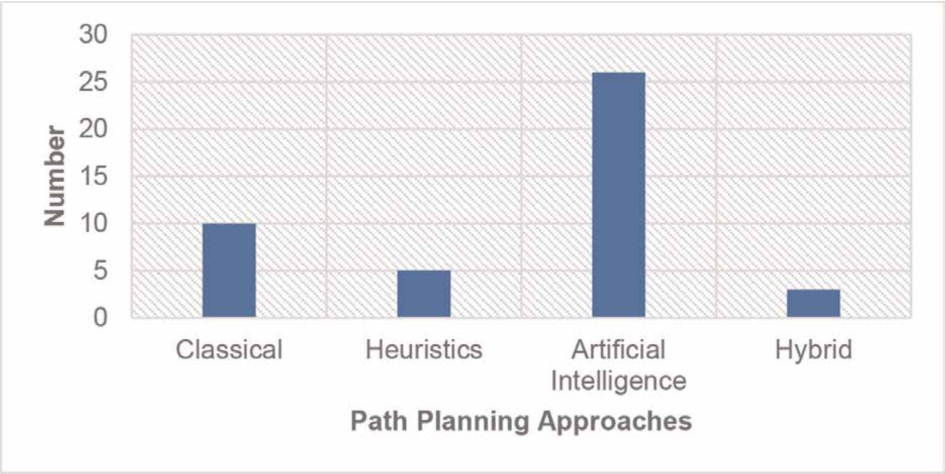


Figure 9.
Distribution of path planning approaches.

5. Conclusion

This review investigated the applications of different path-planning approaches for a mobile robot operating in a known or partially known environment with obstacles and defined start and goal locations. The path-planning problem aims to find an optimal or feasible path that allows the mobile robot to navigate from its starting position to the target destination while avoiding collisions with obstacles. It is evident from the reviewed literatures that considerable research attention has been given to the field of path planning for mobile robots. Researchers have developed various approaches to generate optimal paths, considering criteria such as path length, cost, time, and energy consumption. Among the classes of the path-planning approaches reported in the literature, artificial intelligence-based techniques stand out as the most frequently used method for generating collision-free motion paths, owing to their promising performance and adaptability.

It is also evident that the artificial intelligence-based approaches have utilized to enhance classical and heuristic-based methods, improving the overall efficiency and effectiveness of mobile robot path planning. By leveraging learning capabilities and intelligent decision-making, these techniques have paved the way for more advanced and autonomous robot navigation.

Despite the significant progress made in the field of path planning of mobile robots, it is clear that the current approaches do not give sufficient consideration to cost and energy consumption of mobile robots in path optimization. As mobile robots become more prevalent in various applications, optimizing cost and energy efficiency becomes increasingly critical.

Therefore, there is a clear need for the development of new path-planning approaches that place a strong emphasis on generating optimal paths while minimizing cost and energy consumption. Future research efforts should focus on exploring novel techniques that integrate these performance criteria into the path-planning process.

By addressing these challenges and advancing the field of mobile robot path planning, we can unlock the full potential of mobile robots, enabling them to navigate efficiently and safely in diverse and dynamic environments.

Author details

Zaharuddeen Haruna*, Muhammed Bashir Mu'azu, Abubakar Umar
and Glory Okpowodu Ufuoma
Computer Engineering Department, Ahmadu Bello University, Zaria, Nigeria

*Address all correspondence to: hzaharuddeen@abu.edu.ng

IntechOpen

References

[1] Haruna Z, Mu'azu MB, Sha'aban YA, Adedokun EA. Obstacle avoidance scheme based elite opposition bat algorithm for unmanned ground vehicles. Covenant Journal of Information and Communication Technology. 2021;**9**(1):1-15

[2] Haruna Z, Mu'azu MB, Oyibo P, Tijani SA. Development of an optimal path planning using elite opposition based bat algorithm for a mobile robot. Yanbu Journal of Engineering and Science. 2021;**16**(1):1-9

[3] Abbas NH, Ali FM. Path planning of an autonomous mobile robot using directed artificial bee colony algorithm. International Journal of Computers and Applications. 2014; **96**(11):11-16

[4] Alanezi MA, Haruna Z, Sha'aban YA, Bouchekara HREH, Nahas M, Shahriar MS. Obstacle avoidance-based autonomous navigation of a quadrotor system. Drones. 2022;**6**(10):288

[5] Ali ZA, Zhangang H. Multi-unmanned aerial vehicle swarm formation control using hybrid strategy. Transactions of the Institute of Measurement and Control. 2021;**43**(12): 2689-2701

[6] Haruna Z, Musa U, Mu'azu MB, Umar A. A dynamic path planning technique for autonomous mobile robot in unkwown static environment. In: IEEE 1st International Conference on Mechatronics, Automation & Cyber-Physical Computer Systems (MAC 2019). Owerri, Nigeria: IEEE; 2019. pp. 36-41

[7] Hosseinzadeh A, Izadkhah H. Evolutionary approach for mobile robot path planning in complex environment. International Journal of Computational Science and Engineering. 2010;**7**(4):1

[8] Jabbarpour MR, Zarrabi H, Jung JJ, Kim P. A green ant-based method for path planning of unmanned ground vehicles. IEEE Access. 2017;**5**:1820-1832

[9] Reshamwala A, Vinchurkar DP. Robot path planning using an ant colony optimization approach: A survey. International Journal of Advanced Research in Artificial Intelligence. 2013; **2**(3):65-71

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

[11] Haruna Z, Abdurrazaq MB, Umar A, Musa U. A Graphical User Interface for Path Planning of Mobile Robot. Ife Journal of Science and Technology. 2019;**3**(1): 60-73. Available from: www.ifejost.org

[12] Haruna Z, Musa U, Mu'azu MB, Umar A. A path planning technique for autonomous mobile robot. International Journal of Mechatronics, Electrical and Computer Technology. 2020;**10**(35): 4483-4492. Available from: https://www.aeuso.org/

[13] Chen X, Zhou M, Huang J, Luo Z. Global path planning using modified firefly algorithm. In: 2017 International Symposium on Micro-NanoMechatronics and Human Science (MHS). Nagoya, Japan: IEEE; 2017. pp. 1-7

[14] Eslami A, Asadi S, Soleymani GR, Azimirad V. A real-time global optimal path planning for mobile robot in dynamic environment based on artificial

immune approach. GSTF Journal on Computing. 2014;**2**(1):104-109

[15] Lv T, Feng M. A smooth local path planning algorithm based on modified visibility graph. Modern Physics Letters B. 2017;**31**(19–21):1740091

[16] Abbadi A, Přenosil V. Safe path planning using cell decomposition approximation. Distance Learning, Simulation and Communication. 2015;**8**: 1-6

[17] Khatib O. Real-time obstacle avoidance for manipulators and mobile robots. International Journal of Robotics Research. 1986;**5**(1):90-98

[18] Mohamed A, Ren J, Sharaf AM, EI-Gindy M. Optimal path planning for unmanned ground vehicles using potential field method and optimal control method. International Journal of Vehicle Performance. 2018; **4**(1):1-14

[19] Ali ZA, Han Z. Path planning of hovercraft using an adaptive ant colony with an artificial potential field algorithm. International Journal of Modelling, Identification and Control. 2021;**39**(4):350-356

[20] Subramanian S, George T, Thondiyath A. Obstacle avoidance using multi-point potential field approach for an underactuated flat-fish type AUV in dynamic environment. In: Trends in Intelligent Robotics, Automation, and Manufacturing: First International Conference, IRAM 2012, Kuala Lumpur, Malaysia, November 28-30, 2012. Proceedings. Kuala Lumpur, Malaysia: Springer; 2012. pp. 20-27

[21] Asano T, Asano T, Guibas L, Hershberger J, Imai H. Visibility-polygon search and euclidean shortest paths. In: 26th Annual Symposium on Foundations of Computer Science (SFCS 1985). Portland, OR, USA: IEEE; 1985. pp. 155-164

[22] Majeed A, Lee S. A fast global flight path planning algorithm based on space circumscription and sparse visibility graph for unmanned aerial vehicle. Electronics. 2018;**7**(12):375

[23] Canny J. A Voronoi method for the piano-movers problem. In: Proceedings. 1985 IEEE International Conference on Robotics and Automation. Vol. 2. MDPI; 1985. pp. 530-535

[24] Canny J, Reif J. New lower bound techniques for robot motion planning problems. In: 28th Annual Symposium on Foundations of Computer Science (sfcs 1987). Los Angeles, CA, USA: IEEE; 1987. pp. 49-60

[25] Canny J. The Complexity of Robot Motion Planning. London, England: The MIT Press; 1988

[26] Faverjon B, Tournassoud P. A local based approach for path planning of manipulators with a high number of degrees of freedom. In: Proceedings. 1987 IEEE International Conference on Robotics and Automation. Vol. 4. Raleigh, NC, USA; IEEE; 1987. pp. 1152-1159

[27] Uras T, Koenig S. Feasibility study: Subgoal graphs on state lattices. Proceedings of the International Symposium on Combinatorial Search. 2017;**8**(1):100-108

[28] Wang H, Yu Y, Yuan Q. Application of Dijkstra algorithm in robot path-planning. In: 2011 Second International Conference on Mechanic Automation and Control Engineering. Hohhot: IEEE; 2011. pp. 1067-1069

[29] Szczepanski R, Tarczewski T. Global path planning for mobile robot based on

Artificial Bee Colony and Dijkstra's algorithms. In: 2021 IEEE 19th International Power Electronics and Motion Control Conference (PEMC). Gliwice, Poland: IEEE; 2021. pp. 724-730

[30] Gigras Y, Vasishth O. Comparison of BAT with PSO for path planning problems. International Journal of Engineering Research and Development. 2015;**3**(2):590-595

[31] Zhang L, Li Y. Mobile robot path planning algorithm based on improved a star. Journal of Physics: Conference Series. 2021;**1848**(1):12013

[32] Raheem FA, Hameed UI. Path planning algorithm using D* heuristic method based on PSO in dynamic environment. American Scientific Research Journal for Engineering, Technology and Sciences. 2018;**49**(1): 257-271

[33] Stentz A. The D* Algorithm for Real-Time Planning of Optimal Traverses. Pittsburgh, Pennsylvania: Carnegie Mellon University, the Robotics Institute; 1994

[34] Holland JH. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. London, England: The MIT Press; 1992

[35] Eberhart R, Kennedy J. A new optimizer using particle swarm theory. In: MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science. Nagoya, Japan: IEEE; 1995. pp. 39-43

[36] Lamini C, Benhlima S, Elbekri A. Genetic algorithm based approach for autonomous mobile robot path planning. Procedia Computer Science. 2018;**127**: 180-189

[37] Sahu D, Mishra AK. Mobile robot path planning by genetic algorithm with safety parameter. International Journal of Engineering and Computer Science. 2017;**7**(8):14723-14727

[38] Haruna Z, Mu'azu MB, Abubilal KA, Tijani SA. Development of a modified bat algorithm using elite opposition—Based learning. In: 2017 IEEE 3rd International Conference on Electro-Technology for National Development (NIGERCON). Owerri, Nigeria: IEEE; 2017. pp. 144-151

[39] Ali ZA, Han Z, Masood RJ. Collective motion and self-organization of a swarm of UAVs: A cluster-based architecture. Sensors. 2021;**21**(11):3820

[40] Garba I, Sha'aban YA, Mu'azu MB, Haruna Z. Crone controller based speed control of permanent magnet direct current motor. In: 2019 2nd International Conference of the IEEE Nigeria Computer Chapter (NigeriaComputConf). Zaria, Nigeria: IEEE; 2019. pp. 1-8

[41] Vallade B, Nakashima T. Improving particle swarm optimization algorithm and its application to physical travelling salesman problems with a dynamic search space. Applied Computing & Information Technology. 2014;**553**:105-119

[42] Dorigo M, Blum C. Ant colony optimization theory: A survey. Theoretical Computer Science. 2005;**344** (2–3):243-278

[43] Ali ZA, Zhangang H, Hang WB. Cooperative path planning of multiple UAVs by using max–min ant colony optimization along with cauchy mutant operator. Fluctuations and Noise Letters. 2021;**20**(01):2150002

[44] Tian J, Yu W, Xie S. An ant colony optimization algorithm for image edge

detection. In: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence). Vol. 2008. Hong Kong, China: IEEE; 2008. pp. 751-756

[45] Yang X-S, Deb S. Cuckoo search via Lévy flights. In: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC). Coimbatore, India: IEEE; 2009. pp. 210-214

[46] Audee SY, Mu'azu MB, Sani M-Y, Haruna Z, Salawudeen AT, Prosper O. Development of a dynamic cuckoo search algorithm. Covenant Journal of Informatics and Communication Technology. 2019;**7**(2):66-83

[47] Yang X-S. A new metaheuristic bat-inspired algorithm. In: Nature Inspired Cooperative Strategies for Optimization (NICSO 2010). Vol. 2010. Berlin, Heidelberg: Springer; 2010. pp. 65-74

[48] Haruna Z, Mu'azu MB, Abubakar YS, Adedokun EA. Path tracking control of four wheel unmanned ground vehicle using optimized FOPID controller. In: 2021 International Conference on Electrical, Communication, and Computer Engineering (ICECCE). Kuala Lumpur, Malaysia: IEEE; 2021. pp. 1-6

[49] Bassil Y. Neural network model for path-planning of robotic rover systems. arXiv Prepr. arXiv1204.0183. Vol. 2. 2012. pp. 94-100

[50] Lippmann R. Book review: Neural networks, a comprehensive foundation, by simon haykin. International Journal of Neural Systems. 1994;**5**(04):363-364

[51] Jin L, Li S, Yu J, He J. Robot manipulator control using neural networks: A survey. Neurocomputing. 2018;**285**:23-34

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

[53] Das T, Kar IN. Design and implementation of an adaptive fuzzy logic-based controller for wheeled mobile robots. IEEE Transactions on Control Systems Technology. 2006; **14**(3):501-510

[54] Teymournezhad M, Sahingoz OK. Fuzzy logic-based trajectory planning for mobile robots in an uncertain and complex environment. In: 2023 2nd International Conference on Computational Systems and Communication (ICCSC). Thiruvananthapuram, India: IEEE; 2023. pp. 1-6

[55] Raja P, Pugazhenthi S. Optimal path planning of mobile robots: A review. International Journal of Physical Sciences. 2012;**7**(9):1314-1320

[56] Gigras Y, Gupta K. Ant colony based path planning algorithm for autonomous robotic vehicles. International Journal of Artificial Intelligence and its Applications. 2012;**3**(6):31

[57] Gigras Y, Gupta K. Metaheuristic algorithm for robotic path planning. International Journal of Computers and Applications. 2014;**85**(3):26-29

[58] Gigras Y, Gupta K, Choudhury K. A comparison between bat algorithm and cuckoo search for path planning. The International Journal of Innovative Research in Computer and Communication Engineering. 2015;**3**(5): 4459-4466

[59] Mohanty PK, Parhi DR. Optimal path planning for a mobile robot using cuckoo search algorithm. Journal of

Experimental & Theoretical Artificial Intelligence. 2016;**28**(1–2):35-52

[60] Kaplan A, Kingry N, Uhing P, Dai R. Time-optimal path planning with power schedules for a solar-powered ground robot. IEEE Transactions on Automation Science and Engineering. 2016;**14**(2): 1235-1244

[61] Yang Q, Hu J, Wang M, Yu H, Peng Q. Lane-level path guiding method for unmanned ground vehicle. In: 2017 5th International Conference on Computer, Automation and Power Electronics (CAPE). UK: Francis Academic Press; 2017. pp. 175-181

[62] Roy N, Chattopadhay R, Mukherjee A, Bhuiya A, Student BT. Implementation of image processing and reinforcement learning in path planning of mobile robots. International Journal of Engineering Science. 2017;**15211**: 15211-15213

