UNIVERSITY OF CALIFORNIA
SANTA CRUZ

SCALABLE APPROACHES TO DUBINS VEHICLE NAVIGATION PROBLEMS UNDER UNCERTAINTY

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER ENGINEERING
with an emphasis in ROBOTICS AND CONTROL

by

Alexey Munishkin

December 2022

The Dissertation of Alexey Munishkin
is approved:

Professor Dejan Milutinović, Chair

Professor Ricardo Sanfelice

David W. Casbeer, Ph.D.

Peter Biehl
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

Abstract

Scalable approaches to Dubins vehicle navigation problems under uncertainty

by

Alexey Munishkin

The environment around an autonomously navigated vehicle can have an unpredictable number of other vehicles and stationary or moving obstacles that may or may not have harmful intentions. The safe navigation of the autonomous vehicle in the presence of other vehicles and obstacles can be formulated as a stochastic optimal control problem. While in theory one can write down the corresponding Hamilton-Jacobi-Bellman (HJB) equation for any state space control problem, practically solving the equation is computationally infeasible when the state space is large. Moreover, once it is accounted for a time varying number of obstacles and other vehicles, and the associated time varying dimension of the state space, it is clear that new approaches to the design of vehicle navigation have to be considered. This work addresses the problem of autonomous navigation by a scalable integration of stochastic optimal control solutions to problems such as vehicle-to-vehicle, vehicle-to-obstacle, or vehicle-to-goal problems. The scalable navigation means that the autonomous vehicle or team of vehicles can navigate toward their goals while coping with a large number of other vehicles, or obstacles in their proximity. The work is based on the Dubins nonholonomic vehicle model and is illustrated by multiple scenarios in simulations and with real robots.

Acknowledgments

I would like to thank everyone who has ever helped me. My friends and family, especially my mother and dear friends Marco Carmona, Samuel Mansfield and Jeremy Coupe. Especially to my mentor Dejan Milutinović and co-authors David Casbeer and Araz Hashemi from the Air Force who have allowed me to work on this exciting field of research. Finally I would like to thank NASA Ames and my branch chief Shannon Zelinski in the Aviation Systems Division for supporting my graduate studies and allowing me to work on exciting projects in urban air mobility over the summer that have inspired my graduate research work.

This research was partially supported by U.S. Department of Defense (Grant No. FA8650-15-D-2516), UCSC Computer Science and Engineering Fall 2021 Sabbatical Fellowship, and 2019-2020 Piatt Fellowship.

A Note on Previously Published Material

The text of this dissertation includes reprints of the following previously material:

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. "Stochastic optimal control navigation with the avoidance of unsafe configurations." In 2016 international conference on unmanned aircraft systems (ICUAS), pp. 211-218. IEEE, 2016.

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. "Safe navigation with the collision avoidance of a brownian motion obstacle." In Dynamic

Systems and Control Conference, vol. 58295, p. V003T39A009. American Society of Mechanical Engineers, 2017.

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. "Time Efficient Inspection of Ground Vehicles by a UAV Team Using a Markov Inequality Based Rule." In Distributed Autonomous Robotic Systems, pp. 95-108. Springer, Cham, 2019. (**one of three papers nominated for the best student paper award**)

- Munishkin, Alexey A., Araz Hashemi, David W. Casbeer, and Dejan Milutinović. "Scalable markov chain approximation for a safe intercept navigation in the presence of multiple vehicles." Autonomous Robots 43, no. 3 (2019): 575-588.

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. "Navigation with Multi-obstacle Avoidance Composed of Stochastic Optimal Controllers." In 2019 American Control Conference (ACC), pp. 2239-2244. IEEE, 2019.

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. "Min–max time efficient inspection of ground vehicles by a UAV team." Robotics and Autonomous Systems 125 (2020): 103370.

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. "Scalable Navigation for Tracking a Cooperative Unpredictably Moving Target in an Urban Environment." In 2022 control conference on control technology and applications

(CCTA)

- Dejan Milutinović, Munishkin, Alexey A., and David W. Casbeer. ”A Safe Stochastic Optimal Feedback Control Approach to Autonomous Navigation with a Large Number of Obstacles.” Journal of Intelligent & Robotic Systems (submitted in 2022)

With regards to the authors of these reprints, the co-author Dejan Milutinović (dmilutin [at] ucsc.edu), David W. Casbeer (david.casbeer [at] us.af.mil) listed in these publications directed and supervised the research contained within this dissertation. Co-author Araz Hashemi has been instrumental in developing the one-step method, which was the basis for the later developed multi-step method.

Chapter 1

Introduction

1.1 Motivation

With the increased use of mobile robots and autonomous vehicles, there is a pressing need to study control algorithms that improve the autonomous capabilities of these vehicles. In the context of this work that means (1) navigating while taking into account uncertainty, in the trajectories of other vehicles and obstacles, (2) performing maneuvers to avoid ”unsafe” configurations in which damage from another vehicle or obstacle could occur, (3) reaching ”beneficial” or end goal configurations in time optimal (or near optimal) manner, and (4) coping with dynamic multi-agent environments where the number of agents varies in time. For improvement of autonomous capabilities of the multi-agent vehicle scenario discussed above, the following two questions

(Q1) How to navigate our agent in the presence of a moving target or an obstacle under uncertainty?

