Czech Technical University in Prague

Faculty of Electrical Engineering
Department of Computer Science
Multi-robot Systems

Multi-UAV Trajectory Planning in Unknown Environments with Limited Information Sharing

Master's Thesis

Afzal Ahmad

Prague, May 2022

Study programme: Open Informatics
Branch of study: Artificial Intelligence

Supervisor: Ing. Vojtěch Spurný, Ph.D.

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Ahmad Afzal**	Personal ID number:	**489246**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Computer Science**		
Study program:	**Open Informatics**		
Specialisation:	**Artificial Intelligence**		

II. Master’s thesis details

Master’s thesis title in English:

Multi-UAV trajectory planning in unknown environments with limited information sharing

Master’s thesis title in Czech:

Planování trajektorie pro skupinu helikoptér v neznamém prostředí s omezením na množství sdílených informací

Guidelines:

The aim of the thesis is to design and implement a method for collision-free trajectory planning for a multi-UAV system without any prior knowledge about the environment. The task is to navigate the UAVs to their respective goals while avoiding collisions with other UAVs and the environment. The planning method will be realized in a decentralized way with limited sharing of information among UAVs. The student will experiment with the amount of shared information necessary to achieve the task successfully.

Work plan:

- Review the existing literature for safe multi-UAV trajectory planning and identify their strengths and weaknesses when used under limited information (position, velocity, communication range, etc.),
- Based on the identified resources, design or extend an existing method to work with limited information.
- Integrate this method into the MRS system [1].
- Verify and test its functionality in Gazebo simulator under different environments.

Compare the proposed solution with available methods.

Bibliography / sources:

[1] T. Baca, M. Petrlik, M. Vrba, V. Spurny, R. Penicka, D. Hert, et al., “The MRS UAV System: Pushing the Frontiers of Reproducible Research, Real-world Deployment, and Education with Autonomous Unmanned Aerial Vehicles,” Journal of Intelligent & Robotic Systems, vol. 102, no. 26, pp. 1–28, 1 2021

[2] Şenbaşlar, Baskın, Wolfgang Hönig, and Nora Ayanian. "RLSS: Real-time Multi-Robot Trajectory Replanning using Linear Spatial Separations." arXiv preprint arXiv:2103.07588 (2021).

[3] Tordesillas, Jesus, and Jonathan P. How. "MADER: Trajectory planner in multiagent and dynamic environments." IEEE Transactions on Robotics (2021).

[4] Arul, Senthil Hariharan, and Dinesh Manocha. "Swarmcco: Probabilistic reactive collision avoidance for quadrotor swarms under uncertainty." IEEE Robotics and Automation Letters 6.2 (2021): 2437-2444.

[5] Wang, Mingyu, et al. "Game-theoretic planning for risk-aware interactive agents." 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020.

Name and workplace of master’s thesis supervisor:

Ing. Vojtěch Spurný, Ph.D. Multi-robot Systems FEL

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **28.01.2022** Deadline for master's thesis submission: **20.05.2022**

Assignment valid until: **30.09.2023**

Ing. Vojtěch Spurný, Ph.D. Supervisor’s signature	Head of department’s signature	prof. Mgr. Petr Páta, Ph.D. Dean’s signature

III. Assignment receipt

The student acknowledges that the master’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master’s thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student’s signature

Annotation

This thesis focuses on developing a method for avoiding collisions between multiple Unmanned Aerial Vehicles (UAVs) flying in a shared space. The aim to design and implement a decentralized technique for collision avoidance and verify it using simulated experiments. The thesis develops a probabilistic motion prediction method and uses a re-planning algorithm to fly collision-free trajectories around obstacles and other UAVs.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of the information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university thesis.

Prague, May, 2022

Acknowledgments

I am thankful to my thesis supervisor Ing. Vojtěch Spurný for his support and guidance. He allowed me to explore my area of interest and helped me in formulating the ideas into the form of this thesis. I am extremely thankful to Dr. Martin Saska for providing me the opportunity to work in the Multi Robot Systems (MRS) group. His constant encouragement and guidance have been instrumental in my growth. I also want to thank Tomáš Báča for several insightful discussions and feedback. He has taught me how to think critically and formulate my ideas in a structured form.

The masters program has been a demanding journey for me. During this time my friends and lab mates have been incredibly supportive and have helped me get though it. I am incredibly thankful to Yurii Stasinchuk for all his help and support throughout my masters. I have had several fun and insightful discussions with him which have helped me professionally and personally. I am also thankful to Matouš Vrba for his advice and insights whenever I needed it. I am also thankful to him and Petr Štibinger for their valuable feedback on this thesis.

At last, I am thankful to my family and friends who have been extremely patient throughout my masters. This thesis and my studies would not have been possible without their love and support.

Abstract

A system composed of multiple UAVs can solve several complex tasks which are either difficult or unfeasible with a single UAV. Safe and fast operation are crucial when multiple UAVs are simultaneously working in a shared environment. This thesis focuses on studying and developing a method for collision-free trajectory planning by a multi-UAV system. The proposed method uses sensors on-board the UAV to create an online map of the environment and detect the motion of other UAVs. These detections are employed to predict the future trajectory of the UAVs. The map along with the predicted trajectories is used to design a collision-free trajectory to a pre-specified goal. The proposed method is decentralized and does not require any prior information about the environment or other UAVs which makes it easier to scale to large multi-UAV systems. The method is also independent of any external localization infrastructure making it ideal for real-world applications.

Keywords Unmanned Aerial Vehicles, Multi-UAV Path Planning, Collision Avoidance, Decentralized Systems, Motion Prediction

Abstrakt

Systém složený z několika bezpilotních helikoptér může řešit komplexní úkoly které jsou složité nebo někdy i neproveditelné jedinou helikoptérou. Rychlá a spolehlivá reakce je kritická pro jejich nasazení několika helikoptér ve sdíleném prostoru. Tato práce je zaměřená řešerši a vývoj metod plánování bezkolizních trajektorií pro skupinu bezpilotních helikoptér. Prezentované řešení využívá senzory umístěné na palubě helikoptéry pro vytvoření aktuální mapy prostředí a pro detekci pohybu ostatních helikoptér. Série detekcí jsou poté použity pro predikci jejich budoucích trajektorií. Mapa prostředí dohromady s predikcemi pohybu je využita pro nalezení bezkolizní trajektorie do cílové pozice. Prezentovaná metoda je decentralizovaná a nevyžaduje žádnou předem známou informaci o prostředí nebo o ostatních robotech v týmu. Díky tomu je metoda vhodná i pro velké multi-robotické systémy. Další výhodou navrhovaného řešení je nezávislost na externí lokalizační infrastruktuře. Tato nezávislost vytváří dobrý předpoklad pro jeho reálné nasazení.

Klíčová slova bezpilotní helikoptéry, multi-robotické plánovaní, vyhýbání se kolizím, decentralizované systémy, predikce pohybu

Abbreviations

SCP Sequential Convex Programming

UAV Unmanned Aerial Vehicle

UVDAR Ultraviolet Direction And Ranging

ROS Robot Operating System

MPC Model Predictive Control

EKF Extended Kalman Filter

GNSS Global Navigation Satellite System

RTK Real-time Kinematic

SLAM Simultaneous Localization And Mapping

HMM Hidden Markov Model

SAR Search-And-Rescue

RCA Reciprocal Collision Avoidance

GMM Gaussian Mixture Model

SITL Software-In-The-Loop

Contents

Chapter 1

Introduction

Humans have been designing and developing increasingly complex machines for several centuries. The Industrial Revolution boosted their development to facilitate the life of humans in various ways. Cars and trains transformed the ways of travel, and the assembly lines of the big industries revolutionized production. However, most of these machines were merely mechanical apparatus with no autonomy and always relied on the expertise of human operators. The Information Revolution in the last century brought the age of robots. Since their conception, robots have been designed to help reduce human effort. The last few decades have seen rapid improvement in robot autonomy and design. They have also been readily accepted across industries and daily public life. Several examples like delivery[1], extraterrestrial exploration[2], and even medical surgery[3] are shown in Figure 1.1.

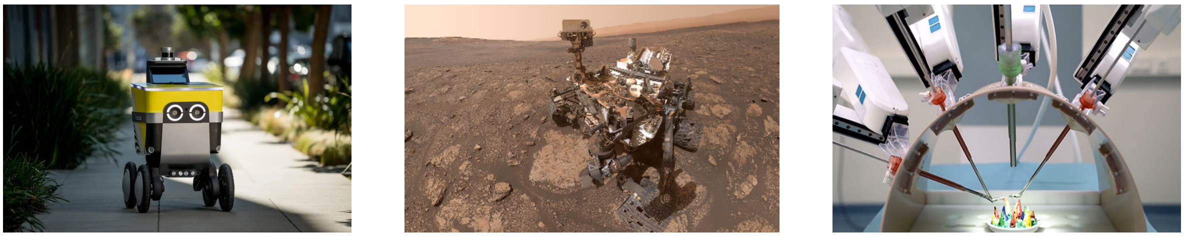


Figure 1.1: Examples of autonomous robots used for delivery, extraterrestrial exploration, and even medical surgery.

Although different robots are suited for different applications, UAVs have gained immense popularity in the last decade due to their wide range of real-world applications. Their use is no more limited to the research community but has seen acceptance across several industries. Unlike ground robots, these aerial vehicles can quickly move through different terrains and environments. They can also travel more considerable distances in a shorter time due to the absence of obstacles in the air. These abilities, combined with the small form factor of modern UAVs, have contributed to the success of the UAV-based systems. Figure 1.2 shows applications of UAVs in delivery[4], surveillance[5] and Mars exploration[6].

In most applications, UAVs are primarily used as standalone devices, often controlled by a pilot who navigates the UAV according to the mission requirements. Commercial plat-

[1] https://www.serverobotics.com/

[2] https://mars.nasa.gov/resources/

[3] https://www.avatera.eu/en/home

[4] https://www.amazon.com/Amazon-Prime-Air/

[5] Reuters

[6] https://mars.nasa.gov

Figure 1.2: Application of UAVs in delivery, surveillance and Mars exploration.

forms developed by DJI[7], Skydio[8] and Autel Robotics [9] can be successfully used for such applications. However, in applications like Search-And-Rescue (SAR) missions [13], [16], [21], [33] and [31], using multiple UAVs simultaneously can drastically reduce the time needed to scout a particular region. Like a team of firefighters or emergency respondents, a team of UAVs can search large areas of disaster-stricken region for fires [25], [27], animals [19], [29], and humans [18]. Multi-UAV systems can also provide rich and extensive data about an environment, which is crucial for monitoring hazardous environments [35] or missions where redundant data is needed for analysis [12].

When multiple UAVs are working simultaneously in a shared environment, avoiding collisions is crucial for safe flight. However, collision avoidance can become a challenge for human pilots with the increasing number of UAVs in the environment. Safe and collision-free motion often requires autonomous mapping and planning algorithms for these multi-UAV systems. Although the commercial platforms from Skydio and DJI are equipped with collision avoidance and mapping algorithms, their application in multi-UAV settings is minimal. One standard method to avoid collisions is to share the position and future trajectory among all the UAVs in the system. Such a system needs a common reference frame like Global Navigation Satellite System (GNSS) to localize all the UAVs. However, obtaining reliable GNSS signal is challenging in many real-world scenarios involving forests and cluttered environments. Relying on such a centralized infrastructure can create a single point of failure, which may lead to the failure of the entire system. Recent progress in relative pose estimation of UAVs has led to the development of decentralized motion planning methods which do not rely on GNSS. Such approaches continuously synchronize the relative reference frames among all the UAVs by sharing information over a communication network.

Although relative pose estimation makes the multi-UAV system independent of the GNSS localization infrastructure, synchronization of reference frames still relies on high bandwidth communication. However, radio communication is often limited by bandwidth and is prone to errors. Also, communication range can become a huge bottleneck when scaling the multi-UAV system to a large size. Thus, several solutions in the literature focus on carefully selecting the content and amount of information to communicate over the shared network. However, even when the UAVs can communicate the entire future trajectory, it is challenging to guarantee collision avoidance in a decentralized system. Since each UAV generates an independent motion plan, it is challenging to synchronize their future motion. Online replanning is often used to react to the changes in the future trajectory of other UAVs, but it is insufficient to ensure safety.

[7] https://www.dji.com/cz/mavic-3

[8] https://www.skydio.com/skydio-2-plus

[9] https://auteldrones.com

This thesis aims to design a decentralized motion planning method for multi-UAV systems with limited use of communication. The proposed system uses relative pose estimation along with shared information between UAVs to modify the future motion plan of each UAV. We explicitly model the uncertainty in the future motion of the UAVs to mitigate the stochastic nature of the real world. The independence from GNSS makes the system suitable for many challenging missions involving forests and cluttered urban environments. The limited communication requirement also opens the possibility of scaling the system to a large team of UAVs.

The remainder of the thesis is organized as follows. The introduction chapter explores state-of-the-art methods and ends with a brief list of contributions. Chapter 2 describes the problem statement and the necessary assumptions. This is followed by a detailed description of the method in Chapter 3 and its application to real-time collision avoidance in Chapter 4 and 5. Chapter 6 presents the experimental analysis of the proposed method and its verification in different scenarios. A brief comparison with the existing literature is presented in Chapter 7. The thesis concludes with remarks on the successfully achieved tasks and ideas for future work.

1.1 Related works

Multi-robot cooperation is a well-studied problem in the literature due to its varied applications in several real-world scenarios. One of the crucial properties of a multi-robot system is planning collision-free paths through the environment. This section explores and analyzes the existing methods from the literature.

1.1.1 Centralized collision avoidance

Several different approaches have been proposed to solve this collision avoidance problem. One of the most common methods is centralized path planning. In such planning techniques, a central motion planner with information about the state of the robots and the environment plans collision-free paths for all the robots. [23] presented a centralized method for multi-robot trajectory generation under non-convex environmental constraints. They formulate trajectory planning as a single optimization problem to obtain collision-free trajectories for all the robots. Similarly, the authors in [28] also solve the trajectory generation problem as a centralized optimization task. However, they consider higher-order dynamics of the robots rather than simpler models used in [23]. Although these methods optimize for several cost functions to account for time-optimality, trajectory smoothness, and collision avoidance, they do not consider the physical properties of UAVs. The work presented in [22] explicitly design the method for UAVs with a multi-level software pipeline. They address the issues related to solving the optimization problem in real-time and downwash from the flying UAVs. The methods in [23], [28] and [22] solve the multi-robot collision avoidance problem and, the resulting trajectories are time and space optimal. However, they rely on the assumption of having precise information about the states of all the UAVs and precise trajectory tracking. Although it is possible to obtain a reasonably precise state estimate using GNSS or Real-time Kinematic (RTK), precise trajectory tracking is challenging to achieve with flying UAVs. Moreover, these methods are not designed for real-time re-planning, which is crucial for safe flight in any real-world mission.

1.1.2 Decentralized and distributed collision avoidance

The centralized path planning techniques rely on a central system (external source or a robot) to plan the desired paths for all the robots. As the failure of the central system can stop the entire mission, it can be challenging to deploy such systems in a real-world environment. Several methods use distributed and decentralized techniques to avoid this centralized bottleneck. In a decentralized multi-robot system, the robots do not rely on a central authority. The work presented in [32] decouples the centralized trajectory optimization problem into individual problems for each robot. The optimization problem on each robot is then solved using Sequential Convex Programming (SCP) with constraints relaxation to find feasible solutions in a shorter time. [11] also presented an online, distributed Model Predictive Control (MPC) method for collision-free trajectory planning. The MPC is coupled with a re-planning technique to avoid collisions caused by disturbances or errors in trajectory tracking. However, they use event-triggered re-planning with no checks for collisions after re-planning. Both these works use B-splines to represent the trajectories of the robots or UAVs, which is faster than computing a trajectory directly in $\mathbb{R}^3$. To make the trajectory computation faster, the authors in [1] present a decentralized multi-robot trajectory planner which uses the trajectory representation from [14]. They show that using [14], the computation time for trajectory planning can be reduced by almost two orders of magnitude when compared to the commonly used B-spine representation. They also present a re-planning technique with several checks to ensure smooth and collision-free flight when moving around other UAVs. Similarly, [8] uses the trajectory representation from [14] to solve a spatio-temporal optimization. [8] uses unconstrained optimization along with local re-planning for collision avoidance. But unlike [1], the authors in [8] demonstrate their method in real-world experiments in a cluttered environment. The distributed and decentralized optimization techniques used in these methods are vital to avoid the risk of a centralized failure. The online re-planning techniques further increase the utility of these methods in a real-world deployment. However, these methods still rely heavily on precise state estimation of all the UAVs in the multi-UAV system. They also assume that the UAVs can always share their current trajectories with others without any lag in communication. However, these assumptions are not valid in many real-world scenarios as the communication between the UAVs is often limited by the channel bandwidth. [8] also uses a frame alignment method to continuously integrate the deviations between relative UAV frames when moving in a forest. This frame alignment is highly dependent on broadcast communication which has bandwidth limitations. Also, to use the shared information, the UAVs need to agree on a common frame of reference. As discussed earlier, GNSS and RTK can be used to provide a common reference frame, but they have degraded performance in forests and dense urban environments.

1.1.3 Collision avoidance without communication

The centralized path planning techniques rely on a central system (external source or a robot) to plan the desired paths for all the robots. When working with UAVs in real-world missions, it is challenging to maintain communication links and other external infrastructure. Communication can become a bottleneck in missions where the UAV collision avoidance can not work without shared information. [9] presented a decentralized collision avoidance method that does not need any communication between the UAVs. The UAVs are assumed to be able to detect the position and velocity of other UAVs at all times and use an MPC to find optimal trajectories. The uncertainty in estimating the position and velocity of other

UAVs is modeled as a Gaussian distribution using a Kalman filter. Similarly, authors in [4] solve the multi-UAV trajectory planning problem using the position and velocity estimation rather than sharing information. However, they model the motion uncertainty of the UAVs using convex polygons around the estimated position of UAVs. These methods do not use a collision check like [1] and [8] but instead use the Reciprocal Collision Avoidance (RCA)-based methods to avoid collisions when moving near other UAVs. The RCA generates a convex polygon around the moving object parametrized by the relative velocity of the object. If the position and velocity uncertainty is bounded, avoiding the RCA generated polygon provides guarantees on collision avoidance. The simplicity of RCA based collision avoidance is beneficial for real-time re-planning and collision avoidance but has its limitations. The convex polygons generated for several moving obstacles and UAVs can over-constrain the optimization problem leading to infeasibility. The work in [6] presents a unified approach combining the advantages of trajectory optimization without communication and strict collision checks. The authors use a hierarchical structure composed of discrete path planning and trajectory optimization only using the position and velocity estimate of other moving robots. The discrete planning can often provide a good initial solution for trajectory optimization, thus accelerating the entire motion planning process. The collision checks help the UAVs to continuously evaluate the trajectories or stop moving if the check fails. The decentralization of decision-making to individual UAVs and independence from communication are crucial for rapid real-world deployment of a large multi-UAV system. The methods in [4], [9] and [6] are ideal candidates for such a system. However, in all these methods, the uncertainty in the motion of other UAVs is implicitly modeled in the convex polygons used to constraint the trajectory optimization. These polygons are often insufficient to represent moving objects. In the worst case, the polygon surrounding the moving object can make the problem infeasible when operating in a cluttered environment.