[63] Han J, Seo Y. Mobile robot path planning with surrounding point set and path improvement. Applied Soft Computing. 2017;**57**:35-47

[64] Al-Jarrah R, Al-Jarrah M, Roth H. A novel edge detection algorithm for mobile robot path planning. Journal of Robotics. 2018;**2018**:1-12

[65] Neydorf R, Yarakhmedov O, Polyakh V, Chernogorov I, Vucinic D. Robot path planning based on ant colony optimization algorithm for environments with obstacles. Improved Performance of Materials. 2018;**72**:175-184

[66] Rostami SMH, Sangaiah AK, Wang J, Liu X. Obstacle avoidance of mobile robots using modified artificial potential field algorithm. EURASIP Journal on Wireless Communications and Networking. Auckland, New Zealand: IEEE: 2019;**2019**(1):1-19

[67] Sedighi S, Nguyen D-V, Kapsalas P, Kuhnert K-D. Implementing voronoi-based guided hybrid a in global path planning for autonomous vehicles. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). Auckland, New Zealand: IEEE; 2019. pp. 3845-3852

[68] Orozco-Rosas U, Picos K, Montiel O. Hybrid path planning algorithm based on membrane pseudo-bacterial potential field for autonomous mobile robots. IEEE Access. 2019;**7**:156787-156803

[69] Ajeil FH, Ibraheem IK, Sahib MA, Humaidi AJ. Multi-objective path planning of an autonomous mobile robot using hybrid PSO-MFB optimization algorithm. Applied Soft Computing. 2020;**89**:106076

[70] Hu X, Chen L, Tang B, Cao D, He H. Dynamic path planning for autonomous driving on various roads with avoidance of static and moving obstacles. Mechanical Systems and Signal Processing. 2018;**100**:482-500

[71] Gigras Y, Jora N, Dhull A. Comparison between different meta-heuristic algorithms for path planning in robotics. International Journal of Computers and Applications. 2016; **142**(3):6-10

Chapter 6

Control Barrier Functions and LiDAR-Inertial Odometry for Safe Drone Navigation in GNSS-Denied Environments

Halil Utku Unlu, Dimitris Chaikalis, Vinicius Gonçalves and Anthony Tzes

Abstract

This chapter is concerned with drone navigation in unknown, indoor environments. This necessitates using the onboard LiDAR and IMU sensors to solve the simultaneous localization and mapping (SLAM) problem. Control barrier functions (CBFs) augmented with circulation constraints are designed for motion planning. CBFs ensure that the drone can safely navigate the unknown environment by avoiding obstacle collisions. The FAST-LIO package is used for SLAM and the generated OctoMap data are transmitted to the CBF-module motion planning algorithm. Simulation studies using the Gazebo Physics Engine with a coaxial hexarotor drone are provided to validate the efficacy of the suggested algorithm.

Keywords: SLAM, control barrier functions, motion planning, unmanned aerial vehicles, safe navigation

1. Introduction

1.1 Motivation

Navigation in unknown environments [1] using Unmanned Aerial Vehicles (UAVs) is an important research area. This task requires integrating many different techniques and algorithms in obstacle avoidance, localization, mapping, and control. For a UAV operating outdoors [2], the integration of a Global Navigation Satellite System (GNSS) can provide accurate position information. When we consider indoor exploration, we cannot assume a consistent presence of GNSS signals. UAVs navigating in unknown indoor confined environments without GNSS coverage [3] provide significant challenges that jeopardize a safe autonomous mission.

1.2 Recent work

Previous works that handled this problem include [4], in which the UAV navigates in an unknown environment while requiring prior knowledge of certain features. Furthermore, Matos-Carvalho et al. [5] presents an algorithm for accurate indoor UAV navigation, using several fixed radio stations to improve the UAV positioning. Such solutions are however not applicable to navigation of purely unknown environments. In the case of multiple deployed agents, relative measurements have been used to guarantee formation control without external positioning [6], but the global planning problem remains unsolved.

The overall problem of safe navigation can be essentially divided into three different problems: (i) to map and locate the robot in the environment, (ii) to plan a movement that achieves the goal given the map, and (iii) to manipulate the lower-level controllers to follow the given plan.

For problem (i), the *mapping and localization problem*, we need the vehicle to be able to keep an estimate of its position with respect to the environment and integrate sensor information at a given position to generate a representation of the surrounding environment. However, simply relying on odometry will lead to inconsistencies for a lengthy mapping session [7], requiring the algorithm to recognize that the vehicle has been at a particular location more than once. The recognition and subsequent correction of the map is termed *loop closure*.

With regard to odometry, vision provides an abundance of information, making it very useful for odometry estimation in UAVs. Camera-only [8] and vision-inertial sensor coupled odometry [9, 10] are viable options. In [11], a hybrid point-line feature-based localization is proposed for UAVs, without needing an IMU sensor. Similarly, deep neural networks are leveraged in [12] for UAV localization in emergency situations. A recent example of visual-inertial localization implementation aimed towards UAVs is described in [13].

Mapping entails storing a representation of the environment for future retrieval. Different representations that have been studied include visual features [11, 14], depth measurements [15, 16], and semantic descriptions [17]. For a more thorough coverage of issues pertaining to the SLAM problem at large, we direct the reader to [18], which covers both the front-end (sensor modality) and the back-end (representation and optimization) aspects of SLAM.

The problem (ii), the *motion planning problem*, has been widely studied. For example, we have optimization-based [19] or classical probabilistic algorithms as Rapid Random Trees [20] and Probabilistic Roadmaps [21], algorithms capable of planning the shortest possible paths [22–25], among others. However, these algorithms often require that the map is known *a priori*, which is often not the case in many practical situations. There are works that use artificial intelligence in order to achieve autonomous navigation of unknown areas [26, 27]. However, such methods often do not offer guarantees.

Algorithms for motion planning in unknown environments have to handle two main problems: (a) providing efficient exploration strategies and (b) planning safe paths along this exploration. In this work, we provide a motion planning algorithm that achieves exploration through a careful selection of *frontier points*, that is, points in the border between the known and unknown areas. The algorithm has a metric to select promising points—considering we aim to achieve a fixed target—and accomplish the high-level planning to these points using a graph structure built while navigating. We then need a *low-level planner*, that is, an algorithm that plans a safe path towards these points. This problem is usually handled by considering a straight line between the points and verifying if they are safe. In this work, this is attained using the combination of several strategies. We leverage the properties of control barrier functions (CBF) [28] to generate a safe path by solving a sequence of Quadratic Programs (QP). These QPs include a modification proposed in [29] that enhances the obstacle-avoidance capabilities of the planner. This safe path is then converted into a linear velocity command to the lower-level controllers using a *vector field* approach proposed into [30].

For problem (iii), the *low-level controller problem*, most commercial UAVs come with autopilot software capable of controlling the motors of the vehicle and allow the users to provide inputs to dictate the flight of the system [31]. In order to keep the developed planners applicable to UAVs with standard autopilots, the control of the UAV is maintained as a simple cascade PID structure in this work. The planner module provides linear velocities which are turned into desired attitude and thrust with a simple PID. Another series of PID controllers are tasked with achieving attitude control of the vehicle, emulating the performance of commonly available flight controllers.

The *mission* is to control a drone in an indoor environment to achieve a desired position while avoiding collision with the environment, under the following assumptions:

- The coordinates of the target are known, specified in a world frame $\mathcal{W}$.
- The drone maintains a given height during its flight, that is, that the mission is achievable by selecting a suitable plane parallel to the ground. The $\mathbf{z}$ vector of the world frame $\mathcal{W}$ is orthogonal to the ground, and thus a constant height means a constant z component when the position is measured in the world frame. Given this planarity assumption, the desired 2D position is termed $\mathbf{p}_{GLOBAL}$.
- The drone is equipped with a LiDAR and IMU.
- The drone's lower-level inputs (motors' thrust) can be controlled.

1.3 Contributions

Figure 1 shows schematics of the suggested approach. The "Algorithm" is divided into three modules: "SLAM", "global planner" and "lower-level controller". Each one of the modules will be discussed separately in the next subsections, but, overall, the "SLAM" module (Section 2) receives data from the sensors, providing the 2D position $\mathbf{p}$ and current map $\mathcal{M}$ to the "global planner" module (Section 3). This module is responsible to generate the 2D linear velocity $\dot{\mathbf{p}}_d$ to accomplish the mission, and send it to the "lower-level controller" (Section 4) to manipulate the low-level command τ (the motors' thrust) to follow this reference. This is sent to the drone, which then provides the data, closing the loop.

More concretely, the contributions of this chapter include the use of CBFs with additional QP constraints to yield collision-free velocity commands. By keeping the outputs of the algorithm vehicle agnostic, the algorithm can be applied to any platform.

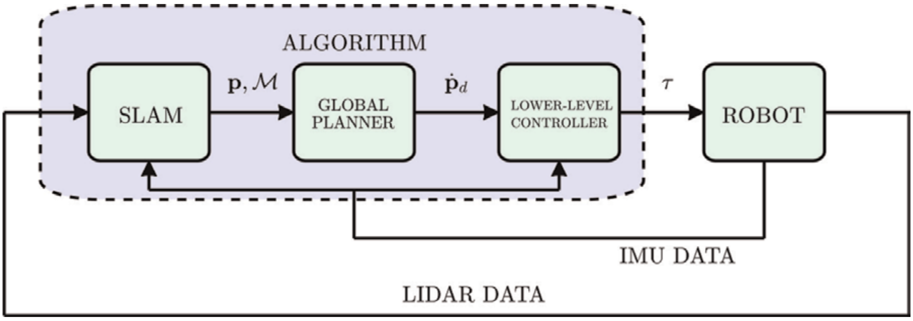


Figure 1.
Schematics of the algorithm.

1.4 Organization

In Sections 2–4 the three modules displayed in **Figure 1** are discussed. Then, in Section 5 simulation studies in the Gazebo environment showcase the algorithm's operation. Finally, in Section 6 we give our concluding remarks.

2. Simultaneous localization and mapping (SLAM) module

The indoor navigation sensor suite is comprised of a 3D LiDAR sensor, attached on top of the UAV, as shown in **Figure 2**, an Inertial Measurement Unit (IMU) on the flight controller, and a downward-facing rangefinder placed underneath the UAV. The addition of the rangefinder provides a robust estimate of absolute distance to the ground since the LiDAR sensor is not expected to sense the floor due to its placement on the top of the vehicle.

The LiDAR point cloud and corresponding IMU measurements are fed to the FAST-LIO algorithm [32], which estimates odometry in a tightly-coupled iterated extended Kalman filter by fusing 3D feature points from LiDAR and measurements from IMU. Odometry from FAST-LIO is reliable for orientation and motion on the *xy*-plane.

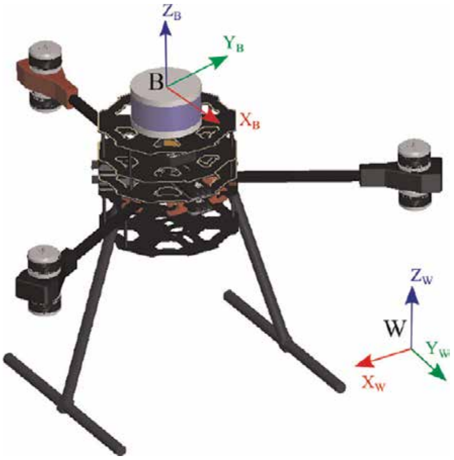


Figure 2.
Depiction of UAV agent with relevant coordinate frames.

An extended Kalman filter (EKF) [33] fuses IMU linear accelerations and angular velocities, rangefinder relative height measurements, and LiDAR-inertial odometry, resulting in an estimate of the UAV state. Let a world-fixed inertial frame be $\mathcal{W}$ and a body-fixed frame be $\mathcal{B}$ on the UAV IMU. Then the UAV's state, as maintained at the EKF, is $\chi = \begin{bmatrix}\mathbf{q}^{\mathrm{T}} & \dot{\mathbf{q}}^{\mathrm{T}} & \mathrm{F}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$, where $\mathbf{q} = [x\ y\ z]^{\mathrm{T}}$ the UAV's 3D-position in $\mathcal{W}$ and $\mathrm{F} = [\phi\ \theta\ \psi]^{\mathrm{T}}$ the roll, pitch, and yaw Euler angles of the UAV respectively.

Odometry estimates of the UAV are used as the basis for 3D occupancy grid mapping. Poses of LiDAR sensor measurements are integrated into a 3D occupancy grid map $\overline{\mathcal{M}}^{3D} \subseteq \mathbb{R}^3 \times [0,1]$ with OctoMap [34] as its backbone. OctoMap framework stores the probability of occupancy of a 3D grid with an arbitrary size efficiently in an octree data structure. Let $\mathcal{Z}(t) = \{{}^{L}\mathbf{r} \in \mathbb{R}^3\}$ be the point cloud of LiDAR measurement at time t, corrected for the motion of the sensor and expressed in the frame coinciding with sensor's origin, and $\mathbf{T}_L = \begin{bmatrix}\mathbf{R}_L & \mathbf{q}_L \\ \mathbf{0}_{1\times 3} & 1\end{bmatrix} \in SE(3)$ be the pose of the LiDAR sensor in the world frame, with $\mathbf{R}_L \in SO(3)$ denoting a rotation matrix and $\mathbf{q}_L \in \mathbb{R}^3$ as the origin of the sensor. Probability of occupancy is reduced for every grid cell in $\overline{\mathcal{M}}^{3D}$ that lie in the line defined as

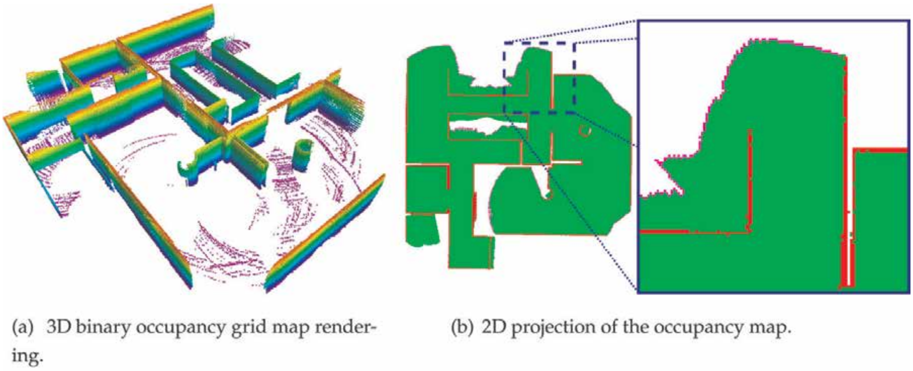
$$\mathbf{l}(d) \triangleq \mathbf{q}_L + d\frac{\mathbf{R}_L{}^{L}\mathbf{r}}{\|\mathbf{R}_L{}^{L}\mathbf{r}\|} \quad ; 0 \le d \le \min\left(\|\mathbf{R}_L{}^{L}\mathbf{r}\|, d_{\max}\right) \tag{1}$$

If the point is within a certain distance from the sensor origin, denoted as $d_{\max}$, then the probability of occupancy of the grid cell containing the endpoint is increased. The reason for including a maximum distance is mainly for efficiency purposes, as ray casting every point in $\mathcal{Z}(t)$ at every time step is an expensive process.

Instead of working with pure probabilities, the maximum likelihood estimate of the map $\overline{\mathcal{M}}^{3D}$ is obtained to classify the map into three distinct classes: occupied (1), free (-1), and unknown (0). The maximum likelihood map is denoted as $\mathcal{M}^{3D} \subseteq \mathbb{R}^3 \times \{-1,0,1\}$. Let the function $Class : \mathbb{R}^3 \to \{-1,0,1\}$ provide the class information for points in the 3D occupancy map.