(Q2) How can we compose the navigation solutions from (Q1) that works in a dynamic multi-agent environment full of time-varying number of obstacles and other agents?

The questions (Q1) and (Q2) serve as a basis for our study of safe, scalable navigation algorithms for a Dubin vehicle [47], which is a non-holonomic kinematic model that can be used to approximate motions of cars, ships and airplanes. Since we know the kinematics and goal for the navigation of our vehicle, but we don't know the intent or motion of other vehicles and obstacles around our vehicle, we use stochastic optimal control to design feedback control navigation strategies to account for these uncertainties. The safety aspect of the design is to account for the implementations with real vehicles or robots and avoid collisions that can result in their damage. Scalable aspect of the navigation algorithms is a final and perhaps most important factor for navigation in multi-agent systems. Scalable algorithms that are able to handle changing multi-agent enviroment with time varying number of agents and possibly new other features in an autonomous or semi-autonomous multi-agent environment are the main driving forces behind the novelty and potential applications of this work. To demonstrate further the potential impact of this work, we will also deploy our algorithms on real robots that run in real-time executing the safe and scalable navigation algorithms.

1.2 Thesis Contributions

The research presented in this thesis addresses the problem of finding control solutions to nonlinear time-varying stochastic differential equation systems [129, 111] for unmanned fixed-wing aerial vehicles (UAVs). UAVs are used in various applications in agriculture, environmental monitoring, surveillance, and in new coming applications such as package delivery, wildfire monitoring and prevention, and transportation services [13].

We use stochastic optimal control solution approaches for one-on-one solutions of vehicle-to-vehicle, vehicle-to-obstacle, or vehicle-to-goal problems. These one-on-one solutions are composed in various ways using the Markov inequality rule for switching among optimal one-on-one solutions, or creating an approximate value function for the overall stochastic system. This composed solution using one-on-one solutions is scalable so that it can cope with a large number of other vehicles, or obstacles in proximity of our controlled vehicle. We have also implemented these scalable control solutions on real-time robotic systems.

1.3 Outline

This dissertation is organized as follows. Chapter 2 summarizes the state of available literature. Chapter 3 introduces some of the preliminary modeling, mathematical

concepts, and tools used in this dissertation. In doing so, we also present a road map of the remainder of the dissertation. Chapters 4- 11 consist of the appended papers detailing research results. Brief summaries of each of these papers appear below. Chapter 12 concludes the dissertation with a summary of discussion and directions for future research.

- **Chapter 4: Stochastic Optimal Control Navigation with the Avoidance of Unsafe Configurations**

  Long-time planning horizons are required to safely navigate one vehicle in the presence of another, possibly non-cooperative vehicle. They give rise to computational issues preventing the real-time implementation of safe navigation algorithms. In this paper, we consider two nonholonomic vehicles, of which one (blue) has the goal to enter the "tail" of the other (red). Neither the goal nor the navigation strategy of the red vehicle is known by the blue vehicle. To anticipate this uncertainty, the blue vehicle uses infinite horizon stochastic optimal control. Using the stochastic optimal control and backward Kolmogorov equation, the blue is navigated to avoid unsafe configurations from which the red can enter the "tail" of the blue and gain advantage over it. Our results are illustrated by numerical simulations and the feasibility of the control for the real-time implementation is tested with small-scale robot experiments.

- **Chapter 5: Safe Navigation with Collision Avoidance of a Brownian Motion Obstacle**

  This paper develops a control that combines deterministic and stochastic optimal control solutions to the problem of safe navigation around a spherical obstacle in order to reach a waypoint location. The solution for navigation towards the waypoint is based on the deterministic minimum time optimal control. Since the intent of the obstacle is unknown to the navigating vehicle, the vehicle anticipates this uncertainty and uses a stochastic optimal control for navigation around the obstacle. The two navigation solutions are combined based on their value functions. Results are illustrated by numerical simulations.

- **Chapter 6: Time Efficient Inspection of Ground Vehicles by a UAV Team Using a Markov Inequality Based Rule**

  We present a control design for N unmanned aerial vehicles (UAVs) tasked with a time efficient inspection of M ground moving vehicles. The navigation and intent of each ground vehicle are unknown, therefore, the uncertainty of its navigation has to be anticipated in the navigation of each UAV. We use the minimum time stochastic optimal control to navigate each UAV towards the inspection of ground vehicles. Based on this control, we formulate assignments of ground vehicles to be inspected by UAVs as an optimization problem to inspect all ground vehicles in the minimum expected time. Accounting for ground vehicle uncertain trajec-

tories, we update the optimal assignment by a Markov inequality rule. The rule prevents the possibility of indefinite updating of assignments without finishing the inspection of all vehicles. On the other hand, it updates an assignment if it leads to a statistically significant improvement of the expected time of inspection. The presented approach is illustrated by a numerical example.