1.1.4 Collision avoidance with uncertainty models

The centralized path planning techniques rely on a central system (external source or a robot) to plan the desired paths for all the robots. Modeling the moving UAVs using static convex polygons ignores their future motion. As the UAVs can not stop or change their direction of motion arbitrarily, they can end up in a region where a future collision can no longer be avoided. Thus, in an environment cluttered with several obstacles, the UAV trajectory should avoid collisions with the current and possible future positions of other UAVs. One way to reason about the future trajectory is to model the motion uncertainty of the UAVs explicitly. The work presented in [2] uses a probabilistic model for collision avoidance in a multi-UAV system. The authors model the motion uncertainty of the UAVs as a Gaussian Mixture Model (GMM) and use it to generate probabilistic constraints for the trajectory optimization problem. The method is decentralized and does not need to communicate any information among UAVs. However, the authors do not consider any obstacles in the environment, making it unsuitable for use in cluttered environments as static or dynamic obstacles can often make the trajectory optimization infeasible.

This thesis focuses on designing a decentralized collision avoidance method by explicitly modeling the motion uncertainty. The presented method uses a motion planner to avoid the static obstacles and moving UAVs using an occupancy prediction method. We develop on the work presented in [17] for multi-UAV collision avoidance. The authors in [17] present a method where the UAVs communicate their future trajectories for collision avoidance and estimate the motion of the humans using a Bayesian model. This Bayesian model uses a reward function parametrized by the goal of the human. As the environment is cluttered with other humans, obstacles, and UAVs, the simple goal-directed reward function cannot account for the various sources of uncertainties in human motion. Thus, the Bayesian model is further parameterized by the confidence of the UAV in the model of human motion. We use this technique to model the motion of UAVs using the parametrized Bayesian model. However, we do not assume any humans in the environment. We also present a cooperative re-planning technique to avoid deadlocks between the UAVs. This cooperative re-planning is based on communicating a preference number over a shared network. The UAVs also communicate their respective goals for Bayesian motion prediction. The method proposed in this thesis is decentralized, so each UAV makes an individual decision about its future trajectory. The UAVs do not communicate their position and velocity but use direct observation for online estimation. The proposed method does not depend on a GNSS or RTK, which makes it useful for real-world deployment.

1.2 Contributions

The main contributions of this thesis are the described below.

1. A decentralized method for collision-free motion planning for a multi-UAV system operating in a cluttered environment.
2. A probabilistic approach for modeling future trajectory of the UAVs which is robust to modeling and sensory errors.
3. A cooperative re-planning technique to avoid deadlocks in a cluttered environment.
4. A multi-UAV system which shares very small amount of information and only in the local neighborhood. This makes the system easily scalable to large group of UAVs. The method is independent of any global localization system which is crucial for real-world deployment.

1.3 Mathematical notation

The rest of the thesis uses the following mathematical notation.

$\mathbf{x}$	a vector
$\hat{\mathbf{x}}$	a unit vector
$\mathbf{I}, \mathbf{0}$	an identity matrix, a zero matrix
$\mathbf{X}[1, ..., n]$	a matrix column, [start, end]
$\mathbf{x}^t, \mathbf{p}^t$	the state and position at the time t, $\mathbf{p}^t = \mathbf{x}^t[1:3]$
$\mathbf{A}, \mathbf{B}, \mathbf{u}$	the LTI system matrix, input matrix and input vector

Table 1.1: Mathematical notation, nomenclature and notable symbols.

Chapter 2

Problem Description

This thesis considers a cooperative group of multiple UAVs operating in a shared environment. Depending upon the task specification, the UAVs are assigned individual goal positions in the environment. Figure. 2.1 presents a simple example of a UAV planning its motion in the presence of other UAV. The thesis aims to design a motion planning method for collision-free navigation to the respective goals of each UAV. The proposed method is decentralized, where each UAV in the multi-UAV system makes independent decisions without the need for a centralized authority. The UAVs only use onboard sensors for motion planning and navigation and share their goal positions over a communication network. The thesis focuses on developing and implementing the method such that the UAVs can be deployed at arbitrary positions in the shared environment.

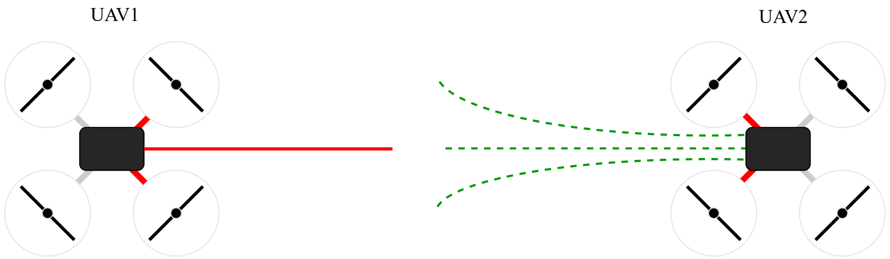


Figure 2.1: The red line represents the trajectory of UAV1 moving towards its goal. The dotted green lines are the possible trajectories for UAV2.

The multi-UAV system is assumed to operate in an environment with obstacles and other UAVs moving towards their respective goals. UAVs must be aware of the obstacles and moving UAVs to reach the goal safely. So each UAV uses a Simultaneous Localization And Mapping (SLAM) system for self localization and creating a map of the unknown environment. We assume that the UAVs do not have a common world reference frame, and they do not share their position with other UAVs. Instead, each UAV uses the Ultraviolet Direction And Ranging (UVDAR) system [15], [20], [24] for camera-based relative pose estimation to obtain a position and velocity estimate of other UAVs. As the UAVs only use onboard sensors, the multi-UAV system is independent of any external localization. Each UAV uses a motion planner to obtain a collision-free path toward its goal. When multiple UAVs are moving in a shared space, the proposed method uses their position and velocity estimates to predict the future trajectory of the UAVs. The predictions are based on a Bayesian prediction model which describes the probability of occupancy for any region around the UAVs, at a specific time. When the UAVs move around each other, this occupancy probability guides the re-planning process of the motion planner. As all the UAVs might re-plan their paths in response to the other UAVs, it can lead to a deadlock where UAVs keep re-planning their paths continuously. To avoid such a situation, this thesis also presents a cooperative re-planning technique based on sharing a preference number. The UAV with a lower preference number re-plans its path while the one with higher number continues to move without re-planning. All the UAVs in our multi-UAV system are identical, so the rest of this thesis will describe everything with respect to one of the UAVs, hereafter called the focal UAV. Note that in the proposed decentralized system, all the decisions are made by the individual UAVs making it possible to describe the method from the point of view of the focal UAV.

2.1 UAV model

Generally, the UAVs are modeled in a high dimensional state space to be able to control the motion precisely and with high fidelity. However, it can often be difficult and computationally expensive to use such a model for real-time motion planning. The multi-UAV system might need frequent re-planning when operating around other moving UAVs in an unknown environment. So, we model each UAV by the following simplified model.

$$\mathbf{x}^{t+1} = \mathbf{A}\mathbf{x}^t + \mathbf{B}\mathbf{u}^t, \tag{2.1}$$

where $\mathbf{x}^t = [x, y, z, \dot{x}, \dot{y}, \dot{z}]$ represents the state of the UAV and $\mathbf{u}^t$ is the control input, at time t. We assume that the motion of the UAV can be controlled using velocity input, so the control input $\mathbf{u}^t = [v_x, v_y, v_z]$, where each component is the velocity in x, y, z axis, respectively. As the control input represents the velocity, it is bounded by the physical properties of the UAV. Both the state $\mathbf{x}$ and input $\mathbf{u}$ are expressed in a world coordinate frame as shown in Figure. 2.2. The matrices $\mathbf{A} \in \mathbb{R}^{6\times 6}$ and $\mathbf{B} \in \mathbb{R}^{6\times 3}$ are the system and input matrices given by

$$\mathbf{A} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \mathbf{B} = \begin{bmatrix} \delta t\mathbf{I} \\ \mathbf{I} \end{bmatrix}, \tag{2.2}$$

where δt is the time step. Since the multi-UAV system is cooperative, the UAVs are aware of the motion model of others and use it for predicting their future trajectory.

2.2 Prediction of future trajectories

The UAVs might encounter each other several times while moving towards their goals. In the proposed multi-UAV system, each UAV uses onboard cameras for relative pose estimation [15] and does not share its position with others. However, in order to predict the motion of surrounding UAVs using the motion model in (2.1), the focal UAV must know the control inputs of the observed UAVs. We assume that the UAVs do not share their control input and the velocity. Instead, the focal UAV estimates the state $\mathbf{x} := [x, y, z, \dot{x}, \dot{y}, \dot{z}]$ using an Extended Kalman Filter (EKF). Since the moving UAVs can be occluded by the obstacles in the environment and change their paths by re-planning, the motion model in (2.1) is insufficient to estimate the state. Building on the work in [30], the EKF estimate is based on the position of the surrounding UAV as observed from the onboard cameras.

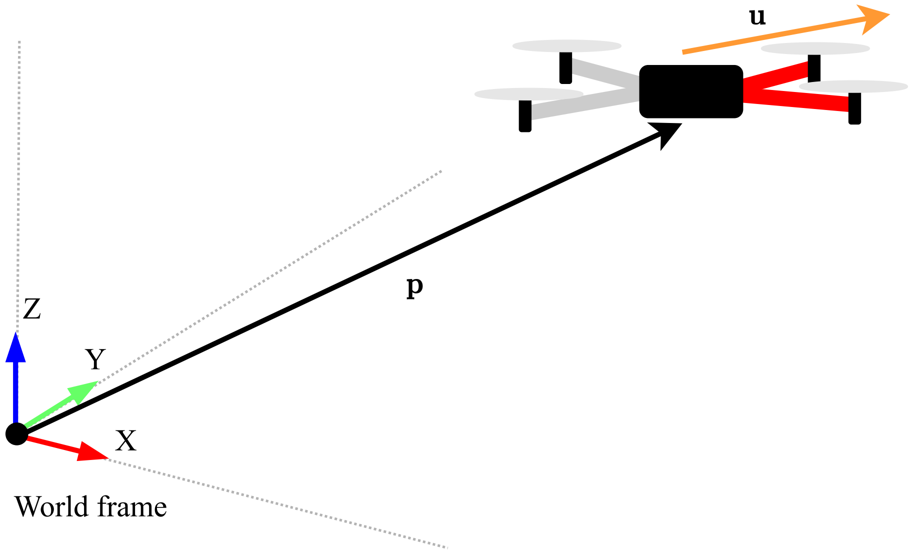


Figure 2.2: The position $\mathbf{p}$ and control input $\mathbf{u}$ of UAV in the world reference frame.

Unfortunately, the estimate of the current state $\mathbf{x}$ is often not enough to avoid collisions with a moving UAV. The future trajectory of UAVs is needed to provide stronger guarantees on collision avoidance [7]. Due to their large size, sharing motion plans or future trajectories can be demanding on the communication network. Several checks are also needed to synchronize the motion plans according to the shared trajectories. Instead, in the proposed multi-UAV system, each UAV shares its goal $\mathbf{g} \in \mathbb{R}^3$. The focal UAV uses a Bayesian predictive model to estimate the future trajectory of the surrounding UAVs using the goal $\mathbf{g}_i$ of the i-th UAV. As the motion model (2.1) is deterministic, the future trajectory $\tau(\mathbf{x}^t, \mathbf{u}^{t:t+T})$ of the i-th UAV can be described as

$$\tau(\mathbf{x}_i^t, \mathbf{u}_i^{t:t+T}) = (\mathbf{p}_i^t, \mathbf{p}_i^{t+1}, ..., \mathbf{p}_i^{t+T}), \tag{2.3}$$

$$\mathbf{p}_i^k = \mathbf{x}_i^k[1:3], \tag{2.4}$$

$$\mathbf{x}_i^{k+1} = \mathbf{A}\mathbf{x}_i^k + \mathbf{B}\mathbf{u}_i^k, k \in \{t, ..., t+T-1\}, \tag{2.5}$$

where $\mathbf{x}_i^t$ is the starting state and $\mathbf{u}_i^{t:t+T}$ is the sequence of control inputs to be applied to $\mathbf{x}_i^t$. T represents the prediction time horizon and is a design parameter. Note that, any possible future trajectory from a given state $\mathbf{x}^t$ can be described by an appropriate sequence of control inputs $\mathbf{u}^{t:t+T}$. Thus, we define the set of all possible reachable positions from $\mathbf{x}^t$ in time horizon T as

$$\mathcal{R}(\mathbf{x}_i, t) = \{\tilde{\mathbf{p}}_i : \exists \mathbf{u}_i^{t:t+T}, \tilde{\mathbf{p}}_i \in \tau(\mathbf{x}_i^t, \mathbf{u}_i^{t:t+T})\}. \tag{2.6}$$

The set $\mathcal{R}(\mathbf{x}_i, t)$ is called the reachable set of the i-th UAV (see Figure. 2.3). The reachable set is bounded by the 3D space that can be traversed by the UAV using a bounded control input in time horizon T. The probability $P(\mathbf{p}_i^k; \mathbf{g}_i)$ of any position $\mathbf{p}_i \in \mathcal{R}(\mathbf{x}_i, t)$ then describes the occupancy of the position at time $k : t \leq k \leq t+T$. This occupancy probability is used by the motion planning method to design a collision-free path. Chapter 3 discusses the process of obtaining and updating the occupancy probability $P(\tilde{\mathbf{p}}^t; \mathbf{g})$ in more detail.

The reachable set of positions can grow significantly with an increase in the time horizon T. Updating occupancy probability on such a large set is computationally expensive. However,

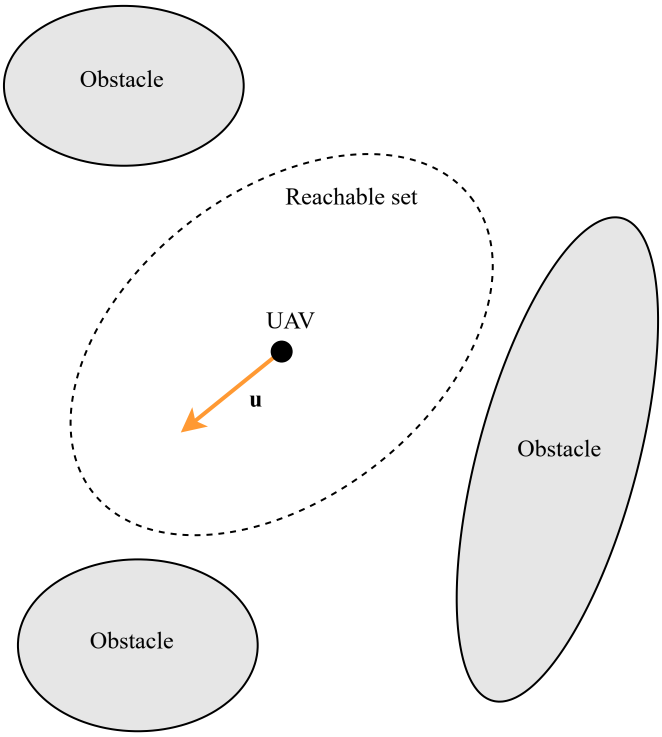


Figure 2.3: Reachable set of positions.

it is often not needed to keep track of the trajectories of UAVs far away as a collision might only occur far in the future or not at all. So we define the set of immediate neighbors of the UAV at time t as

$$\mathcal{N}^t \quad = \quad \{i : \|\mathbf{p}^t - \mathbf{p}_i^t\| \leq D_0\}, \tag{2.7}$$

where $\mathbf{p}^t$ is the position of the focal UAV and $\mathbf{p}_i^t$ is the estimate of the position of i-th UAV, at time t. D_0 is the minimum distance needed to be considered a neighbor.

2.3 Motion planning

A motion planning algorithm uses information about obstacles in the environment to generate a collision-free trajectory towards a given goal. The motion planner must know the positions of static obstacles and trajectories of moving obstacles to generate this collision-free motion plan. We represent the path generated by the planner as

$$\pi : [0, 1] \to \mathbb{R}^3, \tag{2.8}$$

where $\pi(0) = \mathbf{x}^t$ and $\pi(1) = \mathbf{g}$. So, the motion plan starts at the position of a UAV at time t and ends at the goal position. Given the motion plan of a UAV, we say that the motion plan has a collision with a neighbor $i \in \mathcal{N}^t$ if the following condition is satisfied for any $k \in [0, 1]$.

$$\pi(k) \in \mathcal{R}(\mathbf{x}_i, t), \tag{2.9}$$

where $\mathcal{R}(\mathbf{x}_i, t)$ is the reachable set of positions from state $\mathbf{x}_i$ at time t.

Ideally, the motion planner can generate a collision-free path by avoiding all the positions in the reachable sets of all its neighbors. However, the reachable set often encompasses a large volume even for shorter time horizons. Planning a path with such strong constraints can often make it impossible to find a path towards the goal. Fortunately, in many cases it is not needed to avoid all the possible future positions of the neighbors as the UAVs do not move arbitrarily but towards their respective goals. We model the future motion of the neighbor UAVs and use the occupancy probability to ensure collision-free motion. The occupancy probability is also parameterized by the confidence of the focal UAV in the model of its neighbor.

For each neighbor in $\mathcal{N}^t$, the focal UAV computes the occupancy probability at all the positions inside the reachable set. Thus, the UAV maintains a probability distribution over the reachable set of positions for each time instance $k : t \leq k \leq t + T$. A path is discarded if the collision probability is more than a pre-defined threshold value. The motion planner then plans a new path with a modified map of the environment to avoid the previously found collisions.

2.4 Cooperative re-planning

As the UAVs make individual motion plans, it is not possible to synchronize the re-planning process when a collision is detected with another UAV. Thus the UAVs can keep re-planning a new path in response to the motion of the other UAV which creates a deadlock situation. To avoid such a situation, a preference number is shared between the neighbor UAVs. The focal UAV uses these preference numbers to select neighbors for collision avoidance. The rest of the unselected neighbors are assumed to avoid the focal UAV by re-planning their paths.

Chapter 3

Confidence-aware motion prediction

While moving in a cluttered environment, a UAV can encounter several different types of obstacles. The static obstacles that form part of the environment are represented in the map by most of the standard SLAM algorithms. However, moving objects are hard to represent in such a map as most SLAM algorithms rely on static landmarks. The noise in sensor measurement also makes it difficult to represent a moving object reliably. As seen in Figure. 3.1, the object is represented by a trail of blue boxes which do not reflect original size. Thus, it is often easier to model these objects separately from the static map of the environment and reason about them independently. Due to their dynamic nature, collision avoidance with moving objects is complex and might not be guaranteed in many cases. When the UAV moves around another moving object, it must reason about the current and possible future states of the object to prevent any collisions. Not considering the future motion of a fast object can easily lead the UAV to a state where the collision can no longer be avoided. This chapter develops a motion prediction method to obtain a probability distribution describing the occupancy of the region around a moving UAV.