For a continuous mapping mission with insufficient feature variety in an environment, a loop closure module to account for odometry drifts may be required. Minuscule errors in pose estimation via visual or visual-inertial odometry accumulate to introduce large inconsistencies. However, the odometry estimation obtained via the aforementioned EKF formulation provides sufficient accuracy for an indoor environment. Indoor spaces are seldom larger than the coverage a commercial 3D LiDAR sensor can provide. Therefore, a loop closure module is omitted, and the odometry estimates are relied on solely for mapping. However, it is possible to implement a 3D map-based loop closure module, similar to the vision-based approaches [35] using an equivalent global descriptor for depth data [36].

In this work, the navigation requirements of the UAV are limited to 2D. As such, only the occupancy of a subset of the 3D grid in $\mathcal{M}^{3D}$ that lies within a height range $z \in [h_{\min}, h_{\max}]$ is of importance. To consolidate the information contained in the given height range, the occupancy information is reduced to a 2D grid via the projection of each point coordinate onto xy-plane. More concretely, let the coordinates of an arbitrary grid center in $\mathcal{M}$ be denoted as $\mathbf{g}_i = \begin{bmatrix}x_i & y_i\end{bmatrix}^{\mathrm{T}}$, and the set of grid centers with



(a) 3D binary occupancy grid map rendering.

(b) 2D projection of the occupancy map.

Figure 3.
A 3D rendering of the occupancy grid view, denoting the occupied cells with color coding, along with a sample 2D projection of the occupancy map. Occupied, free, unknown, and frontier cells are depicted in red, green, black, and magenta colors respectively. A zoomed-in view of the 2D occupancy map is provided for clarity.

the same xy-coordinates in $\mathcal{M}^{3D}$ be denoted as $\mathbf{r}_i(z) = \begin{bmatrix} x_i & y_i & z \end{bmatrix}^{\mathrm{T}}$. Note that the xy-coordinates of the points are shared between $\mathbf{g}_i$ and $\mathbf{r}_i(z)$. We defined the projection function $Proj : \mathbb{R}^2 \rightarrow \{-1,0,1\}$ as

$$Proj(\mathbf{g}_i) \triangleq \begin{cases} 1 & \exists \ \mathbf{r}_i(z) \in \mathcal{M}^{3D}, \ z \in [h_{\min}, h_{\max}], \ Class(\mathbf{r}_i(z)) = 1 \\ -1 & \forall \ \mathbf{r}_i(z) \in \mathcal{M}^{3D}, \ z \in [h_{\min}, h_{\max}], \ Class(\mathbf{r}_i(z)) = -1 \\ 0 & \text{otherwise.} \end{cases} \qquad (2)$$

The projection function given in (2) is conservative: the 2D coordinates are considered occupied even if a single point in the column of all points within the given height range is occupied, and the cell is considered free only if all of the points in the same column are free.

An example projection from a 3D map to a 2D map is provided in **Figure 3(a)**. On the left is the binary OctoMap occupancy representation of the environment, color-coded based on the obstacle height. Upon projecting a height range of the map onto a 2D plane, the map representation on the right is obtained. Occupied, free, unknown, and frontier cells are represented in red, green, black, and magenta colors, respectively.

3. “Global planner” module

Important concepts and subroutines will be clarified prior to the motion planning algorithm.

3.1 Map and distance functions

For the motion planning, the *map* $\mathcal{M}$ implies a collection of points $\mathbb{R}^2$ that represent the geometry of the environment detected so far using the LiDAR sensor. As this information evolves as the drone navigates the environment, it changes continuously, the term $\mathcal{M}(t)$ emphasizes the map up to a specific time t.

It is essential for the motion planning algorithm to compute distances between the drone and the map in order to plan a safe path. Furthermore, distance gradients are also necessary for the approach. Before considering the geometry of the robot, we will be concerned with the distance between a *point* $\mathbf{p} \in \mathbb{R}^2$ and an obstacle. Since we have a collection of points $\mathcal{M} \subseteq \mathbb{R}^2$, described as $\mathbf{g}$, a first idea is to compute the point-to-map distance $E_P(\mathbf{p}|\mathcal{M})$ as the minimum Euclidean distance between $\mathbf{p}$ and the points $\mathbf{g}$. However, there is an issue with this approach: the function $E_P(\mathbf{p}|\mathcal{M})$ is not differentiable in $\mathbf{p}$ due to two reasons: the fact that the individual Euclidean distances $\|\mathbf{p} - \mathbf{g}\|$ are non-differentiable on $\mathbf{p}$ and also the fact the minimum function is non-differentiable at some points. The first issue can be easily solved by using the squared Euclidean distance $\|\mathbf{p} - \mathbf{g}\|^2$, and the second can be addressed by using an approximated version of the minimum that can be guaranteed to be differentiable. The *softmin* function [37] is a good candidate, defined (with an averaging term) as

$$\overset{h}{\min}\left(g_1, g_2, \ldots, g_m\right) \triangleq -h \ln\left(\frac{1}{m}\sum_{k=1}^{m} \exp\left(-\frac{g_k}{h}\right)\right). \tag{3}$$

This function is always differentiable in the arguments g_i and approximates the minimum when h is positive and close to 0. With softmin as the minimum, the *point-to-map squared distance* is defined

$$E_P(\mathbf{p}|\mathcal{M}) \triangleq \overset{h}{\min_{\mathbf{g} \in \mathcal{M}}} \|\mathbf{p} - \mathbf{g}\|^2 \tag{4}$$

in which h is a parameter. With this definition, the gradient of $E_P(\mathbf{p}|\mathcal{M})$ is computed as

$$\nabla_{\mathbf{p}} E_P(\mathbf{p}|\mathcal{M}) = \mathbf{p} - \frac{\sum_{\mathbf{g} \in \mathcal{M}} \exp\left(-\|\mathbf{p} - \mathbf{g}\|^2/h\right)\mathbf{g}}{\sum_{\mathbf{g} \in \mathcal{M}} \exp\left(-\|\mathbf{p} - \mathbf{g}\|^2/h\right)}. \tag{5}$$

The drone is modeled by a sphere of radius R_{ROB}. Thus, using the previously-defined point-to-map distance, the *robot-to-map squared distance* is defined as $E(\mathbf{p}|\mathcal{M}) \triangleq E_P(\mathbf{p}|\mathcal{M}) - R_{ROB}^2$, in which $\mathbf{p} \in \mathbb{R}^2$ given by $\mathbf{p} = [x \ y]^{\mathrm{T}}$ is the drone's 2D position. Note that this is different than $\mathbf{q} = [x \ y \ z]^{\mathrm{T}}$ introduced in the previous section: $\mathbf{p}$ is the projection of $\mathbf{q}$ in a plane parallel to the ground. Since the planner is 2D only, it does not require the height z. Nevertheless, we can easily compute the gradient of this function as $\nabla_{\mathbf{p}} E(\mathbf{p}|\mathcal{M}) = \nabla_{\mathbf{p}} E_P(\mathbf{p}|\mathcal{M})$, in which $\nabla_{\mathbf{p}} E_P(\mathbf{p}|\mathcal{M})$ is computed according to (5).

3.2 Control barrier functions with circulation constraints

The utilized global planner needs a *local planner* that plans a (usually short) path from one point to another while avoiding obstacles. It is common to plan using a straight line, and the planner is successful if this line does not cross any obstacle within a safety margin. Although this strategy is simple to implement, it may be inefficient because the path may be often blocked by obstacles, requiring many calls to the local planner until a path (formed by line segments) is found. We will propose another approach that, although not as simple to implement as a straight line, provides successful paths more often because it is able to deviate from the obstacles.

Suppose the existence of a point $\mathbf{p}$ in which its velocity $\dot{\mathbf{p}} = \mathbf{u}$ is controlled. This control input $\mathbf{u}$ should guide $\mathbf{p}$ from its starting position to a final position $\mathbf{p}_G \in \mathbb{R}^2$, while avoiding collision with the map $\mathcal{M}$. During the planning phase starting at time t_0, the map $\mathcal{M}(t)$ is assumed to be constant, being $\mathcal{M}(t_0)$.

$\dot{\mathbf{p}}$ is computed using *control barrier functions with circulation constraints* [29]. At each point $\mathbf{p}$ the velocity $\mathbf{u}(\mathbf{p})$ is computed by solving a QP problem that guides the point towards the target while avoiding obstacles. This QP has two constraints: one directly from the CBF, and another derived from it, that induces a *circulation behavior*. The circulation behavior is a novelty introduced in [29] that helps deviate from obstacles with a negligible increase in the cost of the algorithm.

The following parameters of this formulation are defined:

- Let $\Omega \in \mathbb{R}^{2\times2}$ be a skew-symmetric matrix, that determines how the environment is circulated;

- Let $\alpha : \mathbb{R} \mapsto \mathbb{R}$ be a continuous, decreasing function with $\alpha(0) = 0$, that determines how fast obstacles can be approached given how close are;

- Let $\beta : \mathbb{R} \mapsto \mathbb{R}$ be a continuous, decreasing function such with $\beta(0) > 0$ and $\beta(E(\mathbf{p}_G|\mathcal{M})) < 0$, that determines how much an obstacle is forced to be circulated given how close to it the robot is.

- Let K_C be a positive constant that determines the approach speed to the target.

Furthermore, define the *normal* and *tangent* vectors:

$$\mathbf{N}(\mathbf{p}) \triangleq \frac{\nabla_{\mathbf{p}} E(\mathbf{p}|\mathcal{M})}{\|\nabla_{\mathbf{p}} E(\mathbf{p}|\mathcal{M})\|} \quad \text{and} \quad \mathbf{T}(\mathbf{p}) \triangleq \frac{\Omega \nabla_{\mathbf{p}} E(\mathbf{p}|\mathcal{M})}{\|\Omega \nabla_{\mathbf{p}} E(\mathbf{p}|\mathcal{M})\|} \tag{6}$$

Both are well defined (i.e $\nabla_{\mathbf{p}} E(\mathbf{p}|\mathcal{M})$ and $\Omega \nabla_{\mathbf{p}} E(\mathbf{p}|\mathcal{M})$ are non-null vectors). Finally, define $\mathbf{u}_d(\mathbf{p}) \triangleq -K_C(\mathbf{p} - \mathbf{p}_G)$. Then the CBF + QP formulation to compute $\mathbf{u}(\mathbf{p})$ is as follows [29]:

$$\mathbf{u}(\mathbf{p}) = \arg\min_{\mu} \|\mu - \mathbf{u}_d(\mathbf{p})\|^2 \tag{7}$$

$$\text{such that :} \qquad \mathbf{N}(\mathbf{p})^{\mathrm{T}} \mu \geq \alpha(E(\mathbf{p}|\mathcal{M})) \tag{8}$$

$$\mathbf{T}(\mathbf{p})^{\mathrm{T}} \mu \geq \beta(E(\mathbf{p}|\mathcal{M})). \tag{9}$$

This is a QP problem that always has a solution, and furthermore, this solution is unique because it is a strictly convex problem. Care must be taken when the problem is not well defined. This can be divided into three mutually exclusive conditions:

- $\Omega \nabla_{\mathbf{p}} E = \mathbf{0}$ but $\nabla_{\mathbf{p}} E \neq \mathbf{0}$. In this case, we just discard the tangent constraint. This is especially important because Ω being the 2 × 2 null matrix is a possibility.

- $\nabla_{\mathbf{p}} E = \mathbf{0}$ and $E > 0$. In this case, we can disregard the normal and tangential constraints and just use $\mathbf{u}(\mathbf{p}) = \mathbf{u}_d(\mathbf{p})$.

- $\nabla_{\mathbf{p}}E = \mathbf{0}$ and $E \leq 0$. In this case, we have to stop, since we do not have distance gradient information and we are dangerously close to obstacles.

The planner can fail in three different cases: (i) because the third condition is reached, (ii) because the robot has reached a point in which $\mathbf{u}(\mathbf{p}) = \mathbf{0}$ but $\mathbf{p} \neq \mathbf{p}_G$ or (iii) because a limit cycle has been reached.

3.3 Local planner function

With the description given in the previous subsection, we can define the CBFCircPlanOne function:

$$(\mathcal{P}, S) \leftarrow \texttt{CBFCircPlanOne}\left(\mathbf{p}_0, \mathbf{p}_G, \mathcal{M}, \Omega, T\right) \tag{10}$$

which is at the core of the motion planning algorithm. It takes as arguments a *starting point* $\mathbf{p}_0 \in \mathbb{R}^2$, a *goal point* $\mathbf{p}_G \in \mathbb{R}^2$, a *map* $\mathcal{M}$, a *circulation* Ω and a *time period* T, and tries, using the procedure described in the previous subsection, to plan a collision-free path from $\mathbf{p}_0$ to $\mathbf{p}_G$ while avoiding the obstacles detected in the map and using as a circulation parameter Ω, while integrating the dynamical system $\dot{\mathbf{p}} = \mathbf{u}(\mathbf{p})$ from $t = 0$ to *t at most T* time units. Note that the algorithm can halt before it reached $t = T$ in the integration, either because it detects one of the failure conditions described in the previous subsection or because it reached the goal. Nevertheless, it returns the path $\mathcal{P}$ it obtained until it halted and also a signal $S \in \{TRUE, FALSE\}$ saying if it was successful or not. Naturally, the resulting differential equation cannot be solved analytically except in very simple cases, and the path is computed using a numerical integration scheme. Related to this function, we define the CBFCircPlanMany function:

$$(\mathcal{P}, \Omega_{BEST}, S) \leftarrow \texttt{CBFCircPlanMany}\left(\mathbf{p}_0, \mathbf{p}_G, \mathcal{M}, T\right) \tag{11}$$

which plans many paths from $\mathbf{p}_0$ to $\mathbf{p}_G$, with the map $\mathcal{M}$ and with each one of them integrating until at most $t = T$, using different *previously-defined* rotations Ω. Thus, this function calls CBFCircPlanOne many times. It returns the *shortest path* $\mathcal{P}$ among all the successful paths, the best circulation matrix Ω_{BEST} that achieved this smallest successful path and a signal $S \in \{TRUE, FALSE\}$ saying if at least one of the paths was successful or not. In the proposed approach, three different matrices Ω were used, representing clockwise rotation, counter-clockwise rotation, and a non-rotation. More precisely, let

$$\Omega_Z \triangleq \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}, \Omega_0 \triangleq \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \tag{12}$$

then $+\Omega_Z$ (counter-clockwise), $-\Omega_Z$ (clockwise) and Ω_0 (no induced rotation) is used.

3.4 Vector field path tracking

The global planner periodically revises its plan and eventually commits the drone to a *path* $\mathcal{P}$. Once a path is given, the robot's low-level control input is computed - its linear velocity (see Section 4) - to track this path. For this, the vector field methodology is used [30]. The function $\dot{\mathbf{p}} \leftarrow \texttt{vectorField}(\mathbf{p}, \mathcal{P})$ will return the linear velocity $\dot{\mathbf{p}}$ to be sent to the lower level controllers given the current position $\mathbf{p}$ and the committed path $\mathcal{P}$.

3.5 Exploration graph generation

In order to run the motion planning algorithm, as the drone moves, a *graph* $\mathcal{G}$ is incrementally generated that represents the connectivity of the environment in a simplified and structured way when compared to the map $\mathcal{M}$. Each "node" N of the graph is associated with a position in the map. Two nodes/points $\mathbf{p}_A$ and $\mathbf{p}_B$ are connected if and only if a safe path is secured between $\mathbf{p}_A$ and $\mathbf{p}_B$. As this graph is updated/built as the drone moves, it is also time-variant, described by $\mathcal{G}(t)$.