- **Chapter 7: Scalable Markov Chain Approximation for a Safe Intercept Navigation in the Presence of Multiple Vehicles**

  This paper studies a safe intercept navigation which accounts for the uncertainty of other vehicles' trajectories, avoids collisions and any other positions in which vehicle safety is compromised. Since the number of vehicles can vary with time, it is important that the navigation strategy can quickly adjust to the current number of vehicles, i.e, that it scales well with the number of vehicles.

  The scalable strategy is based on a stochastic optimal control problem formulation of safe navigation in the presence of a single vehicle, denoted as the one-on-one vehicle problem. It is shown that safe navigation in the presence of multiple vehicles can be solved exactly as an auxiliary Markov decision problem. This allows us to approximate the solution based on the one-on-one vehicle optimal control solution and achieve scalable navigation. Our work is illustrated by a numerical example of safely navigating a vehicle in the presence of four other vehicles and by a robot experiment.

- **Chapter 8: Stochastic Optimal Control Approach to Navigation with Multi-Obstacle Avoidance**

  Collision avoidance for unmanned aerial vehicles (UAVs) and mobile robots is a fundamental property for these systems to be considered safe, i.e. collision-free while they navigate towards a waypoint. Modeling these systems as a Dubins vehicle navigating in the presence of many potentially unknown moving circular obstacles is considered in this work. Since neither the intent or motion of these unknown obstacles is known, we develop a scalable navigation strategy by combining the value function associated with the stochastic shortest path optimal control problem to reaching a waypoint in minimum time and reaching a safe configuration from each obstacle in minimum time. Results are illustrated though numerical simulations.

- **Chapter 9: Min-Max Time Efficient Inspection of Ground Vehicles by a UAV Team**

  We present a control design for N unmanned aerial vehicles (UAVs) tasked with an inspection of M ground moving vehicles. Though the location of each ground vehicle is known to each UAV, the navigation and intent of each ground vehicle are unknown, therefore, this uncertainty has to be anticipated in each UAV's navigation. We use the minimum time stochastic optimal control to navigate each UAV towards the inspection of each ground vehicle. Based on this con-

trol, we formulate assignments of ground vehicles to be inspected by UAVs as an optimization problem to inspect all ground vehicles in minimum expected time. Accounting for ground vehicle uncertain trajectories, we update the optimal assignment by a Markov inequality rule. The rule prevents the possibility of indefinite updating of assignments without finishing the inspection of all vehicles. On the other hand, it updates an assignment if it leads to a statistically significant improvement of the inspection's expected time. The presented approach is illustrated with numerical examples.

- **Chapter 10: Scalable Navigation for Tracking a Cooperative Unpredictably Moving Target in an Urban Environment**

  Target tracking in urban environments using a fixed-wing unmanned aerial vehicle (UAV) is challenging due to the line of sight obstructions which are caused by buildings. Even with a cooperative target that sends out its location to the UAV, the vehicle may inevitably lose the line of sight due to its limited turning rate. Parts of the UAV operating space in which the UAV loses the line of sight are denoted in this paper as *shadows*. The shadows have complex shapes and move as the target changes its relative position to buildings. Avoiding the shadows increases the observation time while tracking the cooperative target. We present here a scalable feedback control approach for target tracking with shadow avoidance, which is based on a stochastic optimal feedback control solution. Our

results are illustrated by numerical simulations.

- **Chapter 11: A Safe Stochastic Optimal Feedback Control Approach to Autonomous Navigation with a Large Number of Obstacles**

  This paper develops a novel approach to safe and scalable stochastic optimal control for a short-time navigation of an unmanned fixed-wing aerial vehicle in the presence of multiple moving obstacles under uncertainty. We first solve a stochastic optimal control problem for the vehicle navigation in the presence of a single randomly moving obstacle. We show that the safety of this controller can be analyzed based on the concept of barrier function and that it has an exact probabilistic measure for obstacle avoidance. In the presence of multiple obstacles, we exploit this single obstacle stochastic optimal control solution in a multi-step optimization method. The proposed method allows the navigation that can handle a large number of obstacles and computes an indicator when the solution to navigate around obstacles is not found.

Chapter 2

Related works

The problem of safe navigation can be seen as the problem of collision-free navigation in a cluttered environment [77], with uncertainty, we use stochastic processes in the description of moving obstacles and vehicles. Alternative approaches to the problem are to use results of game theory. In addition to related game theory literature we will also review barrier functions, feedback-based navigation, and obstacle avoidance.

Game Theory: The study of navigation against threats created by other vehicles is tightly interwoven with the development of game theory [82] and the two-target game problem [15], [63, 64]. The game includes two vehicles that navigate around each other until one of the vehicles, the winner of the game, enters its target set. A stochastic variant of such two-target games is considered in [192] and [193]. These and other earlier

theoretical works have been surveyed in [69]. Other lines of work have been focused on real world implementations [50, 84, 115, 181]. Work on extending or solving N-target vehicles has been done in [60, 183].