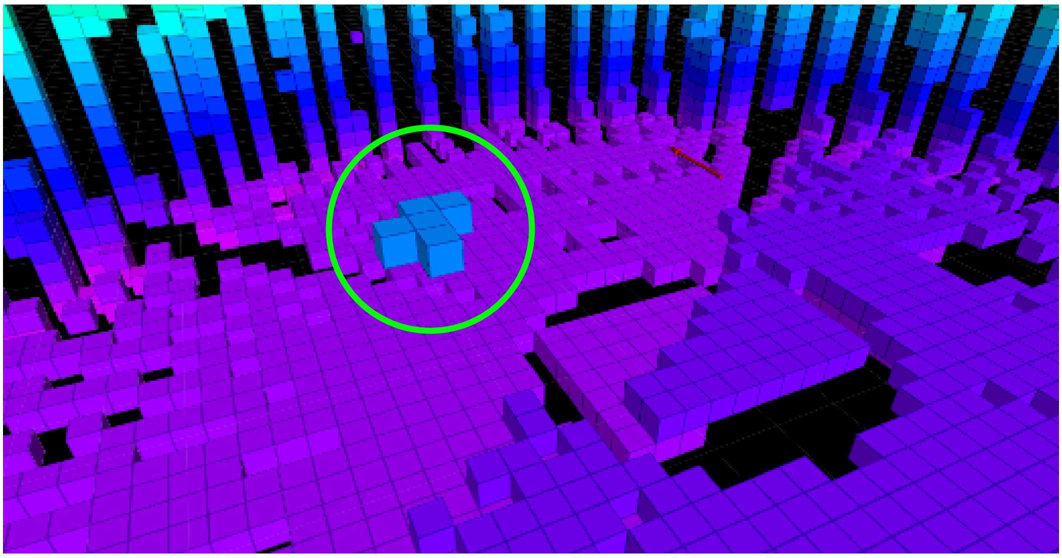


Figure 3.1: SLAM suffering from moving objects. The floating blue boxes inside the green circle show a moving object as represented by the SLAM algorithm.

In the problem discussed in this thesis, the only moving objects are UAVs, which collectively form the multi-UAV system. One of the difficulties in developing a predictive model is correctly describing the motion the moving object. However, in our cooperative multi-UAV system, the UAVs are aware of the motion models which simplifies this problem. The motion model described in (2.1) evolves as a function of the control input $\mathbf{u}$ provided at any time t. When the UAV follows a motion plan, the control input can be obtained from the trajectory generated using the motion plan.

Rather than sharing the entire motion plan, the UAVs share their respective goals with other UAVs. Ideally, the focal UAV can use the goal of the neighbor UAV to find the future trajectory. However, multiple optimal paths can exist toward a goal so it is not feasible to find the exact future trajectory of the neighbor UAV. Instead, we use the goal information to model the motion of the UAV. We also explicitly model the confidence of the focal UAV in the motion prediction and update it as the neighbor UAV moves around the environment. Inspired by the Markovian agent models used in [38], we model the neighbor UAVs as agents, optimizing a reward function that increases in value as the UAV moves towards its goal. This reward $r(\mathbf{p}_i^t, \mathbf{u}_i^t; \mathbf{g}_i)$ is a function of UAV position and control input and is parameterized by the corresponding goal. Given this reward function and under the maximum entropy assumptions described in [36], the probability of choosing a control input at a given position is defined as

$$P(\mathbf{u}_i^t|\mathbf{p}_i^t; \alpha, \mathbf{g}_i) = \frac{e^{\alpha Q(\mathbf{p}_i^t, \mathbf{u}_i^t; \mathbf{g}_i)}}{\sum_{\tilde{\mathbf{u}}} e^{\alpha Q(\mathbf{p}_i^t, \tilde{\mathbf{u}}; \mathbf{g}_i)}}, \tag{3.1}$$

where $Q(\cdot)$ is the value function corresponding to the reward $r(\cdot)$ and α is the model confidence. The Q-function is given as

$$Q(\mathbf{p}_i^t, \mathbf{u}_i^t; \mathbf{g}_i) \quad = \quad \|\mathbf{p}_i^{t+1} - \mathbf{g}_i\|. \tag{3.2}$$

The Q-function gives the value of choosing control input $\mathbf{u}_i^t$ in position $\mathbf{p}_i^t$ when moving towards the goal $\mathbf{g}_i$. The probability in (3.1) is normalized over all the possible control inputs at $\mathbf{p}_i^t$.

Figure 3.2 shows an example of the probability distribution described in (3.1). It is important to note that the reward and subsequently the probability of choosing a particular control input does not depend on the position of any other agent. This assumption does not hold in practice as the UAVs are re-planning their motion to avoid collisions with others. But this interaction can be extremely challenging to model in a simple reward function. It is also computationally expensive to calculate all the possible interactions that might happen even between two UAVs. Instead of reasoning about the interaction of UAVs in the reward function, we limit the re-planning to only one of the UAVs in any pairwise interaction. This is further discussed in Chapter 5.

3.1 Model confidence update

The probability distribution described by (3.1) is also parameterized α. This distribution gets skewed towards the control input leading to the goal, with increasing values of α. Thus, α can be treated as the confidence of the focal UAV in the reward function being able to model the actual motion of the neighbor. As the UAV moves around obstacles and reacts to the motion of other UAVs, the motion modeled by our simple reward function no longer represents the actual trajectory. Thus, a constant value of model confidence α can result in extreme deviation in predicting the probability of control input which makes the model ineffective. In such cases, the model benefits from a time-varying value of model confidence. It is also essential to incorporate the position and velocity observations into the model confidence to track the discrepancy in the reward model.

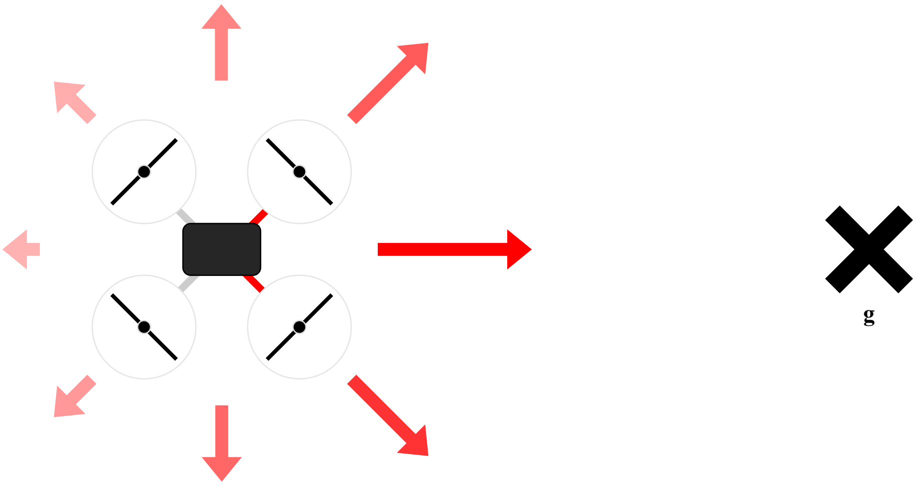


Figure 3.2: Each arrow represents a possible control input for the UAV. The longer and darker arrows have a higher probability than the shorter and lighter ones.

As described in [10], we maintain a Bayesian belief over a set of α values $\mathcal{B}$, which inform our model in (3.1). Since the prediction method is online, performing a Bayesian update over a large set of values is not feasible. Fortunately, as shown in [10], it is sufficient to use a set $|\mathcal{B}| = 5$ to obtain a reasonable real-time performance. The values of α in $\mathcal{B}$ are distributed on a log scale to cover a large spectrum of confidences.

When reasoning about a finite set of the model confidence values in $\mathcal{B}$, we can model them as a state inside a Hidden Markov Model (HMM). The probability distribution over $\alpha \in \mathcal{B}$ describes the belief in each confidence value. For simplicity we start with a uniform probability $h(\alpha)_i^0, \alpha \in \mathcal{B}$. However, other distributions can be used for initial probabilities. As the focal UAV observes the position and velocity of the corresponding neighbors, it updates the posterior probability $f(\alpha)_i^t = P(\alpha|\mathbf{p}_i^{0:t}, \mathbf{u}_i^{0:t})$.

The states of an HMM may transition to other states between two consecutive time instances. Such transitions can occur due to unmodeled dynamics of the HMM states or unobserved variables affecting the states. In the particular case of the UAVs, the model confidence can change as the UAVs re-plan to accommodate new obstacles and other UAVs. If the new path agrees with the reward more than the previous one, it will increase model confidence. However, it can be tricky to model the transition probabilities in the absence of information about re-planning and the paths of other UAVs. Similar to [10], we use a naive transition model for $h(\alpha)_i^t := \mathbb{E}_\alpha[P(\alpha'|\alpha)]$ as

$$h(\alpha)_i^t = \gamma h(\alpha)_i^0 + (1-\gamma) f(\alpha)_i^{t-1}, \tag{3.3}$$

where $h(\alpha)_i^0$ is the initial probability and $f(\alpha)_i^{t-1}$ is the belief from the last time instance. $\gamma \in [0,1]$ is a design parameter. Indeed, $h(\alpha)_i^t$ is the expected conditional probability of any α over the set $\mathcal{B}$. In each time step, the α value may be re-sampled from the initial distribution $h(\alpha)_i^0$ with probability γ or remain the same with probability $1-\gamma$. When a γ is selected very close to 1, it will correspond to the case where the UAV can not rely on the previous belief over the α values and must start reasoning from the initial distribution. On the other hand, small γ values propagate the previous beliefs to the subsequent time steps.

Although the UAV is not aware of the current trajectory of its neighbors, it can still observe their position and velocity. These observations are crucial as they can act as evidence when trying to update the belief $f(\alpha)_i^t$. Using the Markov assumption, the update step for beliefs can be described as

$$f(\alpha)_i^t = \frac{P(\mathbf{u}_i^t|\mathbf{p}_i^t;\alpha,\mathbf{g}_i)h(\alpha)_i^t}{\sum_{\tilde{\alpha}} P(\mathbf{u}_i^t|\mathbf{p}_i^t;\tilde{\alpha},\mathbf{g}_i)h(\tilde{\alpha})}, \tag{3.4}$$

where, $\mathbf{u}_i^t$ is the observed velocity control input and $\mathbf{p}_i^t$ is the estimated position of i-th UAV. The belief in model confidence α for various velocity observations is discussed in the experimental results in Chapter 6. The belief for higher values of α increases if the control input moves the neighbor UAV towards its goal, *i.e.,*the observations support the reward model of the UAV. However, if the control input deviates from the reward model, the belief for higher values of α decreases while the belief for lower values increases.

3.2 Motion prediction

As mentioned earlier in this chapter, the focal UAV is aware of the dynamic model of other UAVs and their respective goals. However, this knowledge is still not enough to predict the actual trajectory of the neighbor UAV. Even with the neighbor trajectory information, it can be challenging to find a path as the trajectories of other UAVs constrains the motion planner. Instead of reasoning about the possible trajectories, the presented motion prediction method reasons about the occupancy probability of a particular volume of space. The occupancy probability of position $\mathbf{p}_i^{t+1}$ can be given by $P(\mathbf{p}_i^{t+1}|\mathbf{p}_i^{0:t},\mathbf{u}_i^{0:t};\alpha,\mathbf{g}_i)$ for any i-th neighbor UAV. Since the UAV motion does not depend on all the previous positions and control inputs (2.1), we can reduce this term under the Markovian assumption to obtain

$$P(\mathbf{p}_i^{t+1};\alpha,\mathbf{g}_i) = \sum_{\mathbf{p}_i^t,\mathbf{u}_i^t} P(\mathbf{p}_i^{t+1}|\mathbf{p}_i^t,\mathbf{u}_i^t;\alpha,\mathbf{g}_i)P(\mathbf{u}_i^t|\mathbf{p}_i^t;\alpha,\mathbf{g}_i)P(\mathbf{p}_i^t;\alpha,\mathbf{g}_i), \tag{3.5}$$

for a particular value of α. As the UAV motion model is deterministic, the expression can be further reduced by summing over the set $\mathcal{I}$ of positions and control inputs that lead to $\mathbf{p}_i^{t+1}$. This set $\mathcal{I}$ is given as

$$\mathcal{I} = \{(\mathbf{p}_i^t,\mathbf{u}_i^t) : \mathbf{x}_i^{t+1} = \mathbf{A}\mathbf{x}_i^t + \mathbf{B}\mathbf{u}_i^t\}. \tag{3.6}$$

The occupancy probability is then rewritten as

$$P(\mathbf{p}_i^{t+1};\alpha,\mathbf{g}_i) = \sum_{(\mathbf{p}_i^t,\mathbf{u}_i^t)\in\mathcal{I}} P(\mathbf{u}_i^t|\mathbf{p}_i^t;\alpha,\mathbf{g}_i)P(\mathbf{p}_i^t;\alpha,\mathbf{g}_i), \tag{3.7}$$

which represents the probability under a particular value of model confidence at time t. As the model confidence is a random variable, marginalizing over α we obtain

$$P(\mathbf{p}_i^{t+1};\mathbf{g}_i) = \mathbb{E}_\alpha[P(\mathbf{p}_i^{t+1};\alpha,\mathbf{g}_i)]. \tag{3.8}$$

The occupancy probability in (3.8) describes the prediction of the future motion of the i-th neighbor UAV. Figure 3.3 shows the occupancy probability example for a moving UAV. The expectation over the probability distribution of α incorporates the belief of varying degrees of confidence in the reward maximizing model of the neighbor UAV. Equipped with this prediction for any position at future time $k : t \leq k \leq T$, the focal UAV can determine the collision probability at time k.

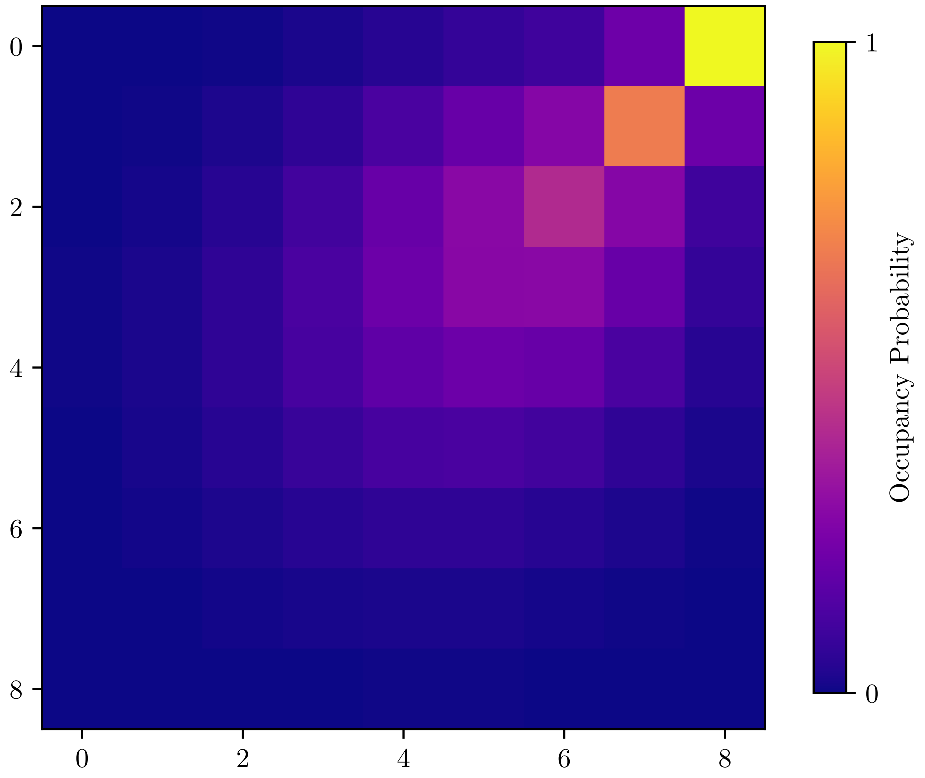


Figure 3.3: Occupancy probability of the region around a moving UAV. The UAV is moving towards the top-right corner in the presented grid.

Chapter 4

Occupancy guided motion planning

In the proposed multi-UAV system, each UAV moves towards a desired goal state inside the environment. Commonly, a motion planning algorithm uses the map of the environment to obtain a collision-free path towards the desired goal. However, it is often difficult to represent moving obstacles on the map so an independent method is needed to reason about their motion. This chapter discusses a motion planning method which uses the map of the static obstacles and the occupancy predictions of moving obstacles (neighbor UAVs) to generate a collision-free path. Since the predictive model represents the occupancy probability, the collision-free guarantees are also probabilistic.

4.1 3D motion planning

As discussed earlier, motion planning relies on having a reliable map of the environment. Such maps are often generated as an output by the SLAM algorithms used for localization and mapping. Since the UAV position and control input lies in $\mathbb{R}^3$, we need to represent the environment and obstacles in 3-dimensions. In this thesis, we use a volumetric map called octomap [34] to efficiently represent the obstacles in 3D.

The octomap is a probabilistic occupancy grid map which uses a simple tree structure called octree to represent a 3D environment. The octree was first introduced by [39] as a method to efficiently model geometric 3D space. The root node of the tree represents the whole environment as a cube of specified dimensions. As the name suggests, the tree has a branching factor of eight. So the volume represented by any node is further subdivided into eight equal parts, each assigned to a child node. The volume is subdivided until a specified volumetric resolution is achieved. Thus each node of the octree represents a part of 3D obstacles in the environment or the open space. In the octomap, the nodes of the octree also specify the occupancy probability of the corresponding volume. The octomap and corresponding octree structure is illustrated in Figure. 4.1.

Equipped with a 3D map of the environment, we use the A* [40] path planning algorithm to find a collision-free path to the goal of the UAV. As the UAV has a finite dimension, it can not go arbitrarily close to the obstacles in the environment. In order to account for its finite size, we inflate the obstacles in the octomap by a distance R_0 using a Euclidean Distance Transform method. Thus, in the inflated map, any node at a distance of R_0 from an obstacle is also considered occupied. An example of the inflated map is presented in Figure. 4.2. This simple procedure is quite effective in accounting for robot dimensions when working with occupancy maps. The A* algorithm uses a heuristic function to guide the path search. We use the Euclidean distance metric as the heuristic function which is given by

$$h(\mathbf{p}, \mathbf{q}) \quad = \quad \|\mathbf{p} - \mathbf{q}\|, \tag{4.1}$$

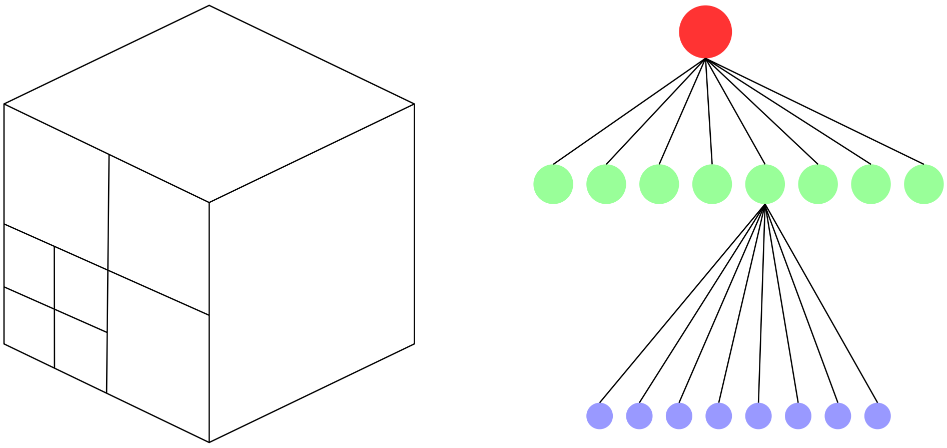


Figure 4.1: The volumetric division of space in an octree. The red node represents the root of the tree and each node has 8 child nodes.

where $\mathbf{p}$ and $\mathbf{q}$ are position vectors of any two nodes in the octomap. As the A* algorithm is optimal under the Euclidean distance heuristic function, the motion plan obtained from the algorithm is the shortest path from the UAV position to its goal position.