Associated with this graph, the following functions can be found:

- $\mathcal{G}' \leftarrow \texttt{updateGraph}(\mathbf{p}, \mathcal{G}, \mathcal{M})$, which updates the current graph $\mathcal{G}$, given the current position $\mathbf{p}$ and the current known map $\mathcal{M}$, returning the updated graph $\mathcal{G}'$. It tries to insert a new node associated with the current position $\mathbf{p}$ by trying to connect this node to another one already on $\mathcal{G}$ (the closest one in which the connection is possible), taking into consideration the obstacles codified into the map $\mathcal{M}$. The graph is undirected and weighted by the distance between the two nodes.

- $\mathcal{N} \leftarrow \texttt{getNearestNodes}(\mathbf{p}, \mathcal{G})$ returns a sorted list of all the nodes $\mathcal{N}$ of the graph $\mathcal{G}$ sorted by their Euclidean distance to the point $\mathbf{p}$ in ascending order.

- $(N, L) \leftarrow \texttt{getNearestReachableNode}(\mathbf{p}, \mathcal{G}, \mathcal{M})$ returns a node N of the graph $\mathcal{G}$ such that there is a path from $\mathbf{p}$ to N, considering the obstacles codified into $\mathcal{M}$. It also returns the length L of this path. This function can be implemented as follows: getNearestNodes is used to get a list of nodes, sorted by the Euclidean distance to $\mathbf{p}$. Then CBFCircPlanMany is used in each one of them until the planner is successful. The first successful node is returned as N. The assumption is that this algorithm never fails, i.e., the graph $\mathcal{G}$ is a good enough representation of the map $\mathcal{M}$ and the local planner is good enough to always move from the current position to one of the nodes of the graph.

- $(\mathcal{N}, L) \leftarrow \texttt{getPath}(N_A, N_B, \mathcal{G})$ returns a list of nodes $\mathcal{N}$ containing the shortest path from node N_A to node N_B given the graph $\mathcal{G}$, and also the length (sum of the weights between the edges) of this path. This shortest path considers only the weights codified into the edges of the graph, does not take into consideration the map $\mathcal{M}$, and can be computed using (for example) Dijkstra's algorithm. Note that, given the nature of the graph building scheme described here, such a path always exists between two given nodes, since it only adds a node to the graph (using the updateGraph function) if a path between that node and another node (already in the graph) exists.

3.6 Frontier points and frontier exploration

Eventually, the motion planner should request the robot to go to an unexplored region of the map [33]. For this, a function $\mathcal{F} \leftarrow \texttt{getAllFrontiers}(\mathcal{M})$ is implemented that obtains a set of *representative* frontier points $\mathcal{F}$ from the currently-known map $\mathcal{M}$.

A *frontier point* is defined as a coordinate in 2D binary occupancy map $\mathcal{M}$ which is classified as a free cell, is in the 8-neighborhood of a cell that is marked unknown, and is not in the 8-neighborhood of a cell that is marked occupied. To this end, standard image processing tools are utilized by treating the 2D binary occupancy as a binary image with 2 channels (maps): an *occupied channel*, denoted $Oc : \mathbb{N}^2 \rightarrow \{0, 1\}$, with pixel value of 1 if the coordinate is occupied, and a *free channel*, denoted as $Fr : \mathbb{N}^2 \rightarrow \{0, 1\}$, whose coordinates take the value 1 for free coordinates. Due to the definition of the projection as given in (2), a coordinate cannot have both values in the two channels be 1. Implicitly, any coordinate at which both channels have a 0 value is unknown.

The occupied channel Oc is firstly dilated by 1 pixel to obtain the image $\overline{Oc}$. Then the internal and external contours of the free channel Fr are extracted, denoted as $\mathcal{C} \subseteq \mathbb{N}^2$. For every contour point $\mathbf{g}$, the corresponding value in $\overline{Oc}$ is checked at the pixel coordinate given by $\mathbf{g}$. All the points $\mathbf{g} \in \mathcal{C}$ that have a 0 value in $\overline{Oc}$ constitute the frontier, $\mathcal{F}$. More concretely:

$$\mathcal{F} = \{\mathbf{g} \in \mathcal{C}, \overline{Oc}(\mathbf{g}) = 0\}. \tag{13}$$

A sample of frontier points is marked in **Figure 3(b)**, which can be seen as the magenta points in the zoomed-in view.

Once these points are obtained, an algorithm decides which point it is going to explore next. This is achieved by function $\{\mathcal{N}, S\} \leftarrow$ `selectExplorationFrontier` $(\mathbf{p}, \mathcal{M}, \mathcal{F}, \mathcal{G})$ that processes the current position $\mathbf{p}$, the current map $\mathcal{M}$, the frontier points $\mathcal{F}$ and the graph $\mathcal{G}$, and returns the path one should travel from the points of the graph $\mathcal{G}$ to reach one of the frontier points (if one is found) and a signal $S \in \{TRUE, FALSE\}$ if it was successful.

The algorithm's structure is:

- **Step 1**: For each point $\mathbf{p}_F \in \mathcal{F}$ the `CBFCircPlanMany` function is used to try to connect N_{TRY} (a parameter) different nodes of $\mathcal{G}$ to the point $\mathbf{p}_F$. The N_{TRY} closest nodes from $\mathbf{p}_F$, obtained from the `getNearestNodes` functions are tried. The points in which it was not possible to connect none of these N_{TRY} nodes to $\mathbf{p}_F$ are discarded, meaning that these frontier points are most likely unreachable, obtaining a new set $\mathcal{F}' \subseteq \mathcal{F}$. For each point in $\mathbf{p}_F \in \mathcal{F}'$, this procedure also allows us to create two maps: $N \leftarrow$ `closestSuccesfulNode` $[\mathbf{p}_F]$ that maps the frontier point into the node N of the graph $\mathcal{G}$ that has the shortest successful path from it, and $L \leftarrow$ `shortPathLengthFromGraphToFrontier` $[\mathbf{p}_F]$ that gives the length of this shortest successful path.
    - If $\mathcal{F}' = \varnothing$, the algorithm **ends**, since most likely anything anymore can be explored.
    - Otherwise, proceed to **Step 2**.

- **Step 2**: For each one of the positions in $\mathbf{p}_F \in \mathcal{F}'$, the `CBFCircPlanMany` function is used to try to connect this point to the *global target* $\mathbf{p}_{GLOBAL}$. Those points in which we were successful are aggregated into a set $\mathcal{F}''$, and this procedure allows us to create a map $L \leftarrow \texttt{shortPathLengthFromFrontierToGoal}[\mathbf{p}_F]$ with this smallest length.

  - If $\mathcal{F}'' \neq \varnothing$, proceed to **Step 3**.

  - Otherwise, proceed to **Step 4**.

- **Step 3**: For each one of the positions in $\mathbf{p}_F \in \mathcal{F}''$, a metric is computed that decides which point is going to be explored. This metric is given by the sum of

  - The path length L_p^{GStart} from $\mathbf{p}$ to a node N_{GStart} in the graph, obtained from $\left(N_{GStart}, L_p^{GStart}\right) \leftarrow \texttt{getNearestReachableNode}(\mathbf{p}, \mathcal{G}, \mathcal{M})$;

  - The path length L_{GStart}^{GEnd} from N_{GStart} to the node $N_{GEnd} = \texttt{closestSuccesfulNode}[\mathbf{p}_F]$, obtained from `getPath` (N_{GStart}, N_{GEnd});

  - The path length L_{GEnd}^{F} from N_{GEnd} to the frontier point $\mathbf{p}_F$ using the shortest planned path, which is $L_{GEnd}^{F} = \texttt{shortPathLengthFromGraphToFrontier}[\mathbf{p}_F]$;

  - The path length L_F^{GLOBAL}, which is

  $$L_F^{GLOBAL} = \texttt{shortPathLengthFromFrontierToGoal}[\mathbf{p}_F].$$

  The point $\mathbf{p}_F$ is returned with the smallest such metric, the list of nodes from N_{GStart} and N_{GEnd} as $\mathcal{N}$, together with a dummy node at the end (that is not on $\mathcal{G}$) that represents the frontier point $\mathbf{p}_F$, and thus the algorithm **ends**. This metric is an estimate of how long the optimistic path from the current point $\mathbf{p}$ to the global target $\mathbf{p}_{GLOBAL}$ is, assuming that the current map is the final map, and thus the most promising frontier points are selected to explore;

- **Step 4**: If $\mathcal{F}'' = \varnothing$, **Step 3** is reproduced but with $\mathcal{F}'$ instead of $\mathcal{F}''$ and without the last term, L_F^{GLOBAL} (since it was not possible to plan a path from the frontier to the goal).

3.7 The motion planning algorithm

With all the necessary functions introduced, the motion planning algorithm can be explained. Let the *state variables* of the planner, be internal variables that must be stored all the time and that are read and modified by the flow of the algorithm:

- The current committed path $\mathcal{P}_C$ that the vector field algorithm needs to follow, which is empty in the beginning.
- The current graph $\mathcal{G}$, starting with a graph with a single node, representing the starting position $\mathbf{p}_0$;
- The current rotation parameter Ω. This can be initialized as any one of the three possible rotations described in (12) (e.g: Ω_0);
- The current goal $\mathbf{p}_G$, which starts with the global goal $\mathbf{p}_{GLOBAL}$;
- The current navigation mode V, that can be either `goingToGlobal` or `goingToFrontier`. It starts with `goingToGlobal`;
- The current list of nodes from $\mathcal{G}$ that we should track, $\mathcal{N}_C$, along with the current node on this list, $N_C \in \mathcal{N}$. These two states are only used when $V = \texttt{goingToFrontier}$, and since V starts with `goingToGlobal`, it does not matter how they are initialized.

As shown in **Figure 1**, the inputs of the algorithm are the current position $\mathbf{p}$ and map $\mathcal{M}$ from the SLAM algorithm, and the output is the desired linear velocity $\dot{\mathbf{p}}_d$ to be provided to the lower level controller. It has three different sub-algorithms, labeled as A, B, and C, that runs in different frequencies (and the latter asynchronously as well) and reads and modifies the state variables.

In a given frequencyf_A—which is the frequency in which the algorithm is going to output the linear velocity—the planner uses the vector field algorithm to track the committed path $\mathcal{P}_C$, providing $\dot{\mathbf{p}}$, and also check if the current target $\mathbf{p}_G$ was reached. In a frequency$f_B < f_A$, it replans the committed path $\mathcal{P}_C$ based on the current goal $\mathbf{p}_G$, and, finally, in a frequency$f_C < f_B$, it updates the graph, replans the goal $\mathbf{p}_G$ itself, the current Ω and the current navigation mode V.

A more detailed description of each sub-algorithm is shown in the sequel:

- **Algorithm A**: running at a frequencyf_A. This algorithm is responsible to verify if the target was achieved, changing the flow of the algorithm if it is the case. It may call **Algorithm C** below. It also sends commands to the lower-level controller.
    - **Step 1.A**: Check whether the current goal is reached by checking if $\|\mathbf{p} - \mathbf{p}_G\| \leq \delta$ for a parameter δ.
        a. If this is false, go to **Step 2.A**;
        b. If this is true and V==`goingToGlobal`, the mission is accomplished and **the whole algorithm ends**.
        c. If this is true and V==`goingToFrontier`, check whether the node N_C is the last on the list $\mathcal{N}_C$, or not.
            1. If not, set N_C to the next of the list and set $\mathbf{p}G \leftarrow N_C$. Call **Algorithm C** and, when it halts, go to **Step 2.A**.

2. If it was the last node of the list, set $V \leftarrow$ `goingToGlobal`, set $\mathbf{p}G \leftarrow \mathbf{p}GLOBAL$. Call **Algorithm C** and, when it halts, go to **Step 2.A**.

- **Step 2.A**: compute $\dot{\mathbf{p}}_d \leftarrow \texttt{vectorField}(\mathbf{p}, \mathcal{P}_C)$ and output it to the lower level controller. Algorithm A then **ends**.

- **Algorithm B**: running at a frequencyf_B. This algorithm is responsible for updating the path that is being tracked. It consists of a single step, running

$$(\mathcal{P}_C, S) \leftarrow \texttt{CBFCircPlanOne}\left(\mathbf{p}_0, \mathbf{p}_G, \mathcal{M}, \Omega, T_B\right) \quad (14)$$

for a fixed time T_B (a parameter), updating $\mathcal{P}_C$. Irrelevant on whether it failed or not, since it is a small local path that is unlikely to reach the current target unless the drone is very close to it. Thus, disregard S. Algorithm B then **ends**.

- **Algorithm C**: running at a frequencyf_C, and asynchronously (when called by **Algorithm A**). It is responsible to replan the target $\mathbf{p}_G$ and also updating the graph.

  - **Step 1.C**: Run $\mathcal{G} \leftarrow \texttt{updateGraph}(\mathbf{p}, \mathcal{G}, \mathcal{M})$ to update the graph. Go to **Step 2.C**.

  - **Step 2.C**: Replan the path to the current goal by running

$$(\mathcal{P}, \Omega_{BEST}, S) \leftarrow \texttt{CBFCircPlanMany}\left(\mathbf{p}, \mathbf{p}_G, \mathcal{M}, T_C\right) \quad (15)$$

for a fixed time T_C (a parameter).

- If the algorithm was a success (S==$TRUE$), set $\Omega \leftarrow \Omega_{BEST}$ and Algorithm C **ends** (do not care about $\mathcal{P}$).

- If the algorithm fails (S==$FALSE$), it means that very likely it is no use to try to go to the current target point $\mathbf{p}_G$, and there is no need to change it. Explore a frontier point, and thus run $\mathcal{F} \leftarrow \texttt{getAllFrontiers}(\mathcal{M})$, and then $\{\mathcal{N}_E, S\} \leftarrow \texttt{selectExplorationFrontier}(\mathbf{p}, \mathcal{M}, \mathcal{F}, \mathcal{G})$.

  - If $S == TRUE$, set $V \leftarrow$ `goingToFrontier`, $\mathcal{N}_C \leftarrow \mathcal{N}_E$, N_C to the first node of $\mathcal{N}_C$, and $\mathbf{p}_G \leftarrow N_C$. Try to replan, going back to **Step 2.C**;

  - If $S == FALSE$, the **whole algorithm ends with failure**, because there is nothing to explore anymore and the global goal was not found;

4. "Low-level controller" module

Given the desired 2D linear velocity $\dot{\mathbf{p}}_d$ commanded by the global planner, this vector is augmented with a 0—since we want to maintain a constant height—at the end to obtain the 3D desired linear velocity $\dot{\mathbf{q}}_d$. The velocity controller computes a desired acceleration vector as

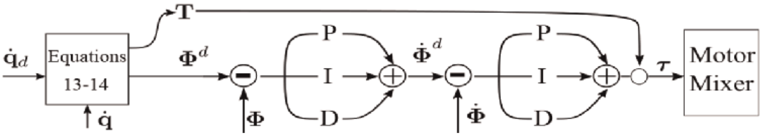


Figure 4.
Cascade PID structure of low-level attitude controller.

$$\alpha_d = \mathbf{g} - K_p(\dot{\mathbf{q}} - \dot{\mathbf{q}}_d) - K_I \int_0^t (\dot{\mathbf{q}} - \dot{\mathbf{q}}_d) d\tau, \tag{16}$$

where K_p, K_I are positive gains and $\mathbf{g} = [0,0,9.81]^{\mathrm{T}} \ ms^{-2}$.

A low-level attitude controller runs on the UAV autopilot, relying on the ArduCopter firmware, controlling the vehicle's orientation and commanding the signals to the motors. Using the vehicle mass m, the desired acceleration vector α is turned into thrust, roll, and pitch commands for the attitude controller as:

$$T = \frac{m\alpha_d^{\mathrm{T}}\mathbf{e}_3}{\cos(\phi)\cos(\theta)} \quad, \phi_d = -\frac{m\alpha_d^{\mathrm{T}}\mathbf{e}_1}{T} \quad, \theta_d = \frac{m\alpha_d^{\mathrm{T}}\mathbf{e}_2}{T} \tag{17}$$

where $\mathbf{e}_i$ is a unit vector along axis i. For the purposes of this work, the commanded yaw angle ψ_d is always kept at zero.