In these earlier works on two-targets, the two vehicles are opponents and the target sets are placed at position from which the winner can harm the other vehicle. The safe navigation inspired by the game theory would be based on the assumption that every vehicle in the surrounding is considered as an adversary. This is the worst case scenario, and it would result in conservative navigation strategies, since in many application the other vehicles are not opponents. For this reason we model the motion of other vehicles as unpredictable and propose a stochastic approach to safe navigation. Safety is addressed by a predefined or computed avoidance set. Computation of the avoidance set is closely related to computations of reachable and unreachable sets associated with games of kind [81]. In the deterministic problems the boundaries between reachable and unreachable state space regions are sharp and avoidance set can also be computed using the deterministic optimal control approach [79]. However, in stochastic problems, reachable and unreachable regions are defined in terms of probability and without sharp boundaries [171].

Extending the game theory or optimal control solutions to multiple agents is difficult because of the so-called curse-of-dimensionality [138], due to the number of agents and associated number or variables necessary to describe their positions. In the discrete

domain, there has been extensive work in solving game theory problems on graphs [2], which are traditionally called cops and robbers, and in [180] a scalable solution of the game was proposed. For the continuous domain, [143] provides a framework for combining the kinematic models of various pursuing agents for a real-time implementation of a chase and search problem. In [45], a hierarchical game extension with a finite time look ahead to the stochastic setting has been proposed, while in [54], a game theory problem is partitioned into smaller problems that are then solved separately, and the solution of the original problem is determined as the lower bound of the smaller problems. Extending two-target game theoretic pursuit-evader problem has been done in [60, 183], and some work in the stochastic setting is in [104]. Most of these works didn't consider the two target problem in a multi-vehicle scenario and recent work in N-target problem considers the multi-vehicle setting in a game theoretic setting and using machine learning and or reinforcement learning approaches [177].

Lyapunov Barrier Functions: The study of safe navigation is typically investigated using barrier functions [22, 141, 186, 131]. These barrier functions are constructed as analytical Lyapunov functions [186]. The main idea behind barrier functions is that the vehicle's safety is guaranteed so long as the vehicle state-space variables do not enter any unsafe regions in the state-space domain. Extending to the stochastic case was done in [141] where one needs to take the expectation of the Lyapunov barrier function. In

the stochastic setting it can only be shown that in expectation or with certain probability that the vehicle remains safe in certain regions of the state-space. Since barrier functions are related to Lyapunov functions, safety has also been considered in the context of using vectored Lyapunov functions [99] or control vectored Lyapunov functions [127]. Our work uses stochastic optimal control which when solved via dynamic programming gives so-called value function [94, 57]. The value function provides many similar properties to Lyapunov barrier functions through so-called Viscosity solutions [56] such as uniqueness and existence of a solution.

Feedback-based navigation: Non path-planning, i.e., feedback-based navigations are frequently found in a line of work called sense and avoid [13]. They also appear in the context of pursuit-evasion games [81]. Scalable algorithms for multi-pursuers/multi-evader games have been considered in [54, 182, 62, 102]. Other non game theory feedback-based approaches use reachable sets [122, 1, 170], potential fields [178, 70], harmonic functions [112], Lyapunov functions [87], machine learning or reinforcement learning [53, 103] and others [191, 72, 200]. Our work can be considered as another example of feedback-based navigation.

Obstacle avoidance: Obstacle avoidance is encountered in game theory applications [190, 52], sense and avoid applications [13], and path planning applications [100].

Many feedback-based navigation methods such as barrier functions [30], reachable sets [1, 122], potential functions [178, 110], machine learning or reinforcement learning [199, 132], and some others [184, 200, 68] are used in the design of obstacle avoidance controllers. Since our work can be considered as another example of feedback-based navigation, it is another example of feedback-based approach to obstacle avoidance.

Chapter 3

Technical Preliminaries

This chapter briefly describes mathematical concepts that we use in Chapters 4-9. With this we highlight the complexity of solving the optimal control problems for realistic UAV scenarios although a well formulated optimal control solution formulation exists. Consequently this chapter illustrates the need for the solution approaches developed in this thesis.

Vehicle and Obstacle Models: In this work we study the navigation of a Dubins vehicle which has the following kinematics model

$$dx = v\cos\theta dt \tag{3.1}$$

$$dy = v\sin\theta dt \tag{3.2}$$

$$d\theta = udt \tag{3.3}$$

where x and y are the vehicle position coordinates and u is the control variable. $u \in [-u_{max}, u_{max}]$, $u_{max} < \infty$. The control variable u is the turning heading rate and is the only variable we control since the vehicle forward velocity $v > 0$ is constant. In the expression above t and d denote infinitesimally small increments of variables. This way of writing differential equations is common in the literature on stochastic differential equations [111, 129].