The volumetric partitioning encoded in the octree structure enables efficient search of neighbors and occupancy of a given volume. Thus the A* algorithm is able to perform fast neighborhood search in the vicinity of any node of the octree. As the neighborhood search is one of the computationally demanding parts of the planning process, the structure of octree is highly beneficial for path planning. Additionally, the hierarchical space partitioning significantly speeds up the computation of distances between 3D points in the map. These features make octomap an ideal choice to be deployed on robots for real-time planning in different kinds of 3D environments [5], [26].

4.2 Neighbor collision avoidance

Unfortunately, the 3D motion planner designed for deterministic obstacles can not be used directly with predictive models of moving UAVs. The planner relies on checking boolean occupancy and has no notion of time varying occupancy probability. In order to prevent collisions with the moving UAVs, the planner must check the collision of a path with the reachable set of the neighbor UAVs. The occupancy probability of positions in the reachable set provide the probability of collision if the UAV follows the reward model. The planner can thus assign a safety value to every path depending upon the probability of collision with any of the neighbor UAVs.

The collision of a path with a reachable set as defined in (2.9) does not take into account the finite size of the focal UAV. It also ignores the tracking error of the state of the focal UAV when following the path. Instead, we use the trajectory of the focal UAV to define the collision condition with the neighbor $i \in \mathcal{N}^t$ as

$$\|\mathbf{p}^k - \tilde{\mathbf{p}}\| \leq R_0, \tilde{\mathbf{p}} \in \mathcal{R}(\mathbf{x}_i, t), \tag{4.2}$$

where $\mathbf{p}^k$ is any point in the trajectory $\tau(\mathbf{x}^t, \mathbf{u}^{t:t+T})$ of the focal UAV. The trajectory is generated using the path obtained from the motion planner. Thus, a path is in collision with

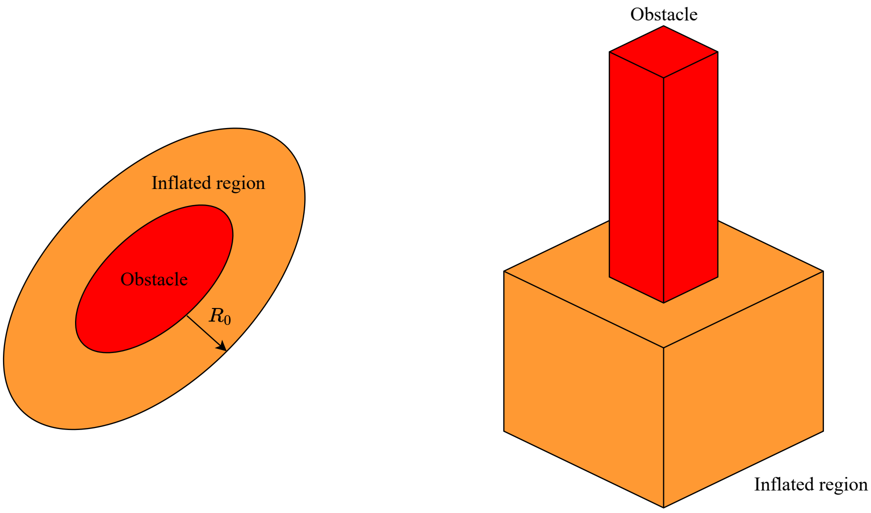


Figure 4.2: Inflated map.

the i-th neighbor if any of the points on the corresponding trajectory is within distance R_0 from the reachable set. We define the set of colliding trajectory points as

$$\mathcal{C}_i := \{\mathbf{p} : \|\mathbf{p} - \tilde{\mathbf{p}}\| \leq R_0, \tilde{\mathbf{p}} \in \mathcal{R}(\mathbf{x}_i, t), \mathbf{p} \in \tau(\mathbf{x}^t, \mathbf{u}^{t:t+T})\}. \tag{4.3}$$

Thus, the probability of collision of any point $\mathbf{p}^k$ on the trajectory is given by

$$P_{cl}(\mathbf{p}^k \in \mathcal{C}_i) = P(\mathbf{q}^k; \mathbf{g}_i), \tag{4.4}$$

$$\mathbf{q} = \operatorname*{argmin}_{\tilde{\mathbf{p}} \in \mathcal{R}(\mathbf{x}_i, t)} \{\|\mathbf{p}^k - \tilde{\mathbf{p}}\|\}, \tag{4.5}$$

where $\mathbf{q}$ is the closest point inside the reachable set. However, this quantity can not be directly give us the collision probability of the entire trajectory. Using the worst case analysis, we can upper bound the probability of collision for trajectory $\tau(\mathbf{x}^t, \mathbf{u}^{t:t+T})$ by

$$P_{cl}(\tau) \leq P_{ub} := 1 - \prod_{k=t}^{t+T} P(\mathbf{p}^k \notin \mathcal{C}_i | \mathbf{p}^s \notin \mathcal{C}_i, t \leq s < k), \tag{4.6}$$

where all $\mathbf{p} \in \tau(\mathbf{x}^t, \mathbf{u}^{t:t+T})$. The right hand side of (4.6) describes the probability that no part of the trajectory is inside the collision set $\mathcal{C}_i$. To evaluate this upper bound, it is necessary to compute the conditional probability $P(\mathbf{p}^k \notin \mathcal{C}_i | \mathbf{p}^s \notin \mathcal{C}_i, t \leq s < k)$. However, this can be difficult to compute in practice as the number of points outside the set $\mathcal{C}_i$ is very large. To make the computation of $P_{cl}(\tau)$ feasible in real-time, we will further simplify the upper bound in (4.6). We use the analysis presented in [10] to make these simplifications. First, we assume that the probability of any $\mathbf{p}^k \notin \mathcal{C}_i$ and $\mathbf{p}^j \notin \mathcal{C}_i$ is independent. This assumption effectively removes the conditioning in the right hand side of (4.6). But this approximation is not necessarily true for continuous trajectory of a UAV as it dictates that any two points on the trajectory can independently lie anywhere in the space. To improve upon this approximation we incorporate the time dependence of collision events into the probability terms. We consider $\mathbf{p}^{k0}$ as the earliest trajectory point to lie inside $\mathcal{C}_i$. So any conditional probability term $P(\mathbf{p}^k \notin \mathcal{C}_i | \mathbf{p}^s \notin \mathcal{C}_i, t \leq s < k), k = t : k0$ can be replaced by 1. Since a collision is already detected at $k0$, we ignore the terms for $k \geq k0$. This approximation gives us a lower bound on P_{ub} from (4.6).

$$P_{ub} \geq 1 - P(\mathbf{p}^{k0} \notin \mathcal{C}_i) = P(\mathbf{p}^{k0} \in \mathcal{C}_i). \tag{4.7}$$

Although this approximation ignores the possibility of future collisions, it still provides a reasonable probabilistic measure of collision. To further improve the approximation, we use the maximum probability at point $\mathbf{p}^{k0}$. Thus, the probability of collision of a trajectory can be approximated as

$$\begin{aligned} P_{cl}(\tau) &\approx \max_{s \in \{t:t+T\}} P(\mathbf{q}^s; \mathbf{g}_i), \\ \mathbf{q} &= \operatorname*{argmin}_{\tilde{\mathbf{p}} \in \mathcal{R}(\mathbf{x}_i, t)} \{\|\mathbf{p}^{k0} - \tilde{\mathbf{p}}\|\}. \end{aligned} \tag{4.8}$$

The collision of a trajectory is thus approximated with the detection of collision at any point in the trajectory as shown in Figure. 4.3. The probability of this collision is then given by the maximum probability of occupancy of the closest point in the reachable set of the i-th neighbor. The maximum operator in fact gives the most conservative approximation of collision at the closest point which improves the approximation of collision. The path planner discards a path π^t if the corresponding trajectory has a collision probability greater than P_{th}. Note that, the collision probability $P(\mathbf{q}^s; \mathbf{g}_i)$ in (4.8) can be obtained directly from the motion prediction method. For a finite time horizon, $P_{cl}(\tau)$ can easily be computed in real-time which makes it feasible to discard colliding trajectories and computing new paths.

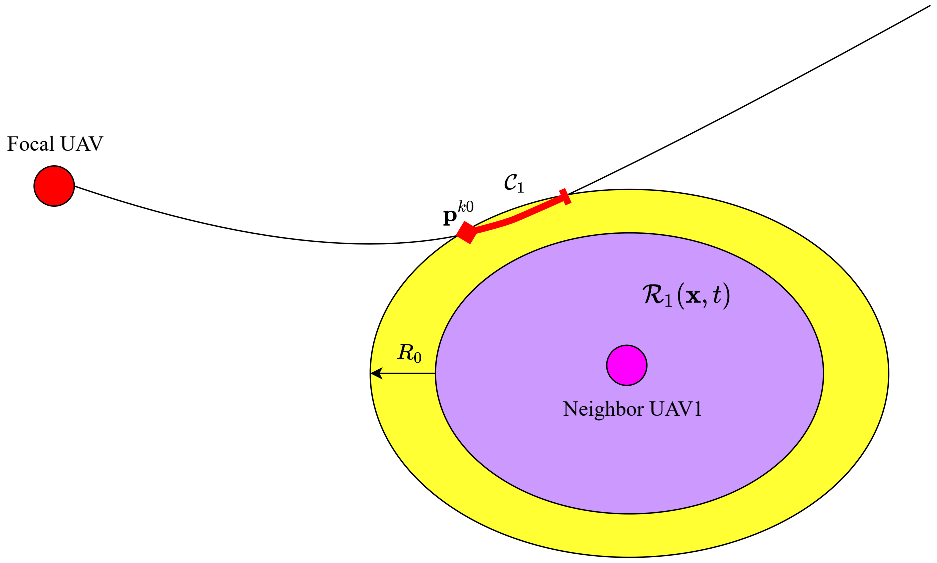


Figure 4.3: This figure show the collision between the trajectory of the focal UAV and the reachable set of UAV1. The red line represents the colliding part of the trajectory where $\mathbf{p}^{k0}$ is the earliest colliding point of the trajectory.

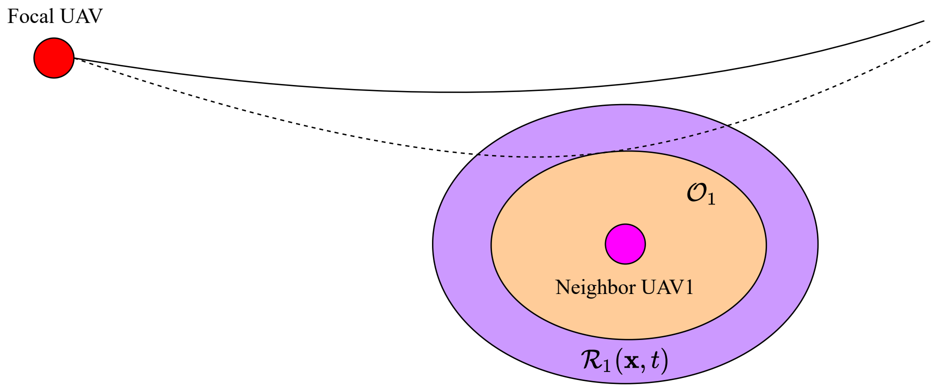


Figure 4.4: Focal UAV re-planning after collision detection. The dashed line represents the original colliding path while the solid line represents the new collision-free path.

4.3 Collision aware re-planning

Equipped with the approximation of collision probability, the planner can discard any paths which collide with probability more than P_{th}. In the event of a collision, the planner must find a new collision-free path for the focal UAV to reach the goal. However, it is necessary for the planner to account for the detected collision in order to avoid it in the new path. As discussed in Chapter 2, considering the entire reachable set of a colliding neighbor as an obstacle can make the planning problem infeasible. Instead, we use the colliding point from the reachable set to design a static obstacle around the neighbor UAV. We define the set $\mathcal{O}_i$ as

$$\mathcal{O}_i = \{\tilde{\mathbf{p}} : \|\tilde{\mathbf{p}} - \mathbf{p}_i^t\| \leq \|\mathbf{q} - \mathbf{p}_i^t\|, \tilde{\mathbf{p}} \in \mathcal{R}(\mathbf{x}_i, t)\}, \tag{4.9}$$

where $\mathbf{q}$ is the closest colliding point from (4.8) and $\mathbf{p}_i^t$ is the position of i-th neighbor at time t. This set contains all the points around the neighbor UAV position that are closer than the colliding point. Since we assume that the trajectories of the UAVs are continuous, any point closer than $\mathbf{q}$ can also lead to a collision. Thus all the nodes corresponding to the points in the set $\mathcal{O}_i$ are marked as occupied in the octomap of the focal UAV. The new path generated after this modification necessarily avoids any collisions with the reachable set up to the collision point $\mathbf{q}$. The nodes are reset after the new collision-free plan is generated. The re-plan process in illustrated in Figure. 4.4.

As the set $\mathcal{O}_i$ uses the Euclidean distance to $\mathbf{q}$ it contains several points that are far away from the colliding trajectory $\tau(\mathbf{x}^t, \mathbf{u}^{t:t+T})$. This might result in setting unnecessary occupancy constraints on the octomap around the i-th neighbor. However, this constraint is still better than considering the entire reachable set as an obstacle. Note that in the worst case, the set $\mathcal{O}_i$ can be equal to $\mathcal{R}(\mathbf{x}_i, t)$. So the worst case performance will be equivalent to setting the entire reachable set as occupied.

Chapter 5

Cooperative collision avoidance

The focal UAV can predict the future trajectory of moving neighbors and use the motion planning method to find a collision-free path towards its goal. In the decentralized multi-UAV system proposed in this thesis, all the UAVs use the same procedure to move towards their respective goals. However, as the neighbor set $\mathcal{N}$ is based on the Euclidean distance, any two UAVs will be neighbors of each other. As a result, in any encounter with a neighbor, both the focal and neighbor UAV make independent decisions about avoiding collision with the other. Such an encounter can easily lead to a situation when both the UAVs keep re-planning indefinitely by reacting to the decision of the other UAV. These situations are often observed among human pedestrians when trying to cross paths with each other (see Figure. 5.1).

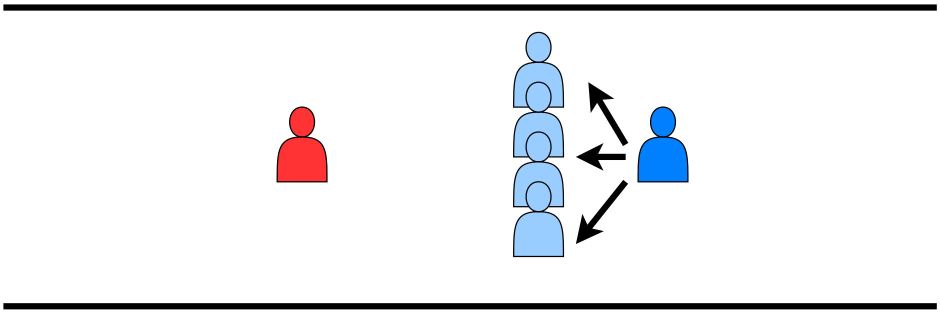


Figure 5.1: Deadlock created between crossing humans as the red human does not know the future position of the blue human.

Unfortunately, it is challenging to model this behaviour in a decentralized system with no hierarchical structure. One of the most common approach is to use a fixed priority of collision avoidance where only the UAVs with lower priority re-plan. In such a setting, the high priority UAVs do not change their trajectory even in the presence of moving UAVs. However, fixing a priority order can often lead to unforeseeable deadlocks in cluttered environments as shown in Figure. 5.2. For instance, when moving in an region occupied by several obstacles, the lower priority UAV might not be able to find a feasible path. In such a situation, it would be crucial for the higher priority UAV to re-plan in order to avoid a collision with the lower priority UAV.

This thesis uses a random hierarchy structure within the multi-UAV system. Whenever the UAVs detect a neighbor, both the UAVs share a randomly generated preference number $\rho \sim \mathcal{U}(0,1)$. So the focal UAV re-plans its path with respect to the reachable set of the i-th neighbor if

$$\rho_f \leq \rho_i, \tag{5.1}$$

where ρ_f and ρ_i are preference numbers of the focal UAV and i-th neighbor, respectively. The preference number is not fixed for any UAV pair and is generated each time a neighbor is added to the set $\mathcal{N}^t$.

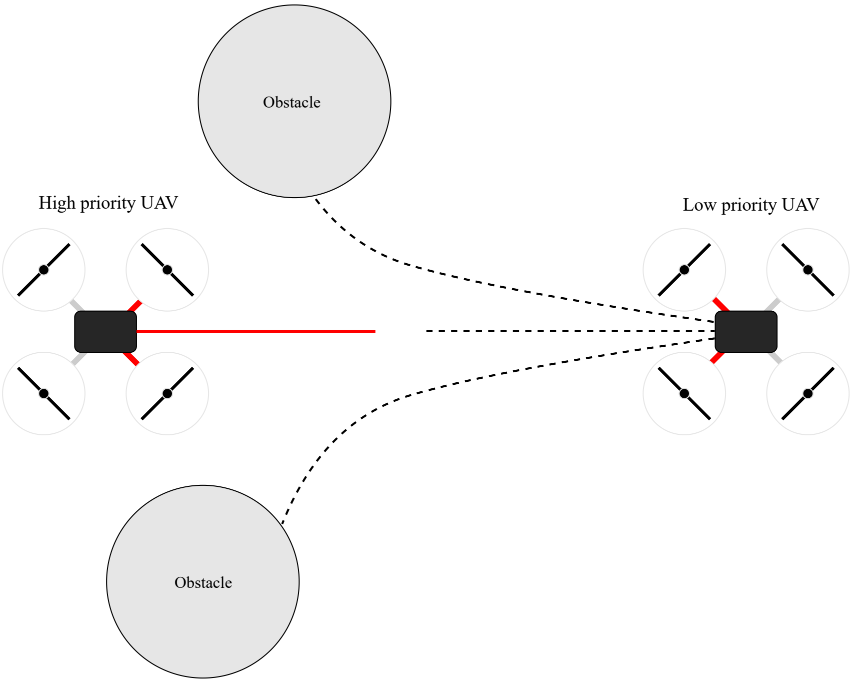


Figure 5.2: Deadlock between UAVs in a cluttered environment. The dashed lines represent the possible trajectories of the low priority UAV. The high priority UAV does not re-plan and follows the red trajectory. However, all the possible trajectories of the low priority UAV lead to a collision.

The randomness introduced by the preference number makes it possible to avoid the deadlock discussed earlier. However, this method can not guarantee prevention of all deadlocks. Figure. 5.3 describes the distribution of the difference between two random variables ρ_f and ρ_i. For a given situation, the deadlock can only be avoided with a specific preference order. Thus, the difference should always lie on one side of the y-axis. As the distribution in Figure. 5.3 is symmetric around the y-axis, the deadlock is avoided with a 50% chance. This is already an improvement over the fixed priority case where the deadlock can not be avoided. It is important to note that the UAVs have a new preference order each time they enter the neighborhood around others. This dynamic nature of the preference further improves the chances of avoiding deadlocks. The simplicity of the preference order based avoidance brings significant advantage when real-time performance is desired. The UAVs in the multi-UAV system can quickly share the preference numbers and re-plan their respective paths without the need for regular communication.