The ArduCopter autopilot uses a cascade PID structure to transform the desired orientation commands into angular velocities and subsequently desired moments, which are combined with the desired thrust result into throttle level commands for the motors, which is the vector $\tau \in \mathbb{R}^4$ depicted into **Figure 1**. The structure of the low-level attitude controller is illustrated in **Figure 4**. The output of the motor mixer is PWM-values for the Electronic Speed Controllers of the UAV motors, controlling the speed of each motor. Current attitude Φ is available from the EKF, while current angular rate $\dot{\Phi}$ is provided directly by the gyroscope.

5. Simulation studies

To showcase the results of the algorithm, a warehouse-like Gazebo environment is used, depicted in **Figure 5** in which the objective is to move from the starting position

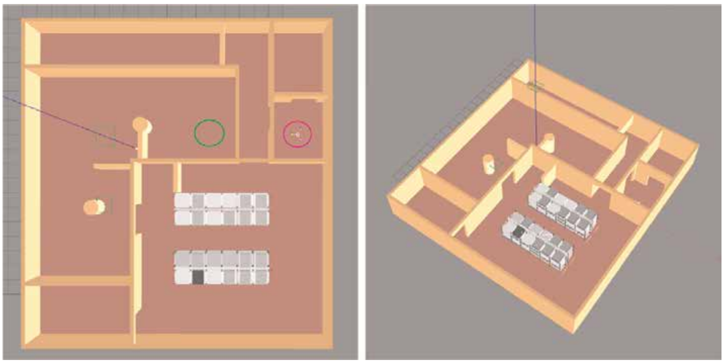


Figure 5.
Environment in two different perspectives.

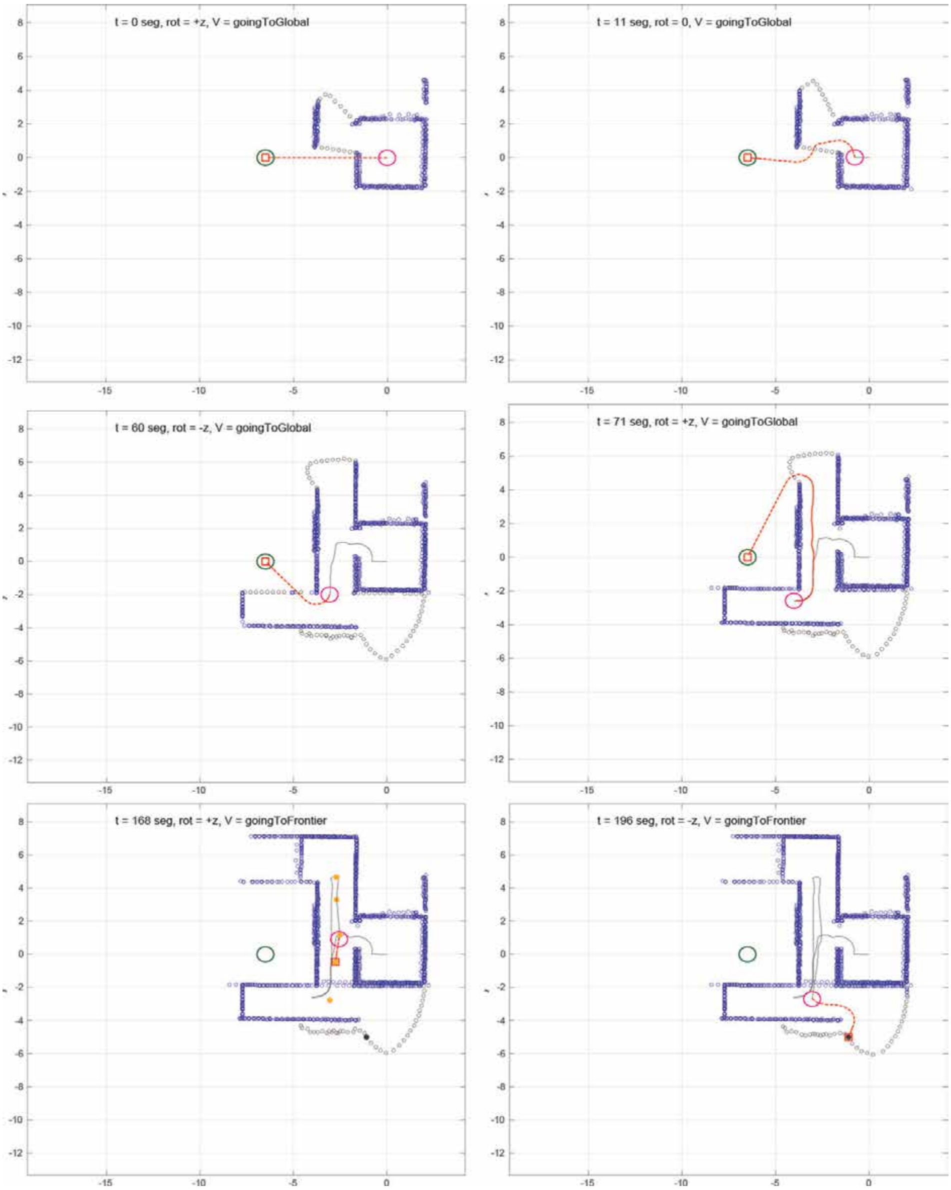


Figure 6.
Snapshots of the simulation.

(magenta circle to the right) to the end position (green circle to the left). The goal is to go from a starting position $\mathbf{p}_0 = [0 \;\; 0]^{\mathrm{T}} \;\; m$ to a global goal position $\mathbf{p}_{GLOBAL} = [-6.5{,}1.0]^{\mathrm{T}} \;\; m$, all written in a *world-frame* which is the starting body drone frame. The drone keeps a constant height value of 1.0 m. The parameters used were:

- *For the map generation*: Both FAST-LIO and OctoMap build their respective maps with a voxel grid size of 0.05 m. Furthermore, $d_{\max}$ that controls the range of updates on occupancy probabilities is set to 5.0 m.

- *For the global planner*: $h = 0.07\,m^2$, $R_{ROB} = 0.9\,m$, $K_C = 0.50\,s^{-1}$, $N_{TRY} = 5$, $f_A = 100\,s^{-1}$,$f_B = 5\,s^{-1}$,$f_C = 0.5\,s^{-1}$, $T_B = 2\,s$, $T_C = 240\,s$, $\delta = 0.13\,m$. Furthermore, $\alpha(E)$ was used such that $-A_{POS}\sqrt{E}$ when $E \geq 0$ and $A_{NEG}\sqrt{-E}$ when $E < 0$ with $A_{POS} = 0.05\,s^{-1}m^{-0.5}$ and $A_{NEG} = 0.60\,s^{-1}m^{-0.5}$. Also, the function $\beta(E) = v_{CIRC}\left(1 - E/D^2_{CIRC}\right)$ was employed with $v_{CIRC} = 0.20\,ms^{-1}$ and $D_{CIRC} = 0.32\,m$.

- *For the UAV controllers*: For the first stage PID (angle) the ArduCopter has $P = (11.6{,}12.0{,}5.4)\,s^{-1}$ for roll, pitch, and yaw respectively with 0 integral and derivative gains. For the second stage (angular rate) the gains were $P = (0.089{,}0.13{,}0.88)\,s^{-1}$, $I = (0.08{,}0.078{,}0.088)\,s^{-2}$, $D = (0.0024{,}0.004{,}0)$ (dimensionless) respectively. For the velocity controller, the gains used were $K_p = 0.5\,s^{-1}$, $K_I = 0.05\,s^{-2}$. The mass of the UAV is $m = 3.5\,kg$.

Figures 6 and 7 show some snapshots of the algorithm at different times. **Figure** 7 also shows the graph $\mathcal{G}$ at the end of the algorithm. The drone's current position is represented as a magenta circle, the target position $\mathbf{p}_{GLOBAL}$ as a green circle, the current goal $\mathbf{p}_{GOAL}$ as a red square, the points discovered so far as blue dots, the frontier points as gray dots, the current planned path with a red dashed line, the traced path in black. When the algorithm is in `goingToFrontier` mode, the graph nodes in orange and the selected frontier point $\mathbf{p}_F$ is represented by a black dot. Each one of the snapshots is further described:

- **Figure 6 left-top**: at the start, the drone is at `gointToGlobal` mode and, given the current map information (blue points), plan to go using a simple path (red dashed path) towards the global target (green start). **Figure 6 right-top**: the point is still at the same mode. **Figure 6 left-middle**: The robot still has hope to achieve the target. **Figure 6 right-middle**: after realizing that it cannot go to the global target, the drone decides that the best plan is to try to go from above since so far it did not discover that it is impossible to go from there. It changes the circulation mode and goes to the top, still in the `gointToGlobal` mode. **Figure 6 left-bottom**: after reaching the top, it realizes that it is blocked there as well: it is a dead-end. It decides to explore a frontier point (black dot) and enter the `goingToFrontier` mode to approach this point. **Figure 6 right-bottom**: after it travels all the nodes, it plans to go to the frontier point.

- **Figure** 7 **left-top**: the drone reaches the frontier point and switches back to the `gointToGlobal`, hoping to achieve the target given its current information of the map. **Figure** 7 **right-top**: eventually it realizes that it is not possible using the planned path. It plans to explore again. **Figure 6 left-middle**: it finds a point to explore and enters the `goingToFrontier` mode. Note that there are no orange points this time: the list of the nodes to be traveled only contains the frontier point $\mathbf{p}_F$, since the robot was already close to it. **Figure 6 right-middle**: it goes back to `gointToGlobal`, hoping to achieve the target given its current information on the map. **Figure 6 left-bottom**: the robot completed the mission successfully. **Figure 6 right-bottom**: graph structure $\mathcal{G}$ on the map.

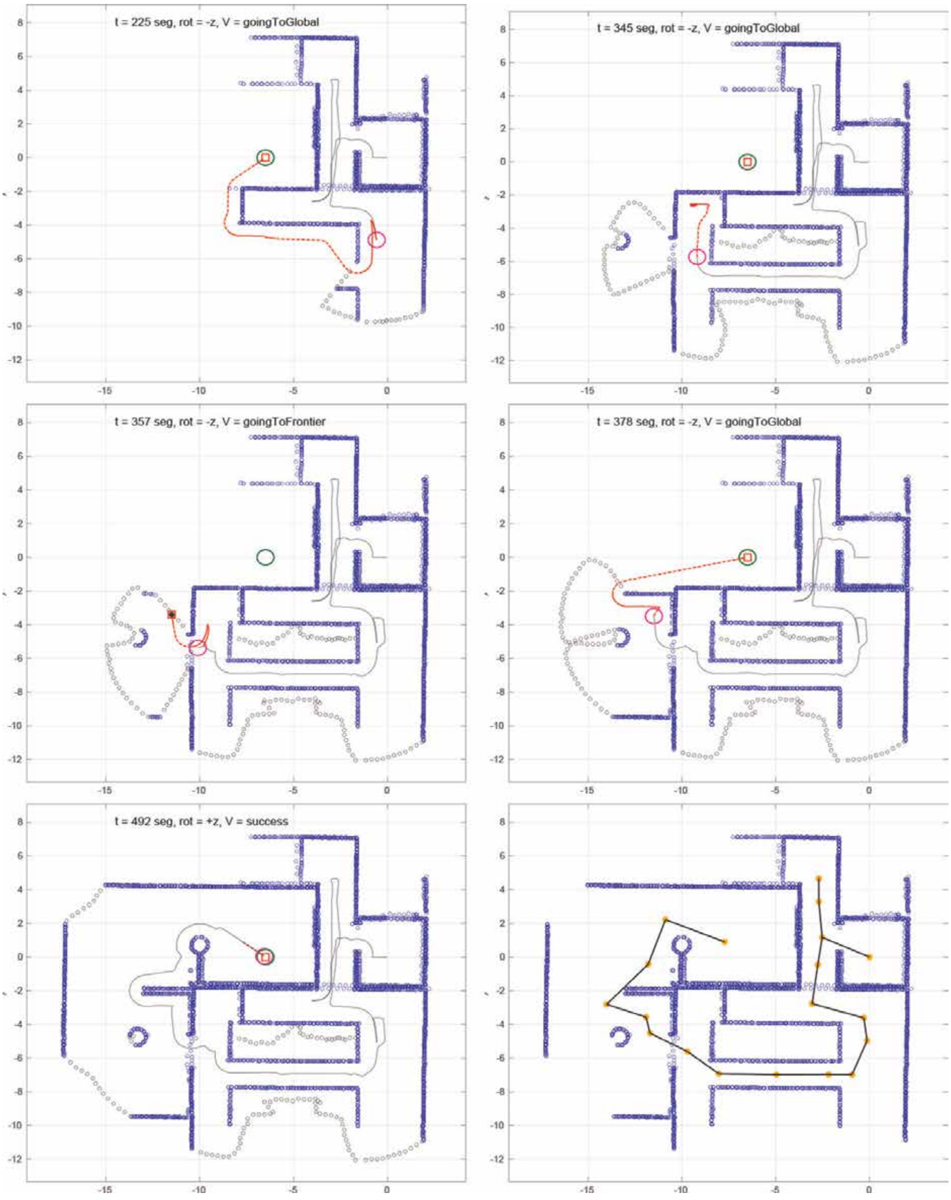


Figure 7.
Snapshots of the simulation and of the graph structure $\mathcal{G}$.

6. Conclusion

In this chapter, a framework is provided for the autonomous navigation of a drone in a GNSS-denied and unknown environment, aiming to reach a target point while avoiding obstacles and being equipped with only a LiDAR and IMU. The 2D-navigation algorithm is divided into three modules, that were discussed in detail. We showcased how the algorithm works through a Gazebo simulation physics engine, in which the simulated drone successfully navigated safely towards the goal.

Acknowledgements

This work was partially supported by the NYUAD Center for Artificial Intelligence and Robotics (CAIR), funded by Tamkeen under the NYUAD Research Institute Award CG010.

Author details

Halil Utku Unlu[1†], Dimitris Chaikalis[1†], Vinicius Gonçalves[2†] and Anthony Tzes[2,3*]

1 New York University, Electrical and Computer Engineering, Brooklyn, NY, USA

2 New York University Abu Dhabi (NYUAD), Center for Artificial Intelligence and Robotics, UAE

3 NYUAD, Electrical Engineering, Abu Dhabi, UAE

*Address all correspondence to: anthony.tzes@nyu.edu

† These authors contributed equally.