For other Dubins vehicles in the environment for which we do not know intent for their navigation we assume a stochastic model, which is

$$dx = v\cos\theta dt \tag{3.4}$$

$$dy = v\sin\theta dt \tag{3.5}$$

$$d\theta = \sigma_R dw_R \tag{3.6}$$

in which the first two expressions are the same as (3.1) and (3.2) while the third one describes the heading angle kinematics using a stochastic process. In expression (3.3) dw_R is unit intensity Wiener process and $\sigma_R > 0$ is a scaling factor. According to the model (3.4)-(3.6), for an infinitesimal time-step dt, the increment of the heading angle is $\theta(t+dt) - \theta(t)$ and has a normal distribution with mean zero and variance $\sigma_R^2 dt$. Therefore, the parameter σ_R describes the agility of the vehicle's change in the heading angle.

In the case of obstacles we always assume the obstacles are non-stationary. To account for the lack of knowledge about obstacle motion, the motion of each obstacle

is modeled as a 2D Brownian motion

$$dx_o = \sigma_o dw_x \tag{3.7}$$

$$dy_o = \sigma_o dw_y \tag{3.8}$$

where the obstacle position is given by x_o, y_o, and dw_x and dw_y denote increments of two mutually independent unit intensity Wiener processes, one along x- and the other along y-direction. In our model for obstacles, the positive scaling parameter $\sigma_o > 0$ is identical for all obstacles.

Note on Brockett's nonholonomic controllability condition: Our vehicle model (3.1)-(3.3) is a kinematics model of nonholonomic vehicle and is subject to Brockett's nonholonomic controllability condition [118]. In addition our differential system models are stochastic and have interesting properties associated with them [56, 111, 129]. Brockett also took this into account [33] and noted that discontinuous piece-wise parametric controls will stabilize stochastic systems. Controlled Markov processes include the set of discontinuous piece-wise parametric controls in feedback fashion to grantee stability and controllability of stochastic systems [56].

Controlled Markov Processes: This section is an abbreviated overview of controlled Markov processes, a more mathematical rigorous treatment is covered in Chapters 3 and 4 in [56]. In this work we consider Itô stochastic differential equations [129] in the

form

$$dx = f(s,x(s),u(s))ds + \sigma dw(s) \tag{3.9}$$

where state-space is $x \in \mathbb{R}^n$, positive time is $s \in \mathbb{R}_+$, and control is $u \in \mathbb{R}^m$. The nonlinear function $f : \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and parameter $\sigma \in \mathbb{R}^n$ satisfy the conditions for uniqueness and existence for stochastic differential equations [56]. Function f can also be a multivariate function which is composed of multiple f_i where each $f_i : \mathbb{R}_+ \times \mathbb{R}^n \to \mathbb{R}^n$ can be unique for each controller u_i.

The minimum expected cost function is

$$J = E_{tx}\left\{ \psi(x(t_1)) + \int_t^{t_1} L(s,x(s),u(s))ds \right\} \tag{3.10}$$

where L is the so-called running cost function and ψ is so-called terminal cost function. The time horizon in general is finite, however for infinite horizon problems $t_1 \to \infty$, a discounting function $e^{-\beta(s)}$ is needed and multiplied to L in order to guarantee solution convergence and obtain stochastic analogs of uniqueness and existence [46, 129, 144].

The optimal control u^* to cost function (3.10) can be computed as the steady state of the Hamilton-Jacobi-Bellman (HJB) partial differential equation

$$-\frac{\partial V}{\partial t} = \min_u \{A^{\underline{u}} V + L(t,x,u)\} \tag{3.11}$$

where V is the so-called value function with boundary conditions

$$V(x(t_1)) = \psi(x(t_1)) \tag{3.12}$$

and differential operator

$$A^{\underline{u}}V = \frac{\partial V}{\partial t} + \frac{1}{2}\sum_{i,j=1}^{n} a_{i,j}(t,x,\underline{u}(t,x))\frac{\partial^2 V}{\partial x_i \partial x_j} + \sum_{i=1}^{m} f_i(t,x,\underline{u}(t,x))\frac{\partial V}{\partial x_i} \tag{3.13}$$

where $x_i, x_j \in \mathbb{R}$ is one variable dimension from state space $x \in \mathbb{R}^n$. In order to insure that $A^{\underline{u}}$ defines a Markov process, discontinuous Markov policies $\underline{u}$ cannot be omitted.

Dynamic programming: This section continues from the previous section and covers how to solve the HJB equation in the finite horizon case. Additional information about dynamic programming in relation to the optimal control of the HJB equation (3.11) is done with the Verification Theorem in Chapter 3 in [56].

The optimal control u^* to the HJB equation (3.11) can be computed using a locally consistent Markov chain discretization method [94]. The discretization yields a Markov chain with control dependent transition probabilities while the problem of solving the HJB equation is converted into a discrete dynamic programming problem which can solved using the so-called value iterations [175]. The result of the computation is a discrete approximation of the value function V^h in (3.11) and approximation of the optimal control u^{*h}, where both of them are in the form of a n-dimensional lookup table in state-space $x \in \mathbb{R}^n$. The superscript h indicates that the value function and control are computed for the discretized problem.