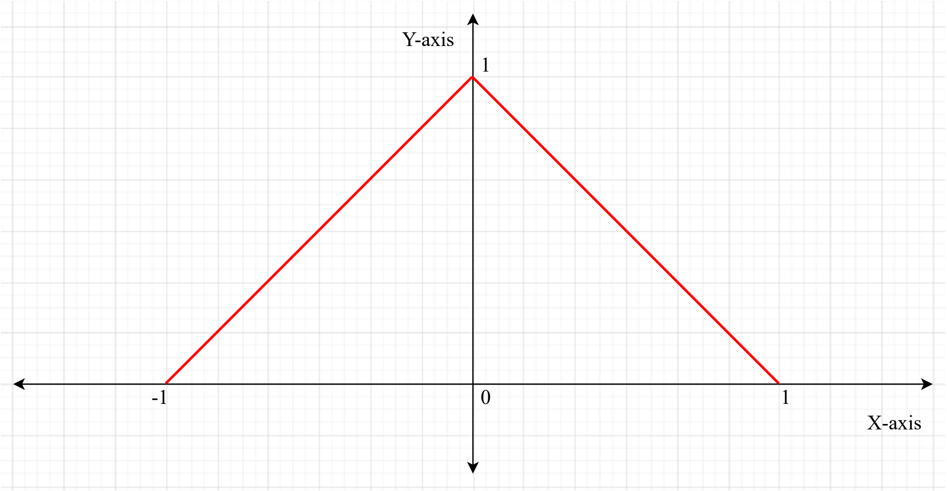


Figure 5.3: Probability density function of the difference between two preference numbers. The X-axis represents the random variable $\rho_f - \rho_i$ while the Y-axis gives the probability of a particular value of random variable.

Chapter 6

Results

This chapter presents the results of various experiments to verify and establish the utility of the method proposed in this thesis. As stated in Chapter 1, this thesis builds on the work from [17] and applies the reward-maximizing prediction model to UAVs. In the first part of this chapter, we verify the Bayesian prediction model from Chapter 3 and analyze its utility and performance when used for UAVs. The next section presents the application of the motion prediction to simulated robots in an ideal simulator. We present a comprehensive analysis of the method in different arrangements of the simulated world. Building upon these results, the last section presents the experimental evaluation in the popular robot simulator Gazebo [37]. The Gazebo simulator uses a physics engine to simulate physical interactions like collisions with the environment and other UAVs. Similar to the ideal simulator, the performance of the method is evaluated in various settings. The physics simulator employs noise models for states of the UAVs which makes the experiments closer to real-world scenarios.

6.1 Occupancy probability

The Bayesian prediction model presented in Chapter 3 is based on a reward function parameterized by goal of the neighbor. The predictions are also parameterized by the model confidence α which is continuously updated using the estimated state of the neighbor UAVs. We present the occupancy probability of the positions in the reachable set $\mathcal{R}(\mathbf{x}, t)$ for various values of the model confidence in this section. To make the analysis easier, we only consider a single neighbor UAV moving in a 2D grid world. It is assumed that the UAV can move to any of its 8-neighborhood cells in a single time instance. In order to visualize the occupancy probability, we limit the size of the reachable set to a grid of size 9×9, centered around the position of the UAV. The motion model from (2.1) was adopted for this case by reducing the dimensions of the quantities, appropriately. At $t = 0$, the UAV is at position $\mathbf{p} = (0, 0)$ and the goal of the UAV is $\mathbf{g} = (100, 100)$. The position and goal of the UAV are in the world coordinate frame and not with respect to the reachable set. We use $\alpha = 1$ as the baseline for comparing the effect of model confidence on the occupancy prediction. As described in (3.1), the probability of the any control input is dependent on the Q-function. However, the Q-function does not change significantly with a single cell transition in the grid so the resulting probability is similar for all the 8-neighborhood transitions. As shown in Figure 6.1, when $\alpha = 1$ the probability distribution spreads towards all the directions with increasing time. Although the probability is higher for the cells closer to the goal $\mathbf{g} = (100, 100)$ (near top-right corner), the difference is not significant as compared to the surrounding cells. As the probability of control input is also dependent on the model confidence, increasing the value of α has a significant effect on the occupancy probability. Figure 6.2 illustrates the case with $\alpha = 10$, where the occupancy probability is higher for the cells closer to the goal. However, when $\alpha = 100$, the occupancy distribution is limited to a small region, as shown in Figure 6.3. The case with $\alpha = 100$ is similar to the situation when the focal UAV is overconfident in the reward-maximizing model of its neighbor UAV. Although, the occupancy probability distribution obtained with $\alpha = 100$ is precise, it tends to ignore the uncertainty is the prediction model. On the other hand, the case with $\alpha = 1$ has high variance and does not provide enough information about the occupancy at any future time. Thus, when the neighbor UAV moves in a cluttered environment any single value of model confidence can lead to an unreliable estimate of the occupancy probability.

Figures 6.4, 6.5 and 6.6 illustrate the occupancy probability for a time horizon $T = 11$. For each value of α, the occupancy probability distribution is similar to the corresponding cases in Figures 6.1, 6.2 and 6.3. As the probability in (3.7) is obtained by computing a sum over the set of all 8-neighbors, it is computationally expensive for a large reachable set. The computation also grows linearly with increase in the time horizon. Since the difference in the distribution from $t = 8$ to $t = 11$ is not significant in Figures 6.4, 6.5 and 6.6, choosing a smaller time horizon can save computation time and resources. Thus we use time horizon $T = 5$ for the rest of the experiments.

As discussed earlier in Chapter 3, it is possible to use different initial belief distributions for model confidence α. Figure 6.7 shows the occupancy probability for a Uniform distribution of belief in α. As the model confidence values are distributed on a log scale the occupancy probability with a Uniform distribution is still overconfident for $t = 5$. On the other hand, when using a Gaussian distribution of belief, the reachable set in Figure 6.8 has a larger variance in occupancy probability as compared to Figure 6.7. However, as the belief distribution is updated using the current position and velocity of the moving neighbor, starting with a Uniform prior does not affect the occupancy probability. The rest of the experiments assume that the initial belief has a Uniform distribution.

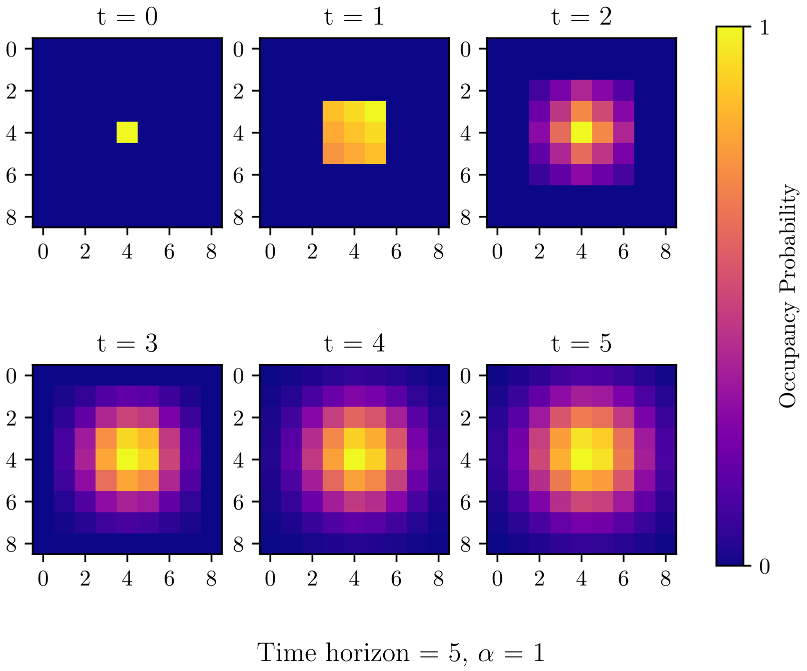


Figure 6.1: Occupancy probability of the reachable set around a neighbor UAV. The current position of the UAV is at the center of the grid.

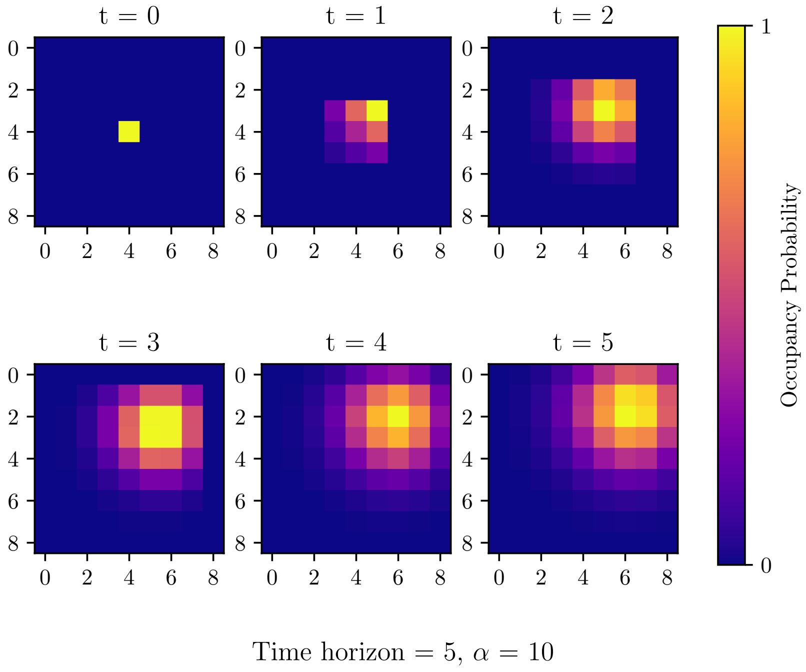


Figure 6.2: Occupancy probability of the reachable set around a neighbor UAV. The current position of the UAV is at the center of the grid.

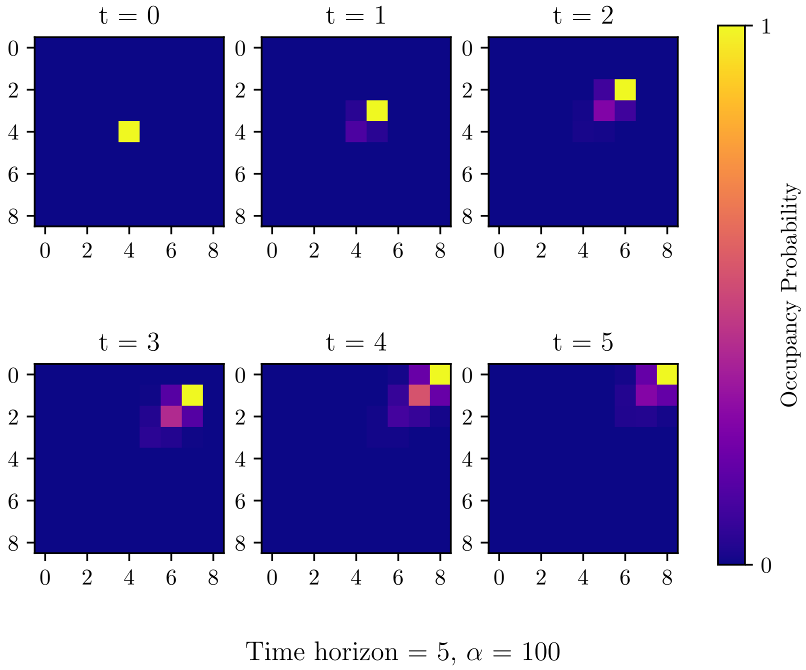
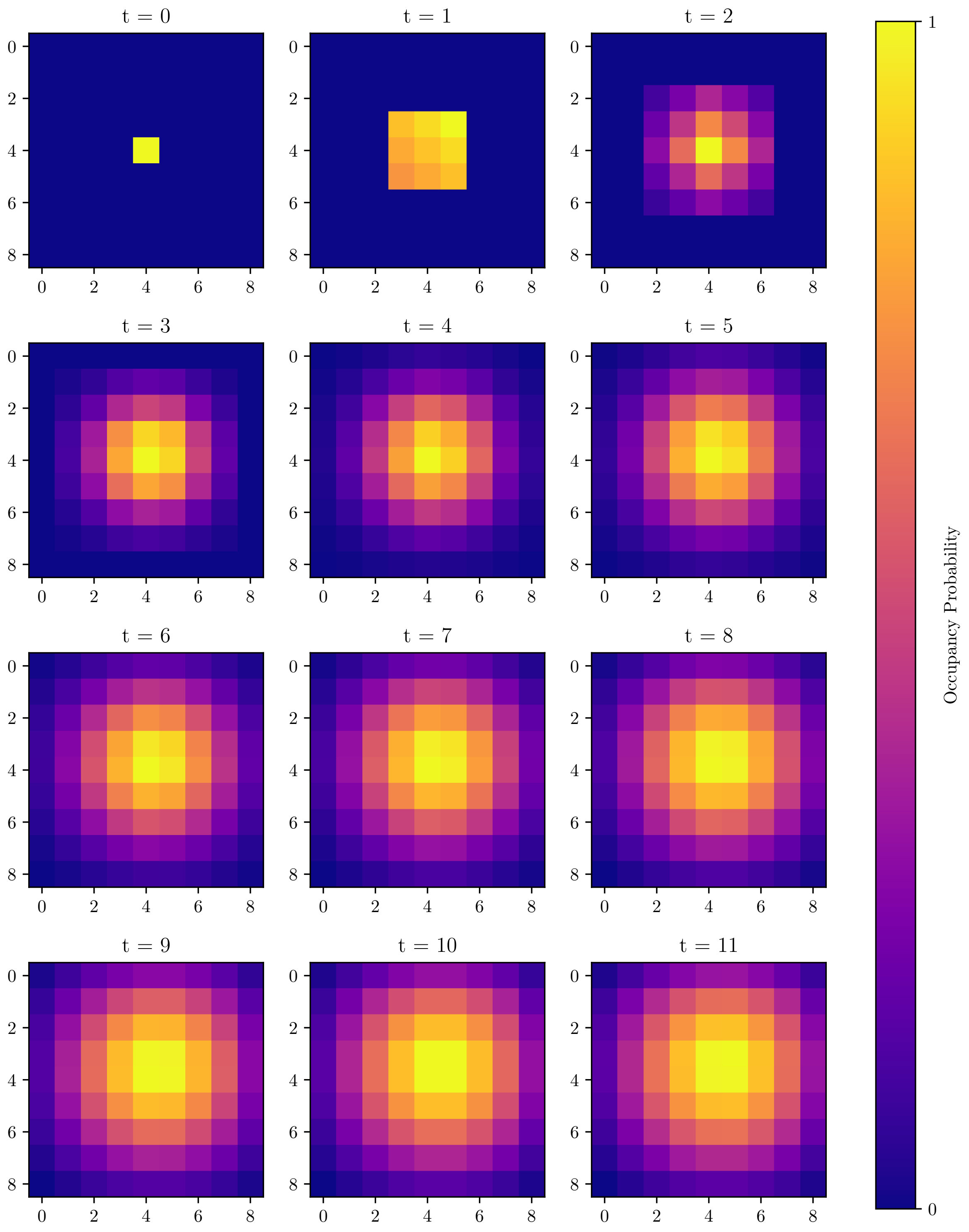
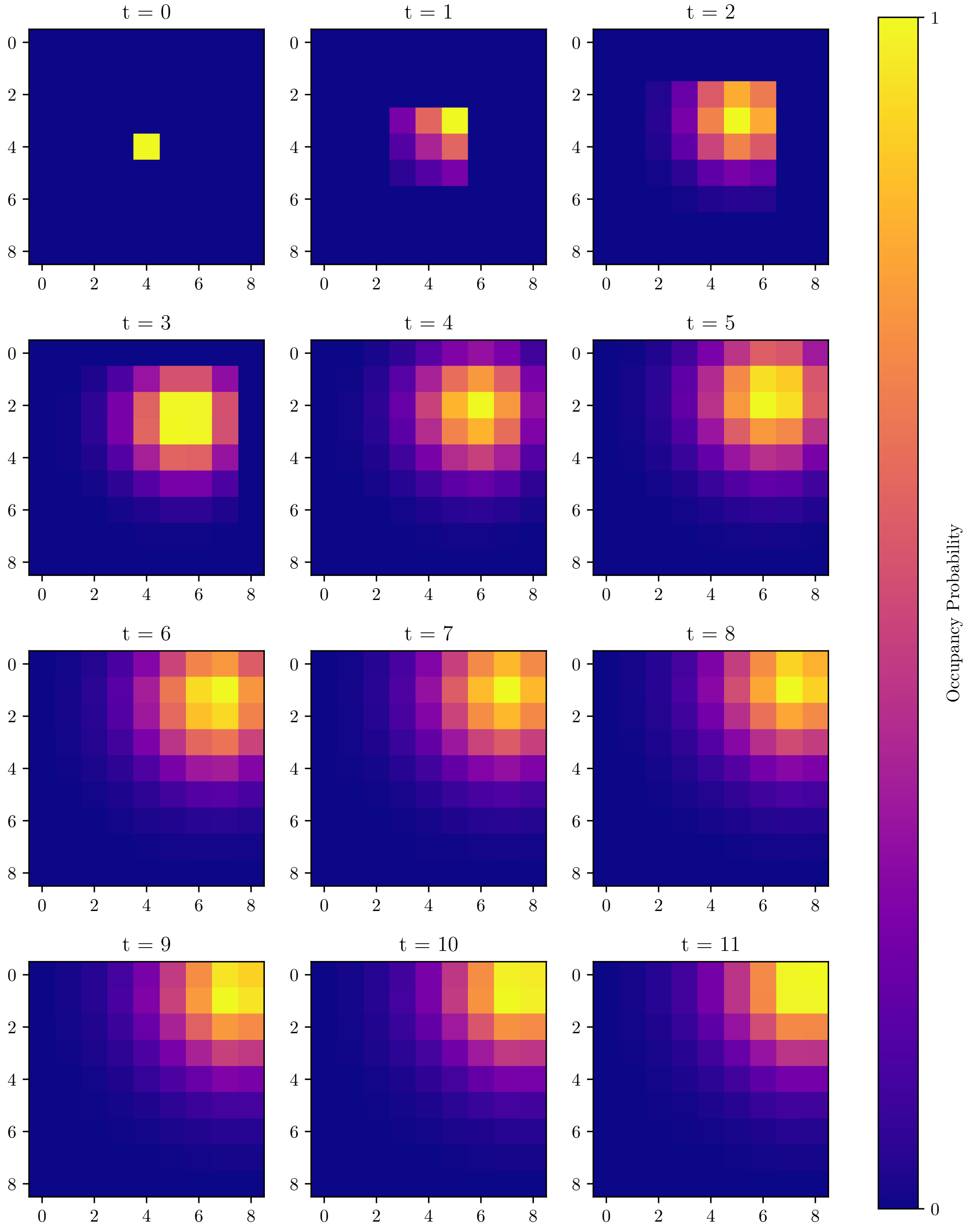


Figure 6.3: Occupancy probability of the reachable set around a neighbor UAV. The current position of the UAV is at the center of the grid.