References

[1] Arvanitakis I, Tzes A, Giannousakis K. Synergistic exploration and navigation of mobile robots under pose uncertainty in unknown environments. International Journal of Advanced Robotic Systems. 2018;**15**(1): 1729881417750785

[2] Ilyas M, Ali ME, Rehman N, Abbasi AR. Design, development & evaluation of a prototype tracked mobile robot for difficult terrain. Sir Syed University Research Journal of Engineering & Technology. 2013;**3**(1): 7-7

[3] Tzes M, Papatheodorou S, Tzes A. Visual area coverage by heterogeneous aerial agents under imprecise localization. IEEE Control Systems Letters. 2018;**2**(4):623-628

[4] Wang F, Wang K, Lai S, Phang SK, Chen BM, Lee TH. An efficient UAV navigation solution for confined but partially known indoor environments. In: 11th IEEE International Conference on Control & Automation (ICCA). Taichung, Taiwan: IEEE; 2014. pp. 1351-1356

[5] Matos-Carvalho JP, Santos R, Tomic S, Beko M. GTRS-based algorithm for UAV navigation in indoor environments employing range measurements and odometry. IEEE Access. 2021;**9**:89120-89132

[6] Evangeliou N, Chaikalis D, Tsoukalas A, Tzes A. Visual collaboration leader-follower UAV-formation for indoor exploration. Frontiers in Robotics and AI. 2022;**8**: 777535

[7] Papatheodorou S, Tzes A, Giannousakis K, Stergiopoulos Y. Distributed area coverage control with imprecise robot localization: Simulation and experimental studies. International Journal of Advanced Robotic Systems. 2018;**15**(5):1729881418797494

[8] Aqel MO, Marhaban MH, Saripan MI, Ismail NB. Review of visual odometry: Types, approaches, challenges, and applications. Springerplus. 2016;**5**:1-26

[9] Huang G. Visual-inertial navigation: A concise review. In: 2019 International Conference on Robotics and Automation (ICRA). Montreal, Canada: IEEE; 2019. pp. 9572-9582

[10] Zeng B, Song C, Jun C, Kang Y. DFPC-SLAM: A dynamic feature point constraints-based SLAM using stereo vision for dynamic environment. Guidance, Navigation and Control. 2023; **3**(01):2350003

[11] Wang H, Wang Z, Liu Q, Gao Y. Multi-features visual odometry for indoor mapping of UAV. In: 2020 3rd International Conference on Unmanned Systems (ICUS). Harbin, China: IEEE; 2020. pp. 203-208

[12] Steenbeek A, Nex F. CNN-based dense monocular visual SLAM for real-time UAV exploration in emergency conditions. Drones. 2022;**6**(3):79

[13] Moura A, Antunes J, Dias A, Martins A, Almeida J. Graph-SLAM approach for indoor UAV localization in warehouse logistics applications. In: 2021 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC). Santa Maria, Portugal: IEEE; 2021. pp. 4-11

[14] Campos C, Elvira R, Rodrguez JJG, Montiel JM, Tardós JD. ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap slam. IEEE

Transactions on Robotics. 2021;**37**(6): 1874-1890

[15] Xin C, Wu G, Zhang C, Chen K, Wang J, Wang X. Research on indoor navigation system of UAV based on lidar. In: 2020 12th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA). Phucket, Thailand: IEEE; 2020. pp. 763-766

[16] Santos MC, Santana LV, Brandao AS, Sarcinelli-Filho M. UAV obstacle avoidance using RGB-D system. 2015 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE; 2015. pp. 312–319

[17] Kostavelis I, Gasteratos A. Semantic mapping for mobile robotics tasks: A survey. Robotics and Autonomous Systems. 2015;**66**:86-103

[18] Cadena C, Carlone L, Carrillo H, Latif Y, Scaramuzza D, Neira J, et al. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. IEEE Transactions on Robotics. 2016;**32**(6): 1309-1332

[19] Ali ZA, Zhangang H, Hang WB. Cooperative path planning of multiple UAVs by using max–min ant colony optimization along with cauchy mutant operator. Fluctuation and Noise Letters. 2021;**20**(01):2150002

[20] LaValle SM, Kuffner Jr JJ. Randomized kinodynamic planning. The International Journal of Robotics Research. 2001;**20**(5):378-400

[21] Kavraki LE, Svestka P, Latombe JC, Overmars MH. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions on Robotics and Automation. 1996; **12**(4):566-580

[22] Aggarwal S, Kumar N. Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges. Computer Communications. 2020;**149**: 270-299

[23] Yang L, Qi J, Song D, Xiao J, Han J, Xia Y. Survey of robot 3D path planning algorithms. Journal of Control Science and Engineering. 2016;**2016**:1-22

[24] Maini P, Sujit P. Path planning for a UAV with kinematic constraints in the presence of polygonal obstacles. In: 2016 International Conference on Unmanned Aircraft Systems (ICUAS). Arlington, VA, USA: IEEE; 2016. pp. 62-67

[25] Delamer JA, Watanabe Y, Chanel CP. Safe path planning for UAV urban operation under GNSS signal occlusion risk. Robotics and Autonomous Systems. 2021;**142**:103800

[26] Padhy RP, Verma S, Ahmad S, Choudhury SK, Sa PK. Deep neural network for autonomous navigation in indoor corridor environments. Procedia Computer Science. 2018;**133**:643-650

[27] Walker O, Vanegas F, Gonzalez F, Koenig S. A deep reinforcement learning framework for UAV navigation in indoor environments. In: 2019 IEEE Aerospace Conference. IEEE; 2019. pp. 1–14

[28] Ames AD, Coogan S, Egerstedt M, Notomista G, Sreenath K, Tabuada P. Control barrier functions: Theory and applications. In: 2019 18th European Control Conference. 2019. pp. 3420-3431

[29] Gonçalves VM, Krishnamurthy P, Tzes A, Khorrami F. Avoiding undesirable equilibria in control barrier function approaches for multi-robot planar systems. In: 2023 31st Mediterranean Conference on Control and Automation (MED). Limassol, Cyprus: IEEE; 2023. pp. 376-381

[30] Rezende AMC, Goncalves VM, Pimenta LCA. Constructive time-varying vector fields for robot navigation. IEEE Transactions on Robotics. 2022;**38**(2):852-867

[31] Chaikalis D, Evangeliou N, Nabeel M, Giakoumidis N, Tzes A. Mechatronic design and control of a hybrid ground-air-water autonomous vehicle. In: 2023 International Conference on Unmanned Aircraft Systems (ICUAS). 2023. pp. 1337-1342

[32] Xu W, Zhang F. Fast-LIO: A fast, robust lidar-inertial odometry package by tightly-coupled iterated Kalman filter. IEEE Robotics and Automation Letters. 2021;**6**(2):3317-3324

[33] Unlu HU, Chaikalis D, Tsoukalas A, Tzes A. UAV indoor exploration for fire-target detection and extinguishing. Journal of Intelligent & Robotic Systems. 2023;**108**(3):54

[34] Hornung A, Wurm KM, Bennewitz M, Stachniss C, Burgard W. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. Autonomous Robots. 2013;**34**:189-206

[35] Labbé M, Michaud F. RTAB-Map as an open-source LiDAR and visual simultaneous localization and mapping library for large-scale and long-term online operation. Journal of Field Robotics. 2019;**36**(2):416-446

[36] Kim G, Kim A. Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Madrid, Spain: IEEE; 2018. pp. 4802-4809

[37] Gao B, Pavel L. On the properties of the Softmax function with application in game theory and reinforcement learning. ArXiv:1704.00805. 2017

Chapter 7

Optimal Trajectory Tracking and Fuzzy-PID Controller Design for Nonlinear Gantry Crane

G. Kassahun Berisha, Hafte Tkue and Yalemzerf Getnet

Abstract

Gantry cranes are widely used in industry for load transfer, but increasing the trolley or rail speed to move loads quickly leads to excessive swaying. This swaying must be minimized before any loading or unloading can occur, a process that consumes time and can stress the hoist, damage surrounding objects, and jeopardize the payload. In the context of gantry crane control, the first step is to develop an optimal motion planner to generate a reference command. Then, a Fuzzy-PID controller is employed. This controller is designed to adapt to real-time system changes, making it well-suited for the gantry crane's nonlinear model, which conventional PID controllers struggle with. The Fuzzy-PID controller's nonlinear nature is particularly advantageous for managing the highly nonlinear gantry crane model. It excels, especially in the presence of external disturbances, outperforming traditional PID controllers. The fuzzy-PID controller is superior in optimizing sway angle tracking and achieving better position control compared to conventional PID controllers, offering enhanced performance in gantry crane operations.

Keywords: gantry crane, fuzzy logic controller, anti-sway, optimal motion planning, fuzzy-PID controller

1. Introduction

The most common cranes are bridge (overhead) cranes, gantry cranes, boom cranes, and tower cranes. Transporting the load as fast as possible without causing any excessive swing at its final destination/position is the main purpose of controlling a gantry crane. However, many of the gantry cranes results in a swing motion when there is a payload they stopped suddenly for a fast motion of the gantry crane [1].

The swing angle (motion) can be scheduled and the reduced to minimum but it becomes time consuming. Furthermore, the gantry crane operation needs a skillful operator to control the sway motion manually to stop the swing at the predetermined position of the gantry crane immediately [2]. Different kinds and degrees of accident in human beings and the surrounding will occur if the control mechanism of a gantry crane is poor [3]. Therefore, the performance of this equipment may be limited by the fact that the loads move in a pendulum-like manner, which is hazardous to industrial security [4].

The main objective of this paper are to design fuzzy-PID Controller for position and anti-sway control of the gantry crane and planning the motion that a gantry crane followed through its operation.

This system has great application in almost all industries many scholars have been studying in this area. Among them a some of articles are reviewed as follows.

In [5], modeling and nonlinear control of gantry crane using feedback linearization method is presented. In this paper, a dynamic model of gantry crane is developed using a powerful method called a Lagrange method. The State feedback linearization technique is applied for the model linearization and its weaknesses are reviewed. To solve these problems, the state feedback gain matrix is calculated using the Linear Quadratic Regulator method and results are fully investigated. Nonetheless, in the linearized model he used, it is difficult to investigate the nonlinear characteristics of the gantry crane. And also, the researcher does not consider the motion planning of the gantry crane. From a system response of the gantry crane, the settling time is long and the rise time is slow. One can notice it is not good enough and requires further demonstration.

In [6], position regulation and sway control of a nonlinear gantry crane system had proposed. With this paper, the controller used is Proportional plus derivative controller (PD) for both the gantry crane's position and sway angle. Having said that, the controller gains K_P and K_D are manually tuned. Also, the motion planning is not studied. The system response in terms of its overshoot is high and rise time is also slower.

In [7], optimal PID controller of automatic gantry crane control using particle swarm optimization (PSO) is presented. The controller used is PID and the particle swarm optimization (PSO) is used to auto tune the PID gains [8]. The good features of this paper are the PID controllers K_P, K_D, and K_I are tuned automatically using the PSO algorithm. The drawbacks of this paper are the motion trajectory is unknown which means not studied and the system response is poor. Also, the system response is not robust.

In [9], fuzzy control scheme for the gantry crane position and load swing control have proposed. The researchers proposed a fuzzy controller for positioning and anti-swing control scheme of the gantry crane. The linearized gantry crane system model which does not represent the nonlinear nature of the crane system dynamics and the fact that the linearized mathematical model mostly does not provide the dynamic behavior of the real system well. The system responses are satisfactory but the consideration of cart optimal motion planning is not included.

In [10], the model predictive control of gantry crane with input nonlinearity compensation studied. The paper evaporates the application of a nonlinear model predictive control method for controlling gantry crane systems. The drawbacks of this paper are that the system's mathematical model is linear and the motion plan trajectory is not considered. The response of the gantry crane to its sway angle is not settling for a long time to its equilibrium points and the steady state is also not minimum.

In [11], the paper titled pole placement control of a 2D gantry crane system with varying pole locations is considered. In this paper, the model of the system linearized model, and the controller used is a pole placement controller which is designed with three poles located at different locations on the left-hand side of the s-plane. Due to highly nonlinearity of gantry crane, controlling the position and sway using manual method of varying location of poles is not advisable criteria. Also, the model does not specify the real worlds specification due to many assumptions while linearizing the model. Again, the motion planning trajectory is not considered. The system responses using pole placement method showed that a higher overshoot and longer time to settle. Therefore, nonlinear controllers are the best option for nonlinear systems.

In [4], the closed-loop schemes for position and sway control of a gantry crane system is studied. This paper the gantry crane control method presented are the performance of linear quadratic regulator (LQR) and proportional plus integral plus derivative control schemes for both payload sway control and trolley position tracking. In an approximate model in which many system parameters are assumed helps the researcher from considering the nonlinearities and removes system modeling complexity but the response is not as acceptable as the true model. Not only this, but also the motion planning or trajectory planning is not considered. Again the system response is not adequate, which means the rise time is slow and higher overshoot/undershoot is detected but the steady state error is very good.

1.1 Problem of statement

Gantry crane is a common type of crane that used to transfer the payload from one position to another desired position. During the load the gantry crane behaves like a pendulum during the transportation that process sway around the destination position. If the sway reaches a final critical limit, the crane process must be stopped until the sway disappears otherwise the situation will lead to accident, injury, or fatality. Crane operation in general requires quick transferring of the load to the desired position without excessive swaying.

Unfortunately, fast transporting heavy loads would result in the excessive swaying of the load. Moreover, the swaying of the load can also be affected by external environmental disturbances like wind and parameter uncertainty in the system. Because of uncontrolled payload swing dynamics, it is necessary to have an anti-sway controller installed in a gantry crane system to suppress the swaying without human interaction. To optimize between fast transporting and less swaying, optimal motion tracking algorithms are a great tool to use.

1.2 Proposed solution

Using optimal motion tracking, a desired velocity profile trajectory that should be tracked by cart/bridge will be produced. Different types of controllers can be used to better track the reference velocity profile signal by the cart or the bridge. The highly nonlinear crane dynamics are bettered controlled with a nonlinear control scheme. Hence, Fuzzy-PID controller is proposed for this system since it has nonlinear nature, and is highly robust to parameter variation, and exploits the benefit of both Fuzzy and PID controllers.

1.3 Motivation and contribution

Since cranes are under actuated oscillatory systems. Many types of research have been done in the area of controlling gantry cranes, aiming at the reduction of positioning errors and traveling time. Since they do not need feedback sensors for measuring the states of the crane, Open-loop strategies are the most used by far for their cheapness. Due to the parameter uncertainties and disturbances we propose an optimal motion planner is an open-loop algorithm that will produce an ideal trajectory that should be tracked by the trolley/bridge of the gantry crane for best performance. This makes optimal motion planning and Fuzzy-PID controller complement each other in solving the problems in the gantry crane. Moreover, the highly nonlinear model of gantry crane is better controlled with a nonlinear controller like Fuzzy-PID controller.

1.4 Paper organization

Section 2 presents gantry crane modeling, which will describe the basic operation of gantry crane, structure of gantry crane, and mathematical model of gantry crane. Section 3 describes the design of the conventional controller, PI and the fuzzy controller, and fuzzy-PID controller. In Section 4 shows the simulation results result and also Section 5 compates the discussion on the proposed controller and compare them with the conventional controller. Finally, the performance of each controller is measured. In Section 6 draws the conclusions on the paper and also in Section 7 recommendation to further works is presented.

2. Mathematical modeling

Since the gantry crane has two movable parts which are the rail and the trolley (cart), its motion dynamics can be mathematically modeled in two separate parts. A trolley is an electromechanical machine installed on the rail that traverses long Y-axis shown in **Figure 1**. The trolley carries a load attached to the hoist along the rail. The rail carries the trolley and moves along the X-axis. With the movement of the rail and the cart, we can reach any point in the x-y plane inside the working region of the crane [12].

2.1 Mathematical model of the trolley/cart movement

When translational force is exerted on the cart by a motor, the cart moves along the Y-axis and this creates swing dynamics of the load along the Y-axis [13]. This swaying of the load along the Y- axis is measured as an angle which is denoted by in

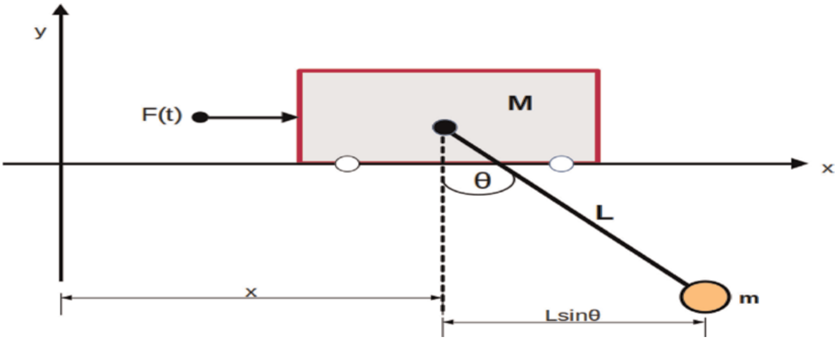


Figure 1.
Gantry crane free body diagram.