There exists many discretization methods and we will focus on the so-called upwind

derivative approximation

$$\frac{\partial V}{\partial x_i} \approx \frac{b^+}{\Delta x_i}\left(V^h(x^h + x_i^h) - V^h(x^h)\right) - \frac{b^-}{\Delta x_i}\left(V^h(x^h) - V^h(x^h - x_i^h)\right) \tag{3.14}$$

$$\frac{\partial^2 V}{\partial (x_i)^2} \approx \frac{\sigma^2}{2(\Delta a_{i,j})^2}\left(V^h(x^h + x_i^h) - V^h(x^h)\right) - \frac{\sigma^2}{2(\Delta a_{i,j})^2}\left(V^h(x^h) - V^h(x^h - x_i^h)\right) \tag{3.15}$$

where the Δ denotes a positive discrete step size for corresponding variable and

$$b^+ = \max\{0, f(s, x(s), u(s))\} \tag{3.16}$$

$$b^- = \max\{0, -f(s, x(s), u(s))\} \tag{3.17}$$

We obtain the dynamic programming expression

$$V^h(x) = \min_u \left\{ \Delta t^h(x,u) + \sum_{\Delta x \in \mathscr{N}} p^{\pm}(x,u) V^h(x + \Delta x) \right\} \tag{3.18}$$

where

$$p^{\pm}(x,u) = \Delta t^h(x,u)\left(\frac{b^{\pm}}{\Delta x} + \frac{\sigma}{2(\Delta x)^2}\right) \tag{3.19}$$

are the discrete Markov-chain transition probabilities $p^{\pm}(x,u)$ from the point x to its neighboring points $\mathscr{N}$ denoted by $x \pm \Delta x$. The implicit time interpolation interval [94] is given by

$$\Delta t^h(x,u) = 1 \Bigg/ \left(\sum_{\Delta x} \frac{|b^{\pm}|}{\Delta x} + \frac{\sigma^2}{(\Delta x)^2}\right) \tag{3.20}$$

where $|\cdot|$ denotes absolute value.

Expression (3.18) is the discrete version of (3.11) and the discrete approximation V^h of the value function V can be solved numerically using value iterations [175] starting

from an initial guess for the V^h values. Our discrete solution V^h is solved on a discrete bounded state-space domain $\mathscr{X} \subset \mathbb{R}^n$.

The dynamic programming expression (3.18) is guaranteed to converge toward the continuous HJB expression (3.11) as the step size $\Delta \to 0$ using a Viscosity solution approach given in Chapter 13 in [94]. The dynamic programming expression (3.18) is guaranteed to give a steady-state optimal feedback control u^{h*} and corresponding value function V^h solution due to Markov-chain discretization method given by the Markov-chain transition probabilities (3.19) which is discussed in detail in Chapter 14, Section 3 in [175].

Chapter 4

Stochastic Optimal Control Navigation with the Avoidance of Unsafe Configurations

This chapter is a reprint of the paper

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. ”Stochastic optimal control navigation with the avoidance of unsafe configurations.” In 2016 international conference on unmanned aircraft systems (ICUAS), pp. 211-218. IEEE, 2016.

4.1 Introduction

With the increased use of mobile robots and autonomous vehicles, there is a pressing need to study control algorithms that improve the autonomous capabilities of these vehicles. In the context of this paper, the improvement means (a) navigating while taking into account uncertainty in the trajectories of other vehicles; (b) performing maneuvers to avoid unsafe configurations in which damage from another vehicle could occur; or, (c) reaching "beneficial" configurations, e.g., those states in which the vehicle has an advantage over another vehicle. While it is likely that at least one of these three characteristics appears in practical applications, all three characteristics exist in the scenario presented in this paper.

We consider a scenario with two vehicles in which the faster blue (B) has to enter the "tail" sector of the slower red (R). This relative position of B with respect to R is considered advantageous. The R's navigation and intent are unknown to B. However, R may have the same intent as B. If this is the case, it would be best if B can avoid any configuration from which R can achieve this potential goal.

A problem in which two vehicles have the intent to reach their respective target sets gives rise to a two-target pursuit-evasion differential game analyzed in [63, 64] with deterministic vehicles and the "capture" condition that the winner has to point to the opponent within a specific radius, e.g. range of the weapon. A stochastic variant of the problem has been considered in [192, 193]. These and other earlier theoretical

works have been surveyed in [69]. Another line of work has been focused more on applications [50, 181]. Observing recurrent computational problems associated with long-planning horizons, which are essential for making good (non-greedy) maneuver choices during air combat, in [115] the authors proposed an approximate dynamic programming method (ADP) on a finite horizon exploiting a variety of techniques, including "extensive feature development, trajectory sampling, reward shaping, and an improved policy extraction technique using rollout" and a neural net classifier to model adversary aircraft maneuvering policy.