Time horizon = 11, $\alpha = 1$

Figure 6.4: Occupancy probability of the reachable set around a neighbor UAV. The current position of the UAV is at the center of the grid.



Time horizon = 11, $\alpha = 10$

Figure 6.5: Occupancy probability of the reachable set around a neighbor UAV. The current position of the UAV is at the center of the grid.

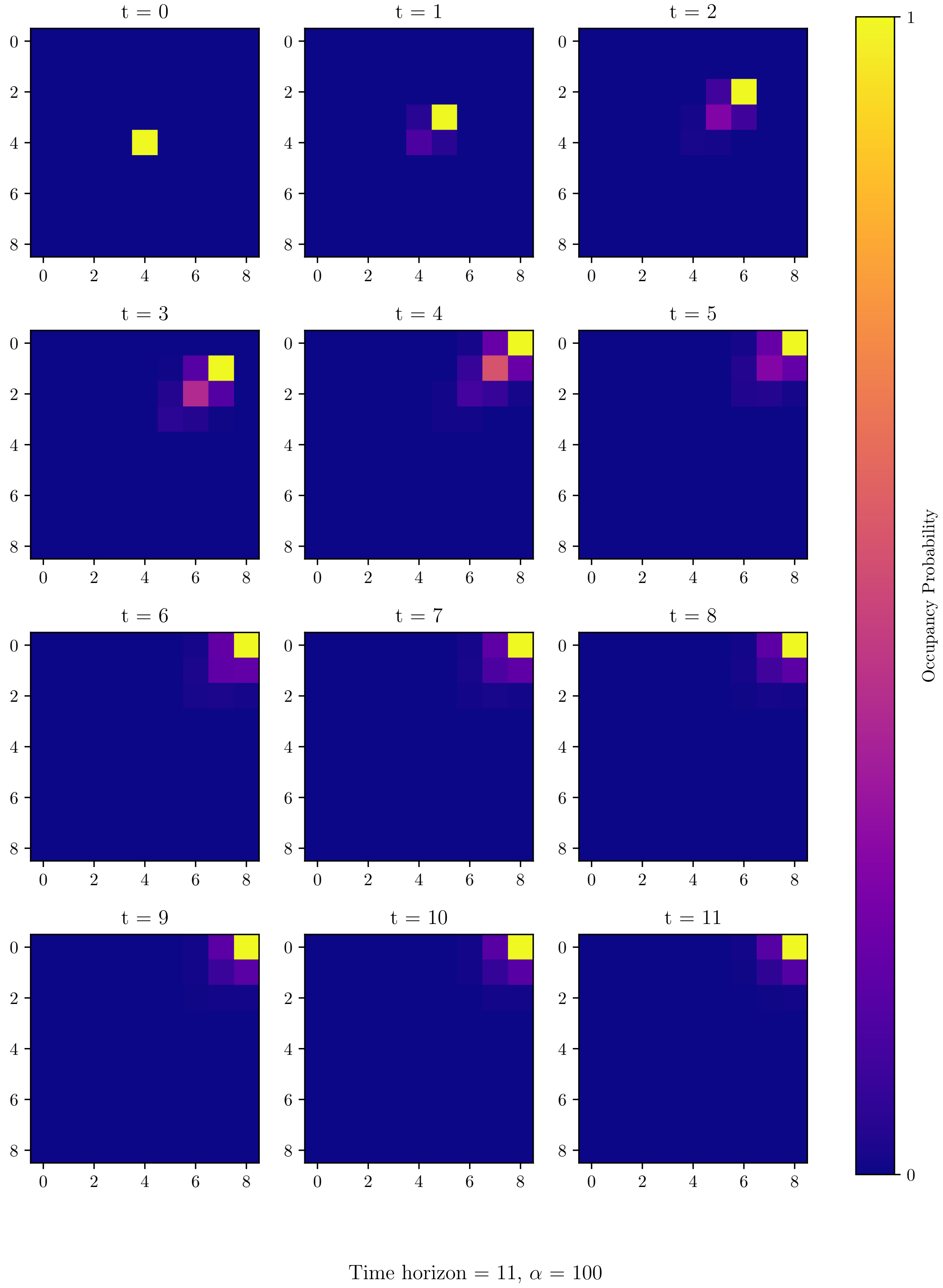
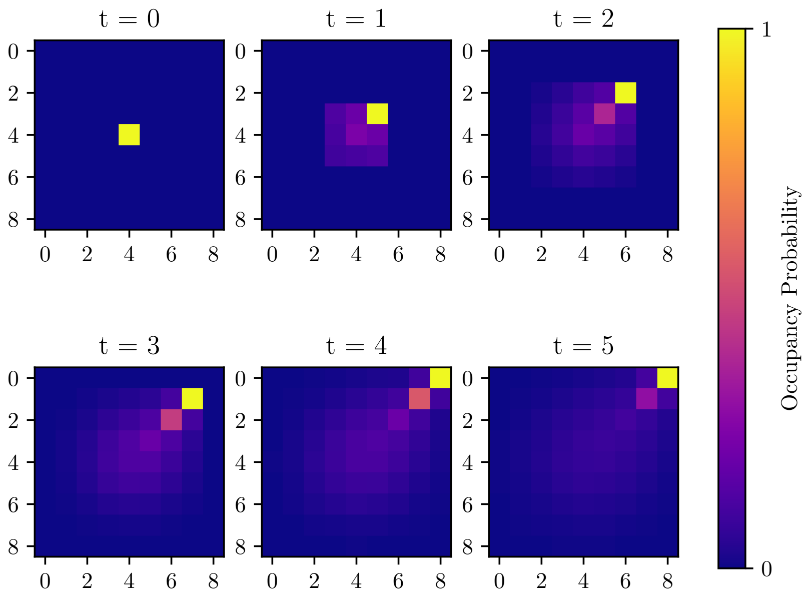
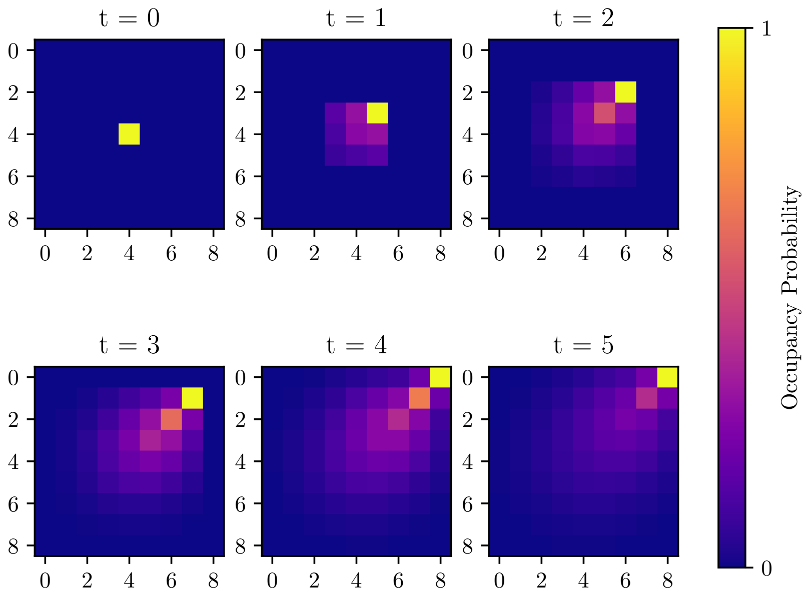


Figure 6.6: Occupancy probability of the reachable set around a neighbor UAV. The current position of the UAV is at the center of the grid.



Time horizon = 5, α with Uniform prior

Figure 6.7: Occupancy probability of the reachable set around a neighbor UAV. The current position of the UAV is at the center of the grid.



Time horizon = 5, α with Gaussian prior

Figure 6.8: Occupancy probability of the reachable set around a neighbor UAV. The current position of the UAV is at the center of the grid.

6.2 Model confidence

The experiments from the previous section demonstrate that using a single value of model confidence can lead to unreliable estimates of occupancy probability. Instead, this thesis uses a belief distribution over a set $\mathcal{B} = \{0.1, 1, 10, 100, 1000\}$ of model confidence values for each neighbor. The belief distribution described in (3.4) is continuously updated using the estimated state of the neighbor UAV. Figure 6.9 demonstrates the evolution of belief distribution for a neighbor UAV moving toward its goal. Since the reward of the neighbor increases when it moves towards its goal, the focal UAV has higher confidence in the reward-maximizing model of this neighbor. As seen in Figure 6.9, the belief in higher model confidence grows rapidly for a neighbor moving to its goal. Similarly, the reward decreases when the neighbor moves away from the goal which decreases the belief in higher model confidence values. This is shown in Figure 6.10, where the neighbor is moving away from its goal. However, in the presence of obstacles and other moving UAVs, the neighbor might not move directly to the goal. So in Figure 6.11, we show the case when the neighbor velocity is uniformly randomly distributed and is not directed towards or away from its goal. It can be seen that the belief update is sensitive to the observed velocity of the neighbor and changes significantly with each time step. Figures 6.9, 6.10 and 6.11 illustrate that the belief distribution correctly reflects the confidence of the focal UAV in the reward model of its neighbor.

6.3 Ideal-world simulations

The proposed decentralized collision avoidance method is first verified in an ideal-world simulation. In this simulator, each UAV is represented using a point object called an agent which has no physical dimensions. Each agent in the simulator has precise information about the position and velocity of all agents and is also aware of the static obstacles in the environment. However, the agents only share goal and preference order information with each other. The future motion plan and trajectory is not shared and the motion prediction model from Chapter 3 is used to estimate the future occupancy probability. Including any noise models in the state estimation of agents and the neighbors can introduce complexity in the simulation. This complexity is undesired as the experiments aim to verify the effectiveness of motion prediction and the decentralized collision avoidance method and not the robustness to noise. The parameters used for the experiments are described in the table 6.1 below.

Variable	Value
D_0	5 m
T	9 s
R_0	0.5 m
P_{th}	0.5
γ	0.5

Table 6.1: Parameters used by the ideal simulator.

The first experiment describes a setting where two agents swap their positions in an environment without any obstacles. Figure 6.12 illustrates the snapshots from the experiment where the red dot represents agent 0 and the blue dot represents agent 1. In this experiment, agent 1 has a higher preference so agent 0 generates a new motion plan when a collision is

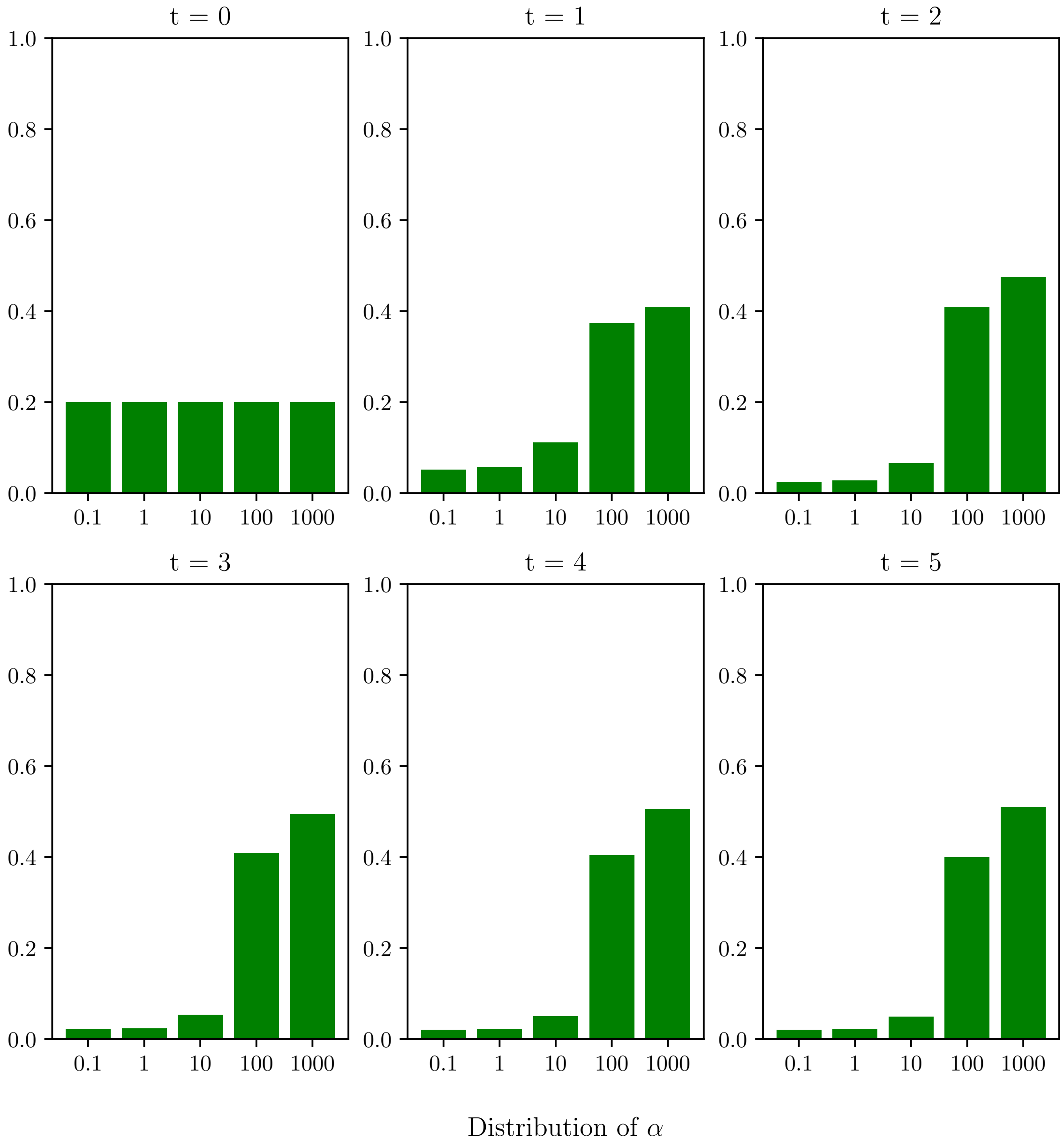


Figure 6.9: Belief distribution of α when the neighbor UAV moves towards the goal.

detected with agent 1 while agent 1 keeps moving towards its goal. Since we are interested in the occupancy probability of agent 1, Figure 6.12 shows the reachable set and belief distribution for agent 1 as estimated by agent 0. The access to precise information about the position and velocity of the neighbors makes it easier to estimate their future trajectory. As seen in Figure 6.12, the belief distribution skews towards higher model confidence within the first few seconds of the simulation. Thus, the occupancy probability distribution over the reachable set at time horizon $t = 5$ has extremely low variance. Using this occupancy probability, agent 0 plans a new collision-free path as soon as a potential collision is detected. Figure 6.12b shows this collision-free path that avoids the moving agent 1. Once the agents cross paths and no potential collisions are detected, agent 0 re-plans a shorter path to its goal (see Figure 6.12c).

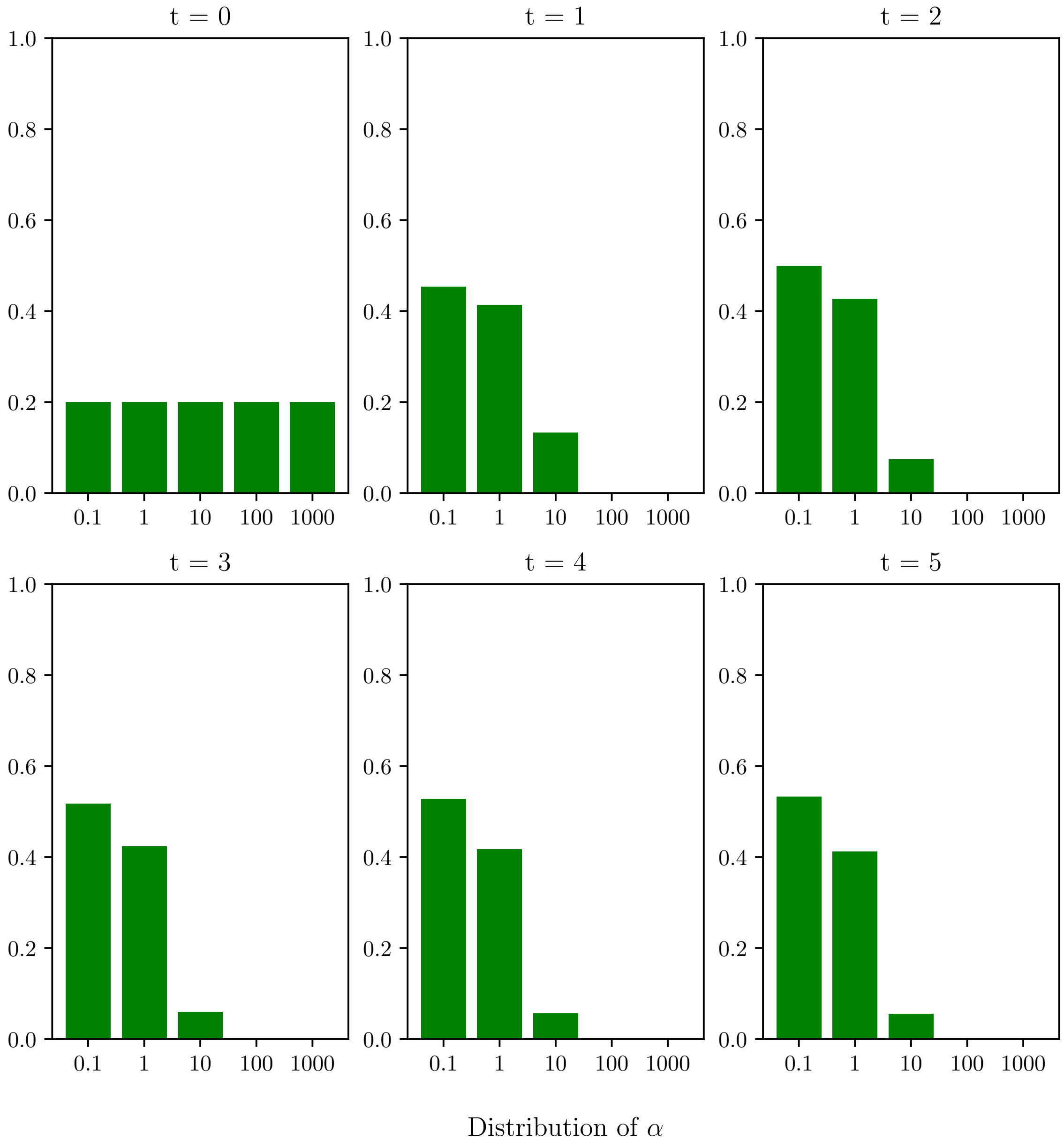


Figure 6.10: Belief distribution of α when the neighbor UAV moves away from the goal.

The second experiment shows the agents swapping their positions in the presence of obstacles. Similar to the previous experiment, agent 1 has higher preference so agent 0 re-plans the path on detecting a collision. Due to the presence of obstacles, agent 1 has a path that goes around one of the obstacles. Since the Q-function in (3.2) does not model the obstacles, the belief in high model confidence decreases when agent 1 is moving around the obstacles. As a result, the occupancy distribution in Figure 6.13b has a higher variance as compared to Figure 6.12b. However, the belief for higher confidence increases as soon as the agents cross their paths and agent 1 directly moves towards its goal. The snapshots presented in Figure 6.13 demonstrate that belief in model confidence is useful when the agents deviate from reward-maximizing model. The re-planning method presented in Chapter 4 ensures that agent 0 only avoids a small region around agent 1. As seen in Figure 6.13d, the trajectory of agent 0 does not deviate far away from the initial trajectory in fig. 6.13a.