Figure 1. While modeling the cart movement, we have to describe the dynamics of the cart position denoted by y and the load swaying angle (θ). The Lagrangian method is used here since it eases the complexities of the steps for gantry crane [12]. Lagrangian function L is described as:

$$L = T - V \tag{1}$$

Where T and V are the Kinetic and potential energy of the entire system. The Lagrangian equation is given by:

$$\frac{d}{dt}\left[\frac{dL}{d\dot{u}_i}\right] - \frac{dL}{d\dot{u}_i} = F_i \tag{2}$$

Where $u_i = [x, \theta]^{\mathrm{T}}$ is generalized displacements and F_i is an external force. The kinetic energy of the gantry crane is

$$T = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}m\dot{x}_p{}^2 \tag{3}$$

The potential energy of the gantry crane is

$$V = mg\dot{y}_p \tag{4}$$

Plugging into fundamental langrage equation to get equation of motion in x-component of gantry crane gives

$$(M + m)\ddot{x} + mL\ddot{\theta}\cos\theta - mL\dot{\theta}^2\sin\theta = F \tag{5}$$

For the rotational component of the gantry crane the Lagrange equation gives

$$m\ddot{x}\cos\theta + mL\ddot{\theta} + mg\sin\theta = 0 \tag{6}$$

Now, by applying small angle approximation of $sin\theta{\approx}\theta$, $\cos\theta{\approx}1$, the equation of motion for the x-component in Eq. (7) and rotational motion in Eq. (8) are expressed.

$$\ddot{x} = \frac{mg\theta + mL\dot{\theta}^2\theta + F}{m + M\theta} \tag{7}$$

$$\ddot{\theta} = -\frac{\ddot{x} + g\theta}{L} \tag{8}$$

2.2 Mathematical model of the rail movement

When a force acts on the stationary rail, the rail moves along the x-axis as shown in the coordinates in the figure and this causes the load to sway along the x-axis. This swaying can be measured as an angle as shown in **Figure 1**. To describe the mathematical model of the rail movement, we have driven the dynamics of the position of the rail denoted by X and the load swaying caused by it denoted by θ. Other than that, we will follow the same fashion as into driving the dynamics of the rail movement.

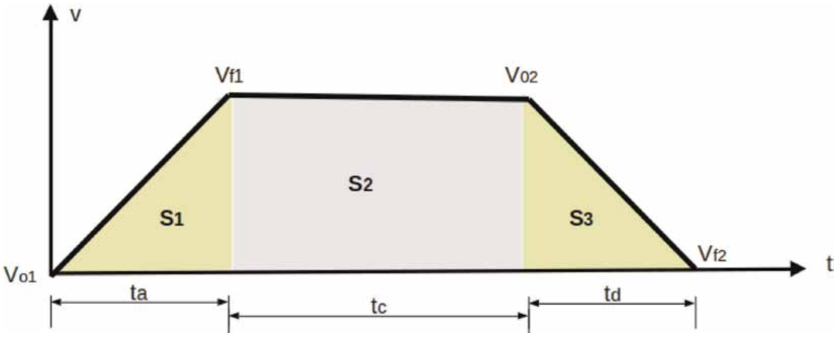


Figure 2.
Trapezoidal velocity plan.

2.3 Optimal motion planning

In this paper, trapezoidal velocity-time curve shown **Figure 2** is used as reference of the position and sway angle of the gantry crane. For the designed optimal motion planning the sway angle at the destination can be lowered to zero.

2.3.1 Trajectory planning

For the 2D gantry crane system the reference signal can be used many types of test signals but in order for the accuracy and safety of operation it is better to plan the trajectory of the reference to track the output of the system. In this paper, trapezoidal velocity – time curve shown in **Figure 2** is used as reference of the gantry system control system. But, the entire shapes of loads to be transported to and/or from are not the same in the practical application. For simplicity it is assumed that acceleration time t_a is equal to the deceleration time t_d in the areas S_1 and S_3 respectively and the total displacement is expressed in Eq. (9):

$$S = S_1 + S_2 + S_3 \tag{9}$$

Where S_1, S_2, S_3 are the area of acceleration, constant speed, and deceleration time zones. At the initial state all parameters are assumed at equilibrium. i.e. sway angle $\theta = 0$, velocity $V_i = 0$, acceleration $a_i = 0$, and applied force $F_i = 0$.

3. Controller design

3.1 PID controller

The classical PID controller is the well known form of feedback controller and became the standard tool when process control emerged in the 1940s [14]. In process control today, more than 95% of the control loops are of PID type, most loops are actually PI control. PID controllers are today found in all areas where control is used [15]. The PID controller attempts to minimize the error value over time by adjustment of a control variable to a new value determined by a weighted sum [4].

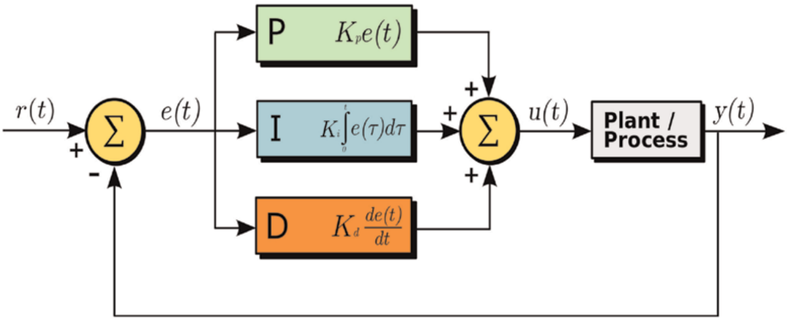


Figure 3.
Structure of a PID controller.

$$u(t) = K_p e(t) + K_I \int_0^{\tau} e(\tau)d\tau + K_D \frac{d}{dt} e(t) \tag{10}$$

Where K_P, K_I, and K_D all non-negative, denote the coefficients for the proportional controller, integral controller and derivative controller terms, respectively **Figure 3**.

3.2 Fuzzy logic controller

Fuzzy control provides a formal methodology for representing, manipulating, and implementing a human's heuristic knowledge about how to control a system. In fuzzy logic, basic control is determined by a set of linguistic rules which are determined by the system. In fuzzy logic control method the mathematical modeling of the system is not required. The input parameters are fuzzified in the fuzification module and represented in fuzzy set notations as membership functions and "IF⋯THEN⋯" rules produce output signal and these signals are deffuzzified to crisp values in the diffuzification module [16].

3.2.1 Fuzzification

The Fuzzification module transforms the physical values of the input reference signal into a normalized fuzzy subset consisting of a subset for the range of the input values and an associate membership function describing the degrees of the input belonging to the given ranges.

The error and rate of change of error are defined by linguistic variables such as negative big (NB), negative medium (NM), negative small (NS), zero (Z), positive small (PS), positive medium (PM), and positive big (PB) characterized by triangular membership functions shown in **Figures 4** and **5** respectively. The output is also defined by seven linguistic variables characterized by membership functions given in **Figure 6**.

3.2.2 Decision making

Mamdani fuzzy inference system is well-suited to human input, intuitive, and can easily obtain the relationship between its inputs and output. In this paper, triangular membership function is used because of simplicity. The "IF⋯THEN⋯" principles characterize a fuzzy inference system (FIS) by associating the output to the inputs (**Figure** 7) [17].

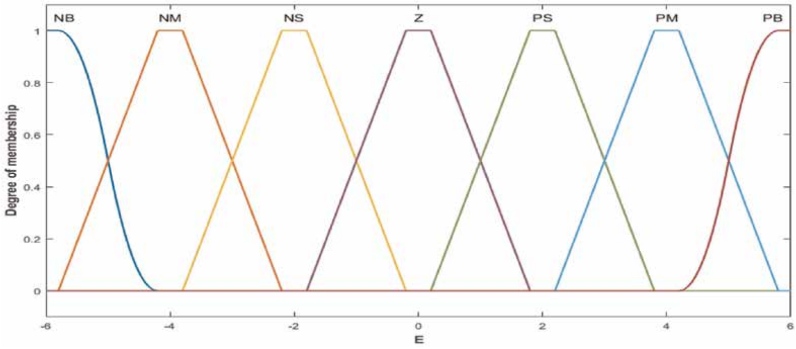


Figure 4.
Error membership function.

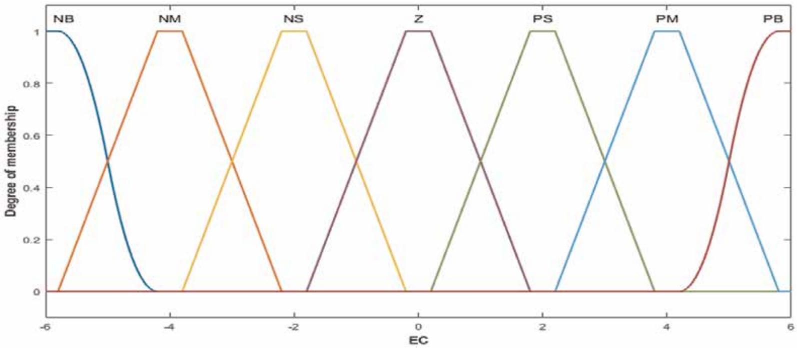


Figure 5.
Derivative of error membership function.

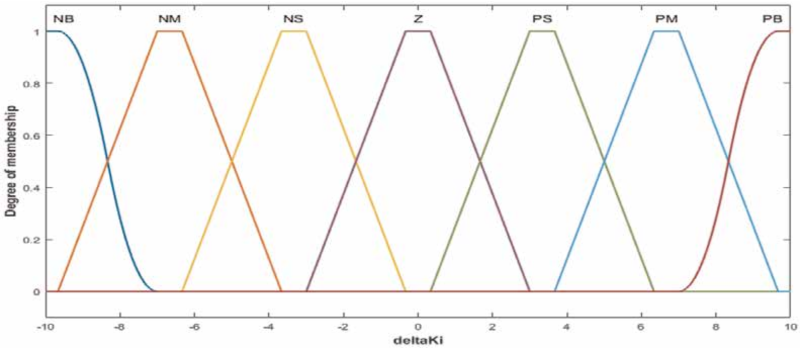


Figure 6.
Output membership function.

The output is produced by fuzzy sets and fuzzy logic operations by evaluating all the rules, **Table 1** which have 49 rules for the seven (7) linguistic variables and the combination of rules are expressed as follows. Where E represents for error and Δ E for change in error.

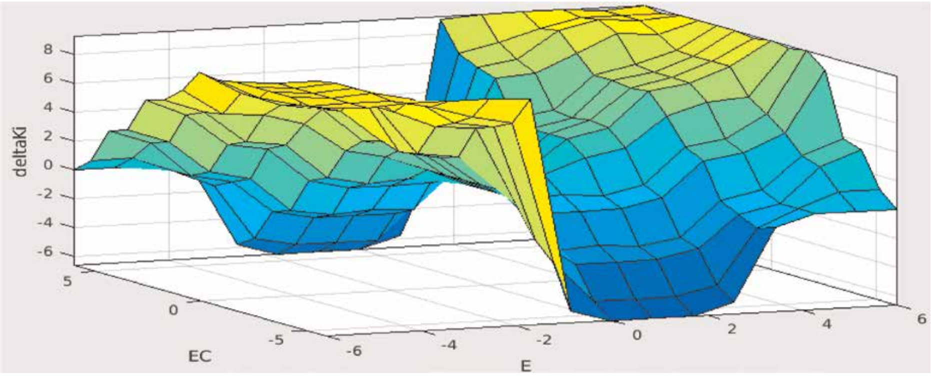


Figure 7.
Surface of membership function.

					E			
		NB	**NM**	**NS**	**Z**	**PS**	**PM**	**PB**
	NB	PB	PB	PB	NM	NM	Z	Z
	NM	PB	PB	PM	NS	NS	Z	PS
	NS	PB	PB	PS	Z	Z	PS	PB
ΔE	Z	PB	PM	PS	Z	PS	PM	Z
	PS	PM	PS	Z	Z	PS	PM	PB
	PM	PS	Z	NS	NS	PM	PB	PB
	PB	Z	Z	NM	NM	PB	PB	PB

Table 1.
Rule base of the fuzzy controller.

3.2.3 Defuzzification

Deffuzification is the process of converting the controller outputs in linguistic labels represented by fuzzy set to crisp values. The final crisp output value from the fuzzy logic controller depends on the type of Deffuzification method used to compute the outcome values corresponding to each label. The input for the Deffuzification process is the aggregated output fuzzy set from the rule base of the fuzzy logic controller and the output is a single crisp number [18].

The rule base outcomes of the fuzzy logic inference mechanism after they have been logically added and then compute a value that will be the final output of the fuzzy controller are examined by the Deffuzification module. The fuzzy controller then sends control signal to the output module to control the plant. Thus, during Deffuzification, the module converts the fuzzy output signals into a real life crisp value [19].

3.3 Fuzzy-PID controller

Fuzzy logic controller is one of the nonlinear controllers that can be used to control gantry cranes. In comparison to the classical PID controller, fuzzy logic controller is a

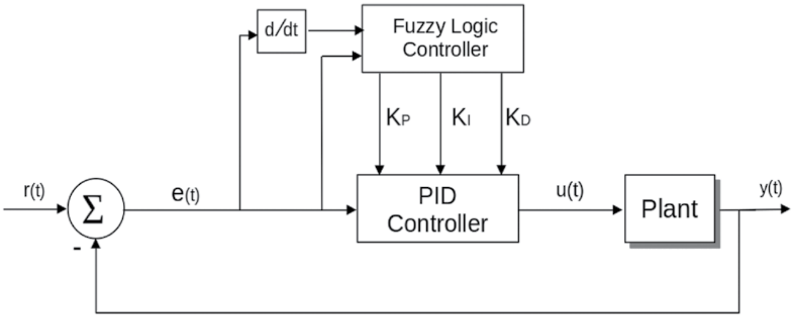


Figure 8.
Fuzzy-PID controller.

non-linear controller that can provide good performance under the parameters uncertainties and disturbances. The fuzzy logic controller is preferable for systems having cumbersome mathematical modeling and calculations [20]. In Fuzzy-PID controller, the error and rate of change of error signals are passed through the PID controller. These signals are processed by the fuzzy logic controller and their magnitudes are brought to a range desirable for input to the designed PID controller. Fuzzy-PID controller is the combination of conventional PID controller and fuzzy logic controller in which hybrid controllers are expected to improve the performance of nonlinear systems like gantry crane. Due to the nature of the gantry crane (which is non-linear) and the exact condition of the fault will happen to the system is random, the PID controller is not recommended alone. Therefore, good features of PID controller and fuzzy logic controller makes the system control strategy better (**Figure 8**).

4. Simulation results

In the simulation result of this paper, the optimal motion planning is planned using velocity trajectory path for the reference signal then the Fuzzy-PID controller is designed and finally the result is simulated as shown in the upcoming figures.

In this paper, the PID controller and Fuzzy-PID controller are simulated and compared with respect to the parameters of the gantry crane both sway angle and position. The optimal motion planning is used to track the output of the gantry crane same as the reference with minimized disturbances (**Figure 9**).

4.1 Simulation results using PID controller

For the designed optimal motion planned reference of the gantry crane the parameters of the PID controller (K_P, K_I, and, K_D) are tuned from the toolbox of the MATLAB.