The contribution of this paper is that it uses infinite horizon stochastic optimal control to compute a navigation strategy for B to (1) enter the "tail" of R while it (2) anticipates the uncertainty of R's trajectory and (3) avoids unsafe configurations from which R may be able to enter the B's tail. To identify unsafe configurations, stochastic optimal control for R to enter the B's tail is computed together with the corresponding expected time to reach the B's "tail". Any configuration in which this time is shorter than the time required for B to enter the R's "tail" is considered unsafe, and included with a high penalty in the computation of the navigation strategy for B. Our target set is closely related to the target set in [115], and although we use infinite horizon stochastic control, there are no real-time computational issues.

The work presented here is based on the methods from [6],[12],[8] and [38], and this paper reports the first real-time implementation of a computed stochastic optimal

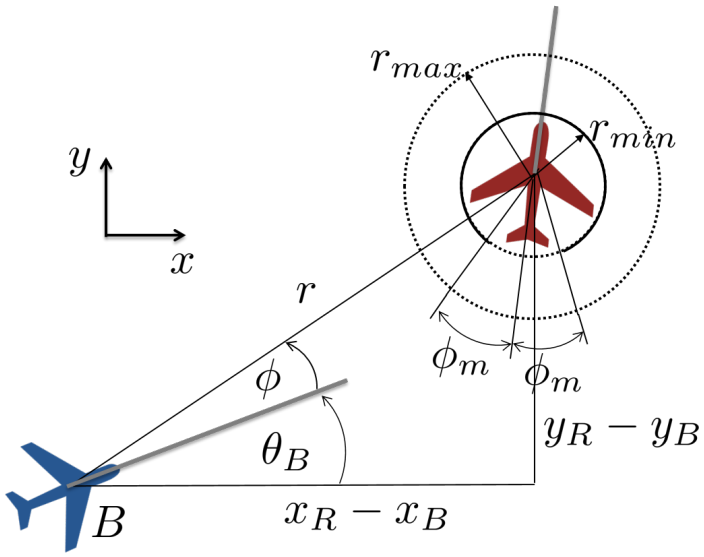


Figure 4.1: The blue (B) and red (R) vehicles

control navigation strategy.

The rest of the paper is organized as follows. In Section II, we present the problem formulation followed by Section III on the state space model. In Section IV and V, minimum time optimal controls for B and R are defined. Section VI discusses the computation of unsafe configurations. Section VII reports our results of numerical simulations and small-scale robot experiments. Section VIII gives conclusions.

4.2 Problem Formulation

The problem we consider here includes two vehicles, blue (B) and red (R), see Fig. 4.1. The goal of B is to take an advantageous position with regard to R while the

intent of R is unknown. Therefore, our problem is to design a navigation strategy for B which anticipates this uncertainty. While navigating B under the assumption that R has the goal to take an advantageous position to it, i.e., that R is its opponent, is too conservative, it makes sense that B should avoid relative configurations from which R can take an advantage.

Our problem formulation is based on the fixed-speed Dubins vehicle model for both B and R. The B vehicle has a constant velocity $v_B > v_R$ and its control variable is the turning rate u_B, i.e.,

$$dx_B = v_B \cos\theta_B dt \tag{4.1}$$

$$dy_B = v_B \sin\theta_B dt \tag{4.2}$$

$$d\theta = u_B dt \tag{4.3}$$

The R vehicle's model has the same structure with a constant velocity v_R, however, from the point of view of B, the R's navigation is unknown. Therefore, we model its turning rate as a stochastic process, i.e.,

$$dx_R = v_R \cos\theta_R dt \tag{4.4}$$

$$dy_R = v_R \sin\theta_R dt \tag{4.5}$$

$$d\theta_R = \sigma_R dw_R \tag{4.6}$$

where dw_R are unit intensity Wiener process increments. Consequently, $\theta_R(t)$ is a random walk on the periodic interval $[-\pi, \pi]$.

Instead of dealing with the six state equations in Cartesian coordinates (4.1)-(4.6), we use the relative coordinates (4.10)-(4.12) and the Itô calculus (see Appendix A to C) to derive a stochastic evolution of the relative position between B and R with only three state variables r, ϕ and α as

$$dr = (v_R \cos(\phi - \alpha) - v_B \cos\phi)\, dt = b_r dt \tag{4.7}$$

$$d\phi = \left(-u_B + \frac{1}{r}(-v_R \sin(\phi - \alpha) + v_B \sin\phi)\right) dt = b_\phi dt \tag{4.8}$$

$$d\alpha = -u_B dt + \sigma_R dw_R = b_\alpha dt + \sigma_R dw_R \tag{4.9}$$

The advantageous position for B is the one in the R's "tail" and with the heading direction θ_B aligned, to a certain precision, with the heading direction of R, θ_R. The "tail" is a conic section of finite radius behind R depicted in Fig. 4.1. The sector is bounded with angles $\pm\phi_m$ and distances r_{min} and r_{max}. While r_{max} defines the "length" of the "tail", the distance r_{min} is the minimal radius at which the two vehicles can approach each other without the risk of collision. As part of the problem formulation, the vehicles are modeled as fixed-speed Dubins vehicles with controllable turning rates, which are bounded by a certain ω_{max}. The velocity of B is greater that the velocity of R, $v_B > v_R$, and detailed vehicle models are presented in the following section.