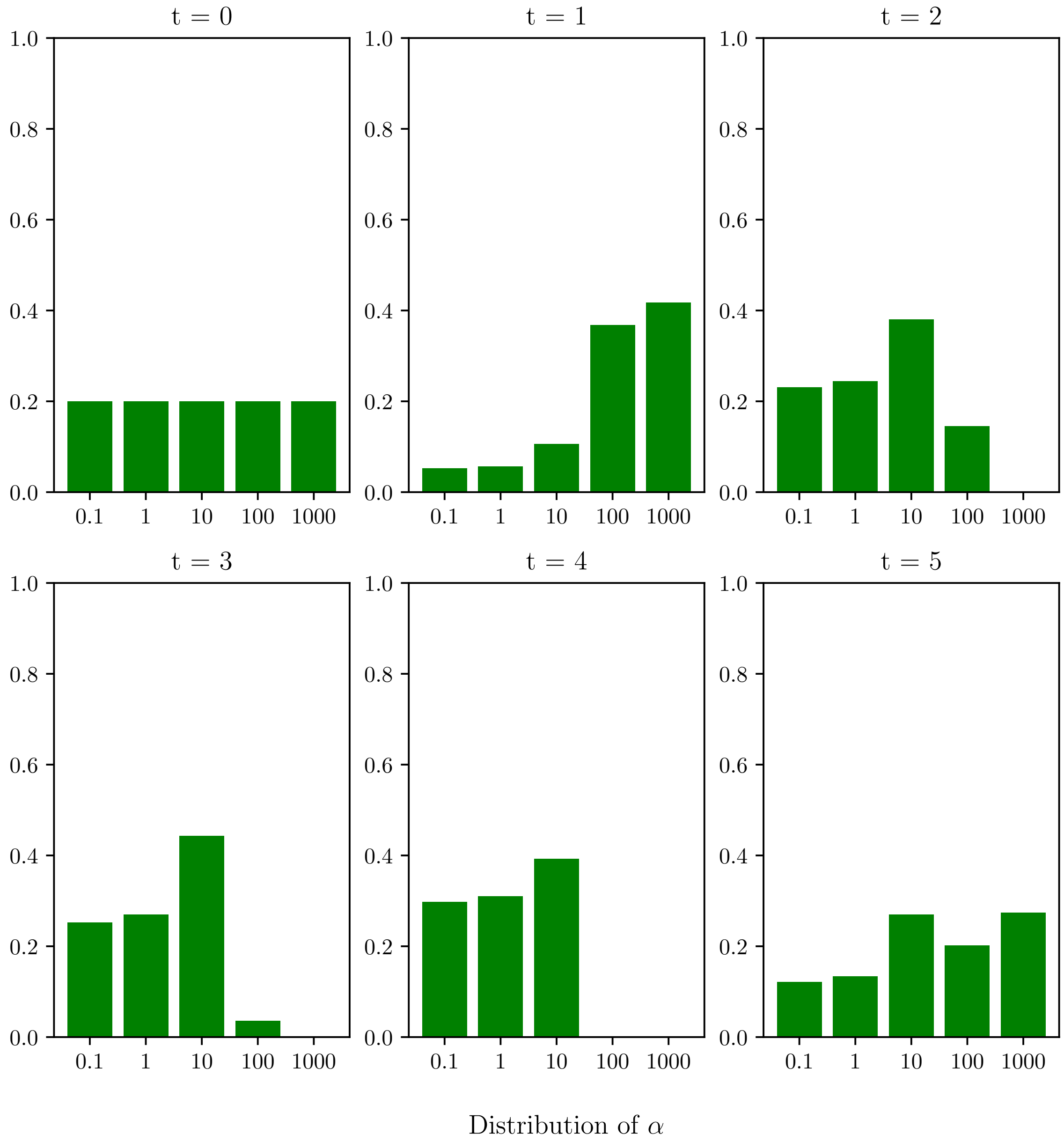
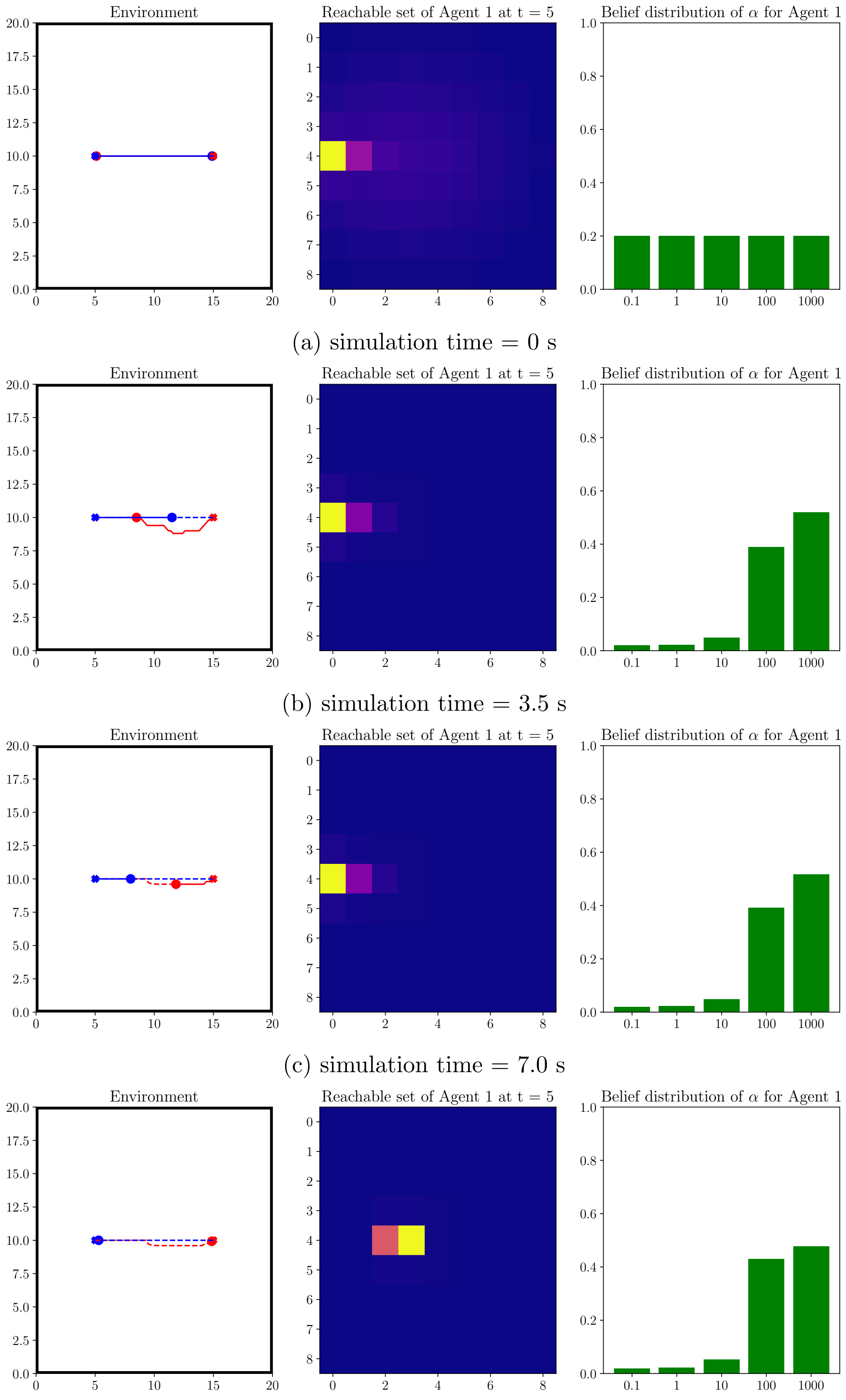


Figure 6.11: Belief distribution of α when the neighbor UAV moves randomly.

Lastly, we present an experiment with four agents and two obstacles in Figure 6.14. In this experiment, additional to agent 0 and 1, the yellow dot represents agent 2 while the green dot represents agent 3. Each agent swaps its position with the diametrically opposite one. For the sake of simplicity, we assume that agent with a higher number has higher preference than any of the agent with lower numbers. Although this preference order represents only one of the possible preference cases, it does not affect the analysis of the collision avoidance. Since agent 0 has the lowest preference, it plans a path to avoid all other agents. Figure 6.14b illustrates the collision-free paths of all the agents when they cross each other. Similar to the previous cases, the agents re-plan a path once they cross the other agents and no future collisions are detected. This experiment demonstrates the utility of the presented decentralized method when multiple agents move in a cluttered environment.



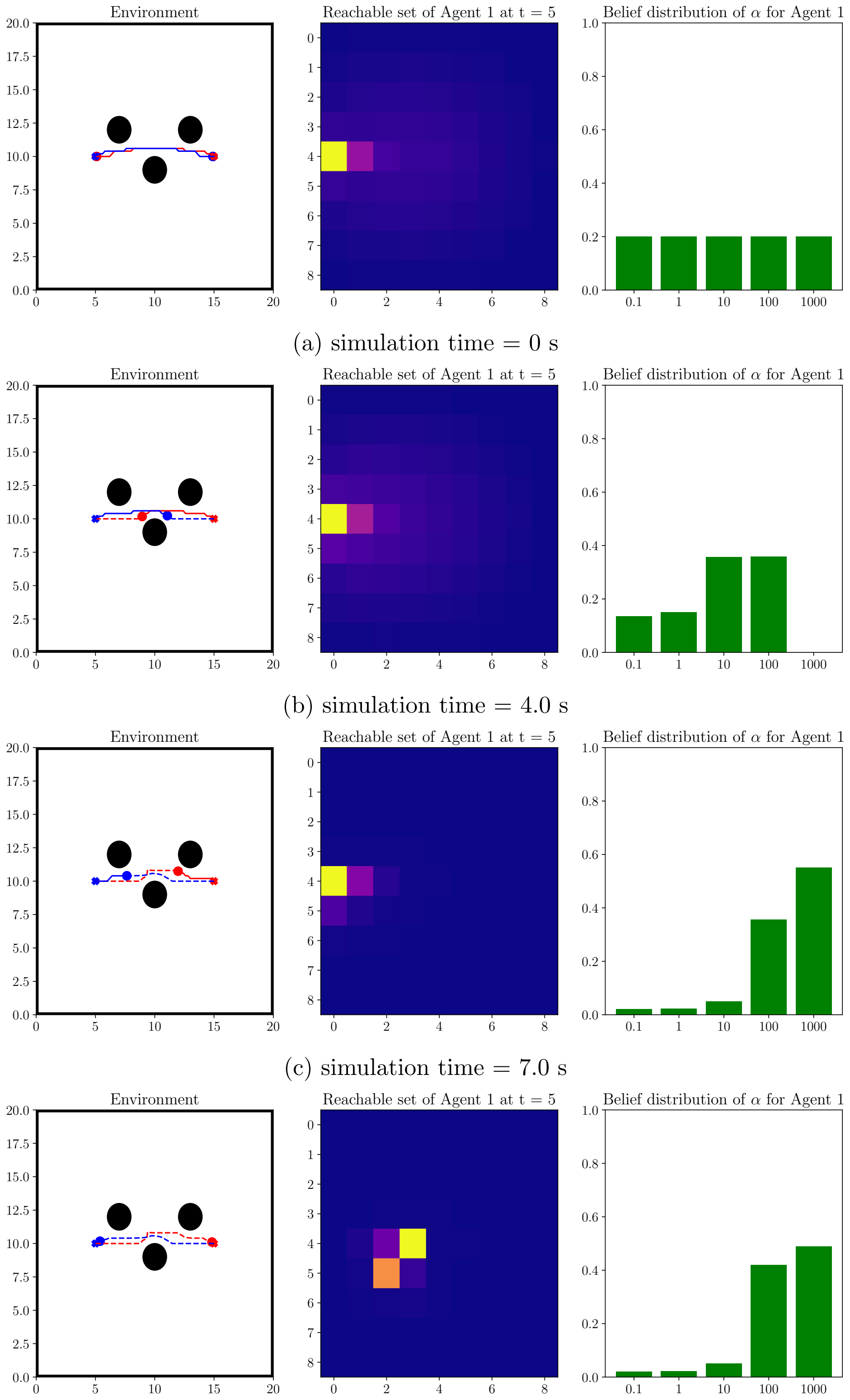
(a) simulation time = 0 s

(b) simulation time = 3.5 s

(c) simulation time = 7.0 s

(d) simulation time = 11.0 s

Figure 6.12: 2 agents swapping positions in an environment with no obstacles. The red dot represents agent 0 while the blue one represents agent 1. The path of the agents are shown in the corresponding colors.



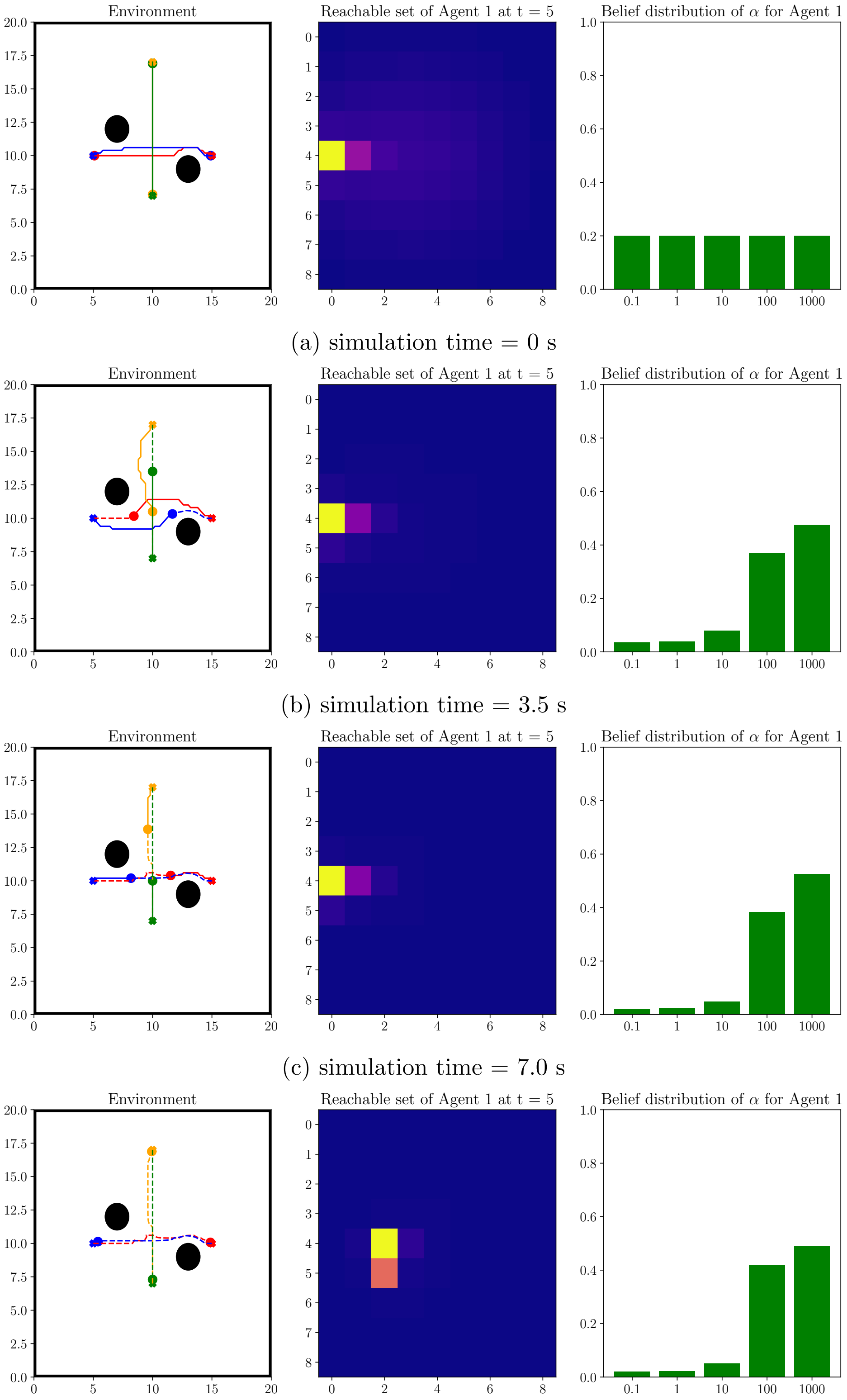
(a) simulation time = 0 s

(b) simulation time = 4.0 s

(c) simulation time = 7.0 s

(d) simulation time = 11.0 s

Figure 6.13: 2 agents swapping positions in an environment with obstacles. The red dot represents agent 0 while the blue one represents agent 1. The path of the agents are shown in the corresponding colors. The obstacles are shown by the black circles.



(a) simulation time = 0 s

(b) simulation time = 3.5 s

(c) simulation time = 7.0 s

(d) simulation time = 11.0 s

Figure 6.14: 4 agents swapping positions in an environment with obstacles. The red, blue, yellow and green dots represents agents 0, 1, 2 and 3, respectively. The path of the agents are shown in the corresponding colors. The obstacles are shown by the black circles.

6.4 Gazebo simulations

The simulations presented in this section are performed in Gazebo simulator. The method was implemented using libraries from Robot Operating System (ROS) and the MRS-system [3]. The MRS-system is a set of libraries and tools developed for fast and easy deployment of both single and multiple UAV missions in the real world. We use the MRS-system to perform Software-In-The-Loop (SITL) simulations in Gazebo for different scenarios. Gazebo simulates the UAV as a rigid body with finite dimensions which can collide with other UAVs and obstacles in the environment. The position and velocity of UAVs is simulated using an in-built noise model. Similar to the ideal simulator, the UAVs only share their goal and preference order with other UAVs and a simulated version of UVDAR is used for direct detection of UAV positions. The snapshots from the Gazebo environments are shown in Figure 6.15. We use the parameters from Table 6.2 for the experiments presented in this section.

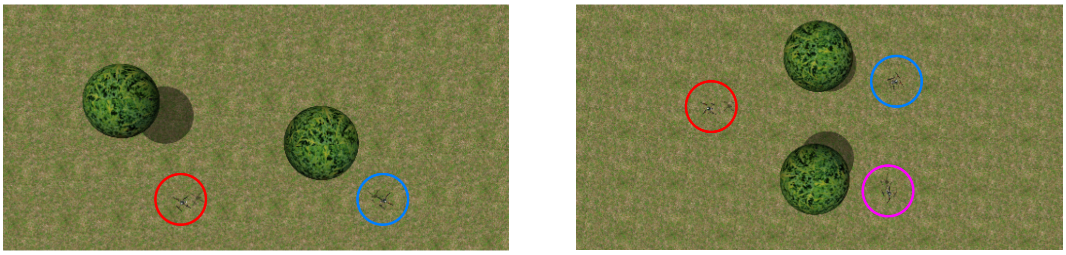


Figure 6.15: The Gazebo environments used for the 2 UAV and 3 UAV experiments.