The velocities of the cart and payload increases linearly until desired maximum velocity is reached. After attaining their respective maximum velocity they moves with constant velocity for a certain time, then it starts decreasing its velocity until it

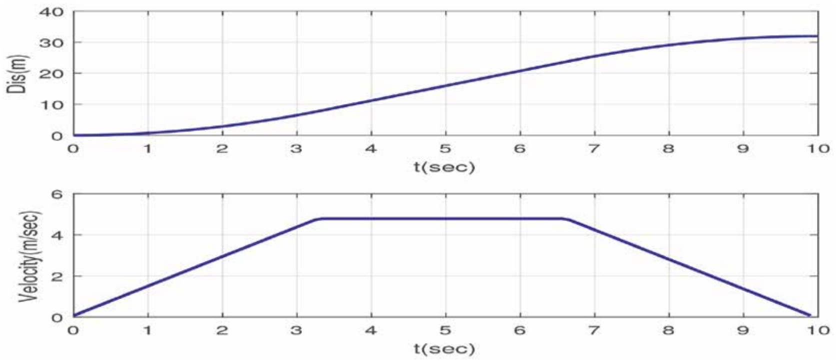


Figure 9.
The motion planned velocity trajectory path for reference signal.

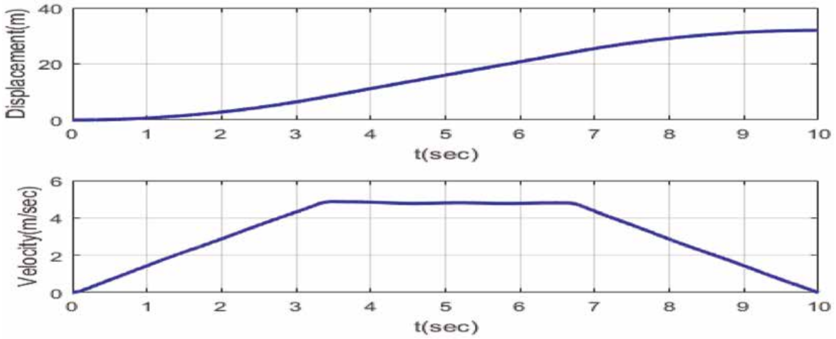


Figure 10.
The trajectory path tracking of gantry crane PID controller.

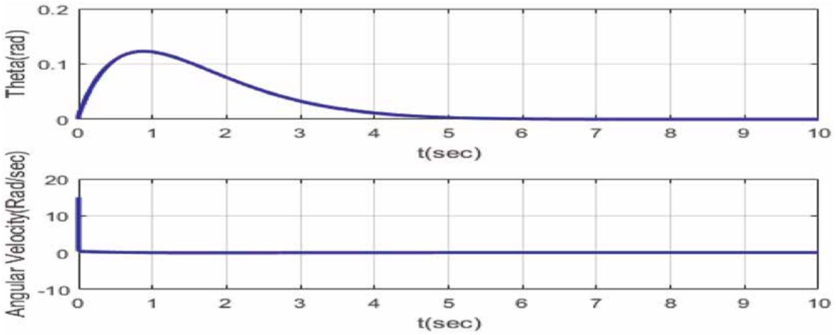


Figure 11.
The sway angle and angular velocity of the payload gantry crane using PID controller.

approach to zero as shown **Figure 10**. The designed optimal motion planning of a 2D gantry crane of reference velocity of the cart and the load looks like as **Figure 10** that tracks the trapezoidal curve shown in **Figure 9**. **Figure 11** shows that the settling time of sway angle of the gantry crane under the classical PID controller is a long time. Due to the nonlinearity features of the gantry crane, the crane will not arrive its predetermined time under classical PID controller.

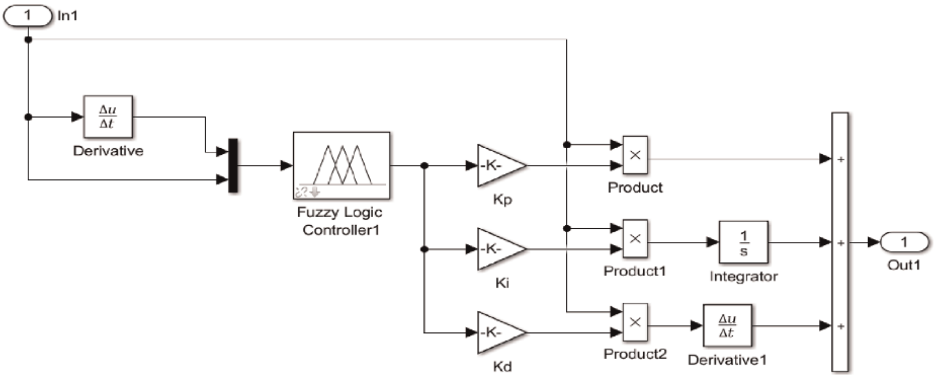


Figure 12.
Simulation diagram of fuzzy-PID controller.

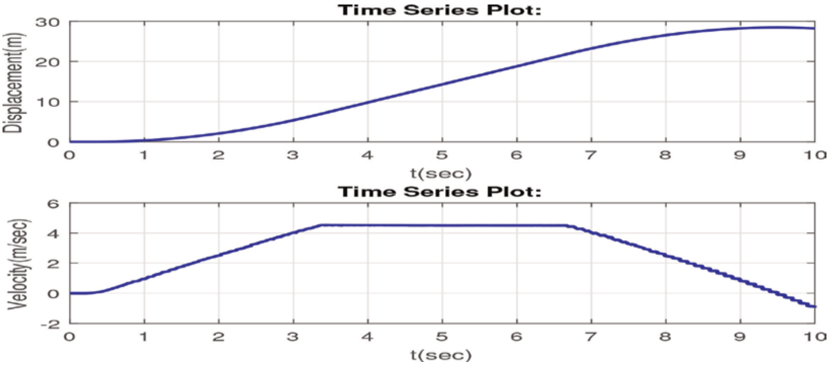


Figure 13.
The trajectory path tracking of gantry crane using fuzzy-PID controller.

4.2 Simulation results using fuzzy-PID controller

In Fuzzy-PID controller simulation, the values of the PID controller gains (K_P, K_I, and, K_D) are updates and evaluated from the rule base of the fuzzy logic controller. So, when any parameters are changing or disturbances are added into the system the fuzzy logic controller updates the values of the classical PID controllers (K_P, K_I, and, K_D). Therefore, classical PID controller alone is not efficient in the presence parameter variations and uncertainties. So, fuzzy-PID Controller can to handle parameter uncertainties and disturbances and the fuzzy-PID controller is depicted in **Figure 12**. From the simulation results, the Fuzzy-PID controller has very small time required to settle about the sway angle compared to PID controller. Therefore, fuzzy-PID controller have a better sway angle tracking performance as compared to classical PID controller shown above (**Figures 13** and **14**).

5. Discussion

From the simulation results, the position control of the gantry crane using PID controller and fuzzy-PID controller are compared. According to the simulation result

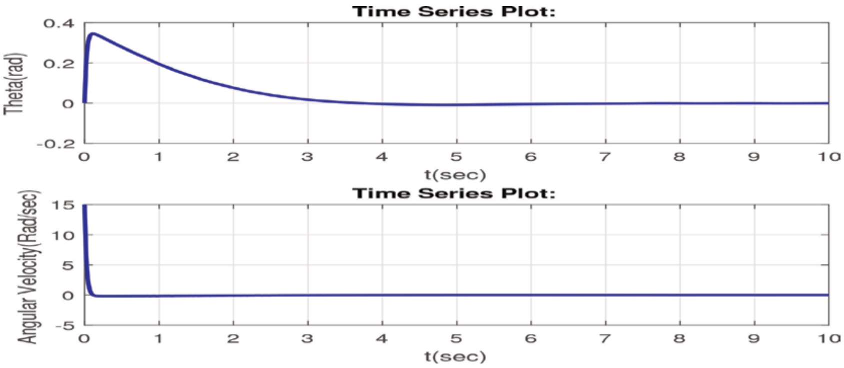


Figure 14.
The sway angle and angular velocity of the payload of gantry crane using fuzzy-PID controller.

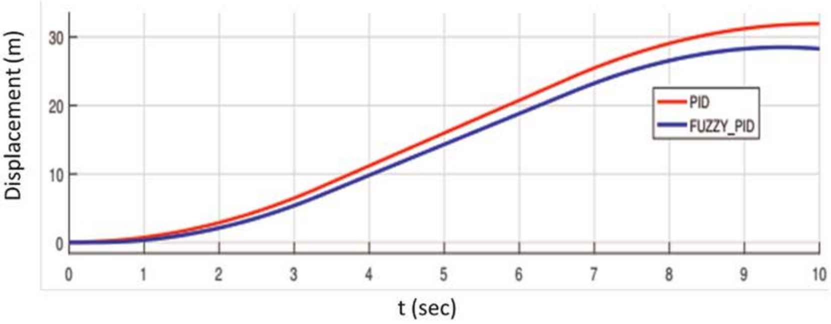


Figure 15.
Comparison of position of gantry crane.

the PID controller is better in position control of the gantry crane with respect to the path tracking of the gantry crane. According to the simulation result the both the PID controller and fuzzy-PID controller are almost similar in velocity control of the gantry crane with respect to the path tracking of the gantry crane. According to the simulation result the fuzzy-PID controller is much better than that of PID controller in angular velocity control of the gantry crane with respect to the path tracking of the gantry crane. Generally, for the control of position and velocity control of a gantry crane PID controller is better but, for the sway angle control and angular velocity control fuzzy-PID controller is better than that of classical PID controller (**Figures 15** and **16**).

6. Conclusion

In this paper, the modeling and simulation of gantry crane controlled by fuzzy-PID Controller have been developed. From the simulation results, during transportation of the payload time efficiency is improved and ensured the safety of the position and sway of the gantry crane. The planned motion of the references of the gantry crane are followed by the designed controllers with minimum time and with small sway angle of the 2D gantry crane. Finally, optimal reference velocity tracking performance is used to compare the performances of the classical PID controller and Fuzzy-PID controller.

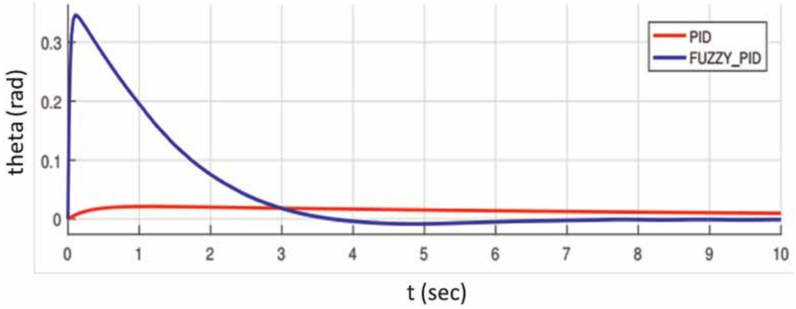


Figure 16.
Comparison of sway angle(theta) of gantry crane.

The simulation result of fuzzy-PID controller has smaller sway angle as compared to classical PID controllers. Therefore, fuzzy-PID logic controller has better performance on the position and sway angle control.

7. Future scope

In this paper, fuzzy-PID control systems and motion planning are proposed, designed, analyzed, and simulated using MATLAB. For the future work this paper can be done in different advanced controllers and other planning methods. Using intelligent control methods like artificial intelligence, reinforcement learning methods will be the best data training methods of gantry crane sway control and position controls. Since controlling the gantry crane with real data's will be good controlling mechanism. Not only the controller but also the dynamics of controllers also will be extended to three dimensional (3D) model of gantry crane.

Author details

G. Kassahun Berisha[1,2*†], Hafte Tkue[2†] and Yalemzerf Getnet[2]

1 Wolaita Sodo University, Wolaita Sodo, Ethiopia

2 Addis Ababa Institute of Technology, Addis Ababa, Ethiopia

*Address all correspondence to: kassahunberisha@gmail.com

† These authors contributed equally.

References

[1] Ahmad M, Nasir A, Najib M, Ishak H. Anti-sway techniques in feedback control loop of a gantry crane system a comparative assessment of pd and pd-type fuzzy logic controller. In: 2009 4th IEEE Conference on Industrial Electronics and Applications. X'ian, China: IEEE; 2009. pp. 2483-2487

[2] Ajayan M, Nishad PN. Vibration control of 3d gantry crane with precise positioning in two dimensions. In: 2014 Annual International Conference on Emerging Research Areas: Magnetics, Machines and Drives (AICERA/iCMMD). Kottayam, India; 2014. pp. 1-5

[3] Ahmad M, Zulkifely Z, Zawawi MA. Experimental investigations of input shaping schemes for sway control of a gantry crane system. In: 2010 Second International Conference on Computer and Network Technology. Bangkok, Thailand; 2010. pp. 483-486

[4] Alhassan A, Danapalasingam KA, Shehu M, Abdullahi A, Tijjani A. Closed-loop schemes for position and sway control of a gantry crane system. International Journal of Simulation Systems, Science and Technology. 2016;**17**:1

[5] Omidi E. Modeling and nonlinear control of gantry crane using feedback linearization method. arXiv Preprint. 2014;**5**

[6] Çakan A, Onen U. Position regulation and sway control of a nonlinear gantry crane system. International Journal of Scientific and Technology Research. 2016;**5**:121-124

[7] Solihin MI, Wahyudi W, Kamal MAS, Legowo A. Optimal pid controller tuning of automatic gantry crane using pso algorithm. In: 2008 5th International Symposium on Mechatronics and its Applications. Amman, Jordan; 2008. pp. 1-5

[8] Åström K, Hägglund T. Advanced PID Control. United States: ISA-The Instrumentation, Systems, and Automation Society; 2006

[9] Popadic T, Kolonic F, Poljugan A. A Fuzzy Control Scheme for the Gantry Crane Position and Load Swing Control. Vol. 6. University of Zagreb; 2006

[10] Su S, Nguyen H, Jarman R, Zhu J, Lowe D, McLean P, et al. Model predictive control of gantry crane with input nonlinearity compensation, world academy of science. Engineering and Technology. 2009;**3**:899-903

[11] Bashir NM, Bature AA, Abdullah AM. Pole placement control of a 2d gantry crane system with varying pole locations. Applications of Modelling and Simulation. 2018;**2**:8-16

[12] Park E, Martinez C. Motion Design Problem for a Gantry Crane. California State University Fullerton; 11 Jul 2021. pp. 4-6

[13] Zoke H. Portabel Gantry Crane. Xinxiang, Henan China: Hehan Zoke Crane CO., LTD; 2022

[14] Schleicher M, Blasinger F. Control Engineering: A Practical Guide. Measurement - Control - Recording. JUMO; 2006

[15] Nise S. Nise's Control Systems Engineering. United States: Wiley; 2010

[16] Guanrong CT. Introduction to Fuzzy Sets, Fuzzy Logic, and Fuzzy Control Systems. United States: CRC Press; 2000

[17] Ahmad AM, Mohamed Z. Hybrid fuzzy logic control with input shaping for input tracking and sway suppression of a gantry crane system. American Journal of Engineering and Applied Sciences. 2009;**2**:241-251

[18] Allam E, Elbab HF, Hady M, Abouel-Seoud S. Vibration control of active vehicle suspension system using fuzzy logic algorithm. Fuzzy Information and Engineering. 2010;**2**(4):361-387

[19] Zadeh L, Klir G, Yuan B. Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems: Selected Papers. World Scientific; 1996

[20] Simanjalam KR. Position and Anti-Sway Control of a Gantry Cranesystem Using Fuzzy-Tuned Pid Controller [Thesis]. Malaysia: University Teknologi Malaysia; Jun 2012

Edited by Zain Anwar Ali and Amber Israr

This book, *Motion Planning for Dynamic Agents*, presents a thorough overview of current advancements and provides insights into the fascinating and vital field of aeronautics. It focuses on modern research and development, with an emphasis on dynamic agents. The chapters address a wide range of complex capabilities, including formation control, guidance and navigation, control techniques, wide-space coverage for inspection and exploration, and the best pathfinding in unknown territory. This book is a valuable resource for scholars, practitioners, and amateurs alike due to the variety of perspectives that are included, which help readers gain a sophisticated understanding of the difficulties and developments in the area of study.

Published in London, UK

IntechOpen

ISBN 978-0-85466-060-5

9 780854 660605