The experiment shown in Figure 6.16 illustrates the scenario with two UAVs crossing paths with each other. The red and blue arrows represent the trajectories followed by each UAV. The UAVs start at the position marked in circles and use the motion planning method presented in Chapter 4 to plan a path. In this experiment, the blue UAV has lower preference number as compared to the red UAV. As a result, the blue UAV plans a new path when a collision is detected using the occupancy probability of the red UAV. The deviation in the trajectory of the blue UAV can be seen in Figure 6.16b. The UAVs also avoid the static obstacles in the environment when moving towards their goals. Figure. 6.16b shows the online octomap created with the obstacles, where the darker colors boxes represent the lower altitude in the map.

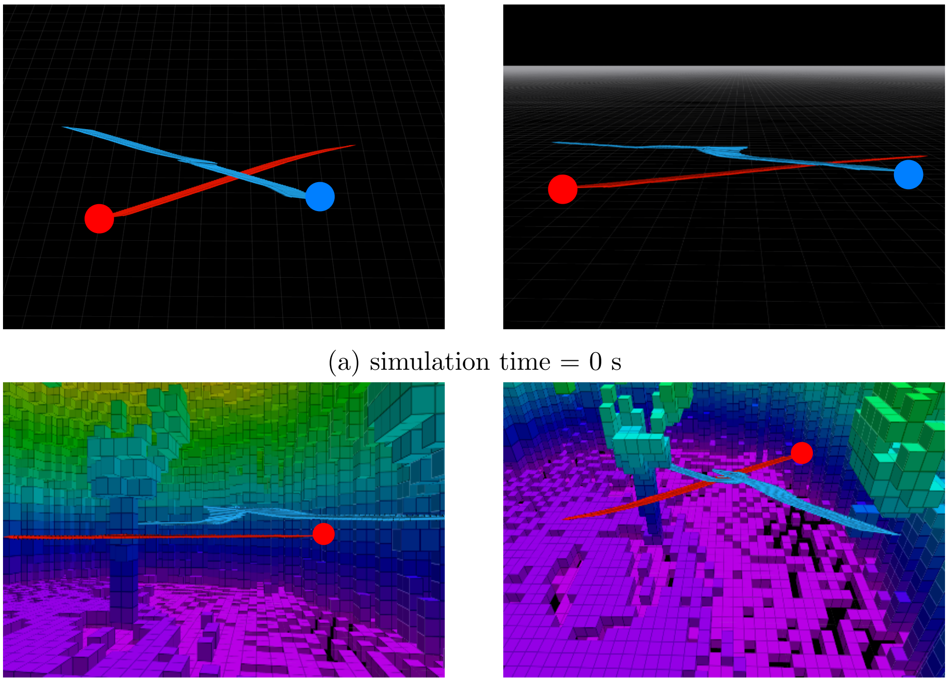
The two UAV experiment in Figure 6.16 represents a basic case of multi-UAV interaction. To test the utility of the proposed method in a more complex setting, an experiment with three independent UAVs in presented in Figure 6.17. As the preference order between the UAVs can be arbitrary, the collision avoidance can deviate the trajectory of UAVs in several different ways. In this three UAV experiment, the red UAV has the highest preference number so it does not need to re-plan its path. The blue UAV has a preference number higher than the purple UAV so it only re-plans the path when a collision is detected with the red UAV. The purple UAV has the lowest preference number and must avoid collisions with all the other moving UAVs. As seen in Figure 6.17a, the red UAV moves directly towards its goal while the blue UAV deviates from its trajectory when crossing the path of the red UAV. The red UAV is moving in a direction opposite to the blue one, so the reachable set has low occupancy probability for the points on the path of blue UAV. Thus, the blue UAV only needs to avoid a small volume when re-planning the path. The resulting trajectory of the blue UAV has a small deviation, as seen in Figure 6.17a. However, as the purple UAV is moving in the same direction as the blue one, the detected collision is with the future position of the blue UAV.

The volume occupied by the set $\mathcal{O}$, as described in (4.9), is reasonably large which leads to significant deviation in the trajectory of the purple UAV. Figure 6.17b shows the purple UAV moving below the blue UAV in order to avoid a collision.

Variable	Value
D_0	3 m
T	3 s
R_0	0.5 m
P_{th}	0.3
γ	0.5

Table 6.2: Parameters used for the Gazebo simulations.

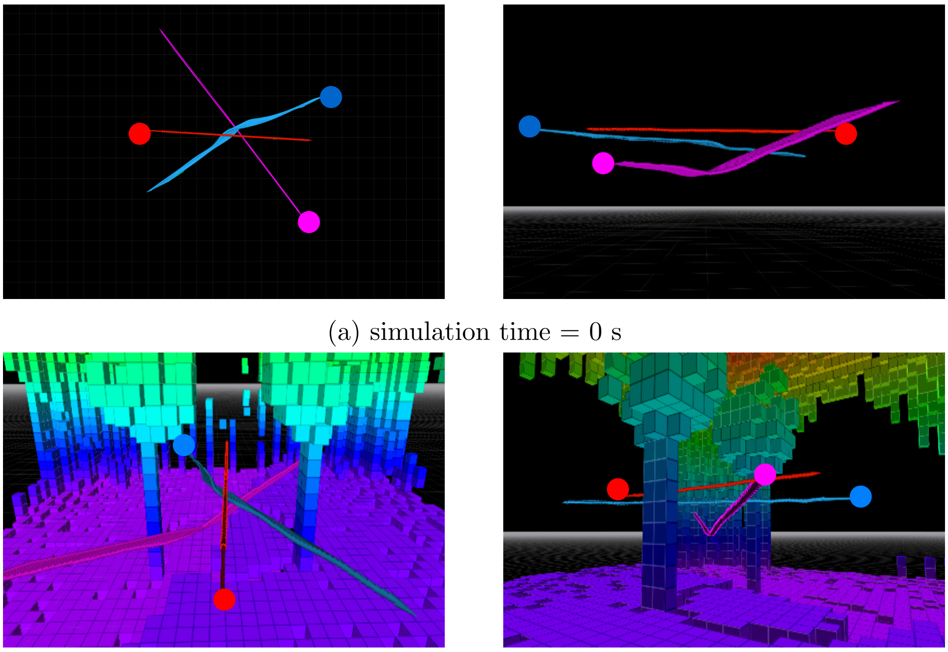
The experiments in Figures 6.16 and 6.17 demonstrate the utility of the presented decentralized motion planning method. However, due to the presence of obstacles in the environment and online map generation the motion planner can often fail to find a path in real-time. In such cases, it is often better to stop and re-plan to a modified goal position. Thus, the UAVs will not cross the paths at the same time and do not need the motion prediction for collision avoidance. The experiments in this section were designed to analyze the presented collision avoidance method so the UAVs always cross paths with each other. A comprehensive analysis of the decentralized collision avoidance can be performed in randomly generated environments with different number of UAVs. However, this analysis is not a part of the thesis and is focus of the future work.



(a) simulation time = 0 s

(b) simulation time = 0 s

Figure 6.16: 2 UAVs moving to their goals in a cluttered environment. The red and blue circles depict the starting positions of the UAVs.



(a) simulation time = 0 s

(b) simulation time = 0 s

Figure 6.17: 3 UAVs moving to their goals in a cluttered environment. The red, blue and purple circles represent the starting positions of the UAVs.

Chapter 7

Qualitative comparison

The method presented in this thesis uses an occupancy prediction model to avoid collisions with moving UAVs. The uncertainty in the prediction is explicitly modeled as a belief distribution over the model confidence. However, the methods discussed in Section 1.1 do not use a predictive model to solve the multi-UAV collision avoidance problem. As the works in [4], [9] and [6] employ a combination of different techniques to approach the collision avoidance, direct comparison of theses methods becomes a challenging task. This section presents a qualitative comparison and analysis of the presented method with several different works described earlier.

The work presented in [4] models the moving obstacles and UAVs using ellipsoids which are used to generate constraints for the trajectory optimization problem. Similarly, authors in [9] use several ellipsoids to bound the volume around the predicted trajectory of other UAVs. These ellipsoids often constraint a large volume around the UAVs which makes the optimization infeasible. [6] uses a discrete motion planner to accelerate the trajectory optimization but still models the UAVs as fixed size convex polygons. However, the occupancy prediction model in this thesis only constraints the volume around the UAV when a collision is detected with a threshold probability. This relaxes the trajectory optimization problem while still avoiding collisions.

Contrary to other methods, [2] uses probabilistic constraints for the trajectory optimization problem. These constraints explicitly model the uncertainty in the future motion of the UAVs which is similar to motion prediction presented in this thesis. However, they do not consider any obstacles in the environment which relaxes the trajectory optimization. Also, the authors do not have an explicit collision check or re-planning technique which makes it difficult to compare with the presented method.

As shown in Chapter 6, the Bayesian motion prediction presented in this thesis can effectively model the motion of moving UAVs and predict collisions. When used in a 3D environment, the Bayesian update of the positions in the discrete reachable set can become computationally expensive as compared to the technique presented in [6]. The computation can be reduced by decreasing the size of the reachable set. However, this would limit the predictions to a very small time horizon which can lead to future collisions. The computation can also be reduced by decreasing the resolution of the octomap but this leads to large constraints for the motion planning method similar to the ellipsoids in [9] and [4].

Chapter 8

Conclusion

This thesis presented a decentralized method for collision-free trajectory planning in a multi-UAV systems. The method does not depend on external infrastructure like GNSS or RTK and uses UVDAR for direct relative estimation of positions of other UAVs. The observed position is used to estimate the current position and velocity of the moving UAVs. The presented method only relies on sharing goal and preference order information to perform real-time collision avoidance. As the UAVs only use onboard sensors for localization, mapping and detecting other UAVs, the method is completely independent from any external infrastructure. The decentralized nature of the motion planning method makes the multi-UAV system robust to individual UAV failures.

The presented method uses a Bayesian model to predict the future occupancy of the region around the moving UAVs. The probability of this occupancy is updated using the current position and velocity of the moving UAV and its desired goal. The uncertainty in the motion of the UAVs is explicitly modeled using a belief distribution over the model confidence parameter. As the UAV moves towards its goal, the belief distribution is updated to reflect the similarity between the reward-maximizing model and the observed motion. The method is verified using an ideal simulator to establish the utility of the prediction method and performance of re-planning process. After the verification, the method is integrated with the MRS-system and analyzed using SITL simulations in the Gazebo simulator.

This thesis successfully completed all the points as specified in the thesis assignment.

- An extensive review of the existing methods for multi-UAV collision avoidance is presented in Chapter 1.
- The Bayesian motion prediction method is selected after identifying the strengths and weaknesses of the existing approaches.
- The presented method is integrated into the MRS-system and used for verification in the Gazebo simulator.
- Several different results are presented in an ideal simulator followed by experiments in Gazebo.
- A qualitative comparison between the presented method and the existing approaches is also presented in Chapter 7.

8.1 Future work

The presented method uses an occupancy prediction model to maintain the occupancy probability of the region around a moving UAV. Although this method reduces the constrained volume in comparison to existing methods, the Bayesian update over a discrete set of points is computationally expensive. The computational load can be reduced by selecting the size of the reachable set and resolution of the octomap. However, more analysis is needed for the selection of these parameters and is proposed as a part of the future work.

The real-time re-planning technique presented in this thesis uses a modified octomap to constraint a specific volume around the colliding UAVs. As the size of the constrained volume depends on the accuracy of the occupancy prediction, it is important to have real-time prediction updates. However, with a lag in the prediction update, the modified octomap can often constraint large regions thus making the motion planning unfeasible. It is possible to use an iterative relaxation of the constrained volume to improve the motion planner. Another idea would be to add re-planning checks to stop the UAV for some time before re-planning again. These improvement can make the presented method robust to various failures that occur during real-world deployment of a multi-UAV system.

Chapter 9

References

[1] J. Tordesillas and J. P. How, “Mader: Trajectory planner in multiagent and dynamic environments,” *IEEE Transactions on Robotics*, 2022.

[2] S. H. Arul and D. Manocha, “Swarmcco: Probabilistic reactive collision avoidance for quadrotor swarms under uncertainty,” *IEEE Robotics and Automation Letters*, 2021.

[3] T. Baca, M. Petrlik, M. Vrba, V. Spurny, R. Penicka, D. Hert, and M. Saska, “The MRS UAV System: Pushing the Frontiers of Reproducible Research, Real-world Deployment, and Education with Autonomous Unmanned Aerial Vehicles,” *Journal of Intelligent & Robotic Systems*, 2021.

[4] J. Park and H. J. Kim, “Online trajectory planning for multiple quadrotors in dynamic environments using relative safe flight corridor,” *IEEE Robotics and Automation Letters*, 2021.

[5] P. Petracek, V. Kratky, M. Petrlik, T. Baca, R. Kratochvil, and M. Saska, “Large-scale exploration of cave environments by unmanned aerial vehicles,” *IEEE Robotics and Automation Letters*, 2021.

[6] B. Senbaslar, W. Hönig, and N. Ayanian, “RLSS: real-time multi-robot trajectory replanning using linear spatial separations,” 2021. arXiv: `2103.07588`.

[7] J. Tordesillas and J. P. How, “Mader: Trajectory planner in multiagent and dynamic environments,” *IEEE Transactions on Robotics*, 2021.

[8] X. Zhou, Z. Wang, X. Wen, J. Zhu, C. Xu, and F. Gao, “Decentralized spatial-temporal trajectory planning for multicopter swarms,” 2021. arXiv: `2106.12481`.

[9] S. H. Arul and D. Manocha, “Dcad: Decentralized collision avoidance with dynamics constraints for agile quadrotor swarms,” *IEEE Robotics and Automation Letters*, 2020.

[10] D. Fridovich-Keil, A. Bajcsy, J. F. Fisac, S. L. Herbert, S. Wang, A. D. Dragan, and C. J. Tomlin, “Confidence-aware motion prediction for real-time collision avoidance1,” *The International Journal of Robotics Research*, 2020.

[11] C. E. Luis, M. Vukosavljev, and A. P. Schoellig, “Online trajectory generation with distributed model predictive control for multi-robot motion planning,” *IEEE Robotics and Automation Letters*, 2020.

[12] P. Petráček, V. Krátký, and M. Saska, “Dronument: System for Reliable Deployment of Micro Aerial Vehicles in Dark Areas of Large Historical Monuments,” *IEEE RAL*, 2020.

[13] M. Petrlík, T. Báča, D. Heřt, M. Vrba, *et al.*, “A Robust UAV System for Operations in a Constrained Environment,” *IEEE RAL*, 2020.

[14] J. Tordesillas and J. P. How, “Minvo basis: Finding simplexes with minimum volume enclosing polynomial curves,” 2020. arXiv: `2010.10726`.

[15] M. Vrba and M. Saska, “Marker-less micro aerial vehicle detection and localization using convolutional neural networks,” *IEEE Robotics and Automation Letters*, 2020.

[16] E. T. Alotaibi, S. S. Alqefari, and A. Koubaa, “Lsar: Multi-uav collaboration for search and rescue missions,” *IEEE Access*, 2019.

[17] A. Bajcsy, S. L. Herbert, D. Fridovich-Keil, J. F. Fisac, S. Deglurkar, A. D. Dragan, and C. J. Tomlin, “A scalable framework for real-time multi-robot, multi-human collision avoidance,” in *2019 International Conference on Robotics and Automation (ICRA)*, 2019.

[18] E. Lygouras, N. Santavas, A. Taitzoglou, K. Tarchanidis, A. Mitropoulos, and A. Gasteratos, "Unsupervised human detection with an embedded vision system on a fully autonomous uav for search and rescue operations," *Sensors*, 2019.

[19] H. V. Nguyen, M. Chesser, L. P. Koh, S. H. Rezatofighi, and D. C. Ranasinghe, "Trackerbots: Autonomous unmanned aerial vehicle for real-time localization and tracking of multiple radio-tagged animals," *Journal of Field Robotics*, 2019.

[20] V. Walter, N. Staub, A. Franchi, and M. Saska, "Uvdar system for visual relative localization with application to leader–follower formations of multirotor uavs," *IEEE Robotics and Automation Letters*, 2019.

[21] R. D. Arnold *et al.*, "Search and rescue with autonomous flying robots through behavior-based cooperative intelligence," *Journal of International Humanitarian Action*, 2018.

[22] W. Hönig, J. A. Preiss, T. K. S. Kumar, G. S. Sukhatme, and N. Ayanian, "Trajectory planning for quadrotor swarms," *IEEE Transactions on Robotics*, 2018.

[23] D. R. Robinson, R. T. Mar, K. Estabridis, and G. Hewer, "An efficient algorithm for optimal trajectory generation for heterogeneous multi-agent systems in non-convex environments," *IEEE Robotics and Automation Letters*, 2018.

[24] V. Walter, N. Staub, M. Saska, and A. Franchi, "Mutual localization of uavs based on blinking ultraviolet markers and 3d time-position hough transform," in *IEEE 14th International Conference on Automation Science and Engineering (CASE)*, 2018.

[25] K. A. Ghamry, M. A. Kamel, and Y. Zhang, "Multiple uavs in forest fire fighting mission using particle swarm optimization," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2017.

[26] J. Chen, T. Liu, and S. Shen, "Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[27] H. Qin, J. Q. Cui, J. Li, Y. Bi, M. Lan, M. Shan, W. Liu, K. Wang, F. Lin, Y. F. Zhang, and B. M. Chen, "Design and implementation of an unmanned aerial vehicle for autonomous firefighting missions," in *12th IEEE International Conference on Control and Automation (ICCA)*, 2016.

[28] S. Tang and V. Kumar, "Safe and complete trajectory generation for robot teams with higher-order dynamics," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[29] S. Ward, J. Hensler, B. Alsalam, and L. F. Gonzalez, "Autonomous uavs wildlife detection using thermal imaging, predictive navigation and computer vision," in *IEEE Aerospace Conference*, 2016.

[30] K. Zhang, J. Chen, Y. Chang, and Y. Shi, "Ekf-based lqr tracking control of a quadrotor helicopter subject to uncertainties," in *42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016.

[31] G. Bevacqua, J. Cacace, A. Finzi, and V. Lippiello, "Mixed-initiative planning and execution for multiple drones in search and rescue missions," in *Proceedings of the International Conference on Automated Planning and Scheduling*, 2015.

[32] Y. Chen, M. Cutler, and J. P. How, "Decoupled multiagent path planning via incremental sequential convex programming," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[33] D. Câmara, "Cavalry to the rescue: Drones fleet to help rescuers operations over disasters scenarios," in *IEEE Conference on Antenna Measurements & Applications (CAMA)*, 2014.

[34] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, 2013.

[35] A. Chamseddine, Y. Zhang, and C. A. Rabbath, "Trajectory planning and re-planning for fault tolerant formation flight control of quadrotor unmanned aerial vehicles," in *American Control Conference (ACC)*, 2012.

[36] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, “Maximum entropy inverse reinforcement learning,” ser. AAAI Conference on Artificial Intelligence, 2008.

[37] N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2004.

[38] A. Y. Ng and S. J. Russell, “Algorithms for inverse reinforcement learning,” ser. International Conference on Machine Learning (ICML), 2000.

[39] D. J. Meagher, *Octree encoding: A new technique for the representation, manipulation and display of arbitrary 3-d objects by computer*. Electrical and Systems Engineering Department Rensseiaer Polytechnic, 1980.

[40] P. E. Hart, N. J. Nilsson, and B. Raphael, “A formal basis for the heuristic determination of minimum cost paths,” *IEEE transactions on Systems Science and Cybernetics*, 1968.