If we denote the states of B and R with (x_B, y_B, θ_B) and (x_R, y_R, θ_R), respectively, then the relative state between B and R is uniquely defined with the distance r, bearing

angle ϕ and difference of the heading angles α defined as

$$r^2 = (x_R - x_B)^2 + (y_R - y_B)^2 \tag{4.10}$$

$$\phi = \arctan\left(\frac{y_R - y_B}{x_R - x_B}\right) - \theta_B \tag{4.11}$$

$$\alpha = \theta_R - \theta_B \tag{4.12}$$

Using (4.10)-(4.12) as state variables, we can formally define the target set for B as

$$\mathscr{T}_{BT} = \left\{ (r,\phi,\alpha) \;\middle|\; \begin{array}{lll} r_{min} & < r < & r_{max}, \\ -\phi_m & < \phi < & \phi_m, \\ -\alpha_m & < \alpha < & \alpha_m \end{array} \right\} \tag{4.13}$$

With the same notation, we can also define the set that B should avoid to prevent a collision with R as

$$\mathscr{T}_{BA} = \left\{ (r,\phi,\alpha) \;\middle|\; \begin{array}{c} r \leq r_{min}, \\ -\pi < \phi < -\phi_m \text{ or } \phi_m < \phi < \pi \end{array} \right\} \tag{4.14}$$

The sets $\mathscr{T}_{BT}$ and $\mathscr{T}_{BA}$ partition the terminal set, $\mathscr{T}_B$, as follows: $\mathscr{T}_B = \mathscr{T}_{BT} \cup \mathscr{T}_{BA}$, where $\mathscr{T}_{BT} \cap \mathscr{T}_{BA} = \emptyset$.

4.3 Minimum Time Optimal Control for B

The minimum time optimal control for B is the control that minimizes the cost function

$$\mathscr{J}(u_B) = g(r(t_B), \phi(t_B), \alpha(t_B)) + \int_0^{t_B} dt \tag{4.15}$$

subject to the state space evolution (4.7)-(4.9), where t_B denotes the time at which B reaches the terminal set $\mathscr{T}_B$ and $g(r(t_B),\phi(t_B),\alpha(t_B))$ is the terminal cost of the terminal state in which $(r(t_B),\phi(t_B),\alpha(t_B)) \in \mathscr{T}_B$. If $\mathscr{T}_B = \mathscr{T}_{BT}$ and $g(r(t_B),\phi(t_B),\alpha(t_B)) = 0$, then we deal with the pure minimum time optimal control cost function. Otherwise, $\mathscr{T}_B = \mathscr{T}_{BT} \cup \mathscr{T}_{BA}$ and

$$g(r(t_B),\phi(t_B),\alpha(t_B)) = \begin{cases} 0, \text{for } (r(t_B),\phi(t_B),\alpha(t_B)) \in \mathscr{T}_{BT} \\ M, \text{for } (r(t_B),\phi(t_B),\alpha(t_B)) \in \mathscr{T}_{BA} \end{cases} \tag{4.16}$$

with a large positive value for M to penalize reaching the set $\mathscr{T}_{BA}$, in this case, (4.15) yields the optimal control that avoids that set.

The optimal control u_B^* can be computed as the steady state of the Hamilton-Jacobi-Bellman (HJB) partial differential equation defined as

$$0 = \min_{u_B} \left\{ \mathscr{L}^B V(r,\alpha,\phi,u_B) + 1 \right\} \tag{4.17}$$

where V is the so-called value, or cost-to-go function, $V(r,\phi,\alpha) = g$ for $(r,\phi,\alpha) \in \mathscr{T}_B$ and $\mathscr{L}^B$ is the differential operator

$$\mathscr{L}^B = b_r \frac{\partial}{\partial r} + b_\phi \frac{\partial}{\partial \phi} + b_\alpha \frac{\partial}{\partial \alpha} + \frac{1}{2}\sigma_R^2 \frac{\partial^2}{\partial \alpha^2} \tag{4.18}$$

The optimal control u_B^* can be computed using a locally consistent Markov chain discretization of the HJB equation. The discretization yields a Markov chain with control dependent transition probabilities while the problem of solving the HJB equation is converted into a discrete dynamic programming problem which can be solved using

the so-called value iterations. The result of the computation is a discrete approximation of the value function V^h and optimal control u_B^{*h}, where both of them are in the form of a three-dimensional lookup table. The superscript h indicates that the value function and control are computed for the discretized problem.

To discretize HJB in the state space, we first define the discrete steps Δr, $\Delta\phi$ and $\Delta\alpha$ for r, ϕ and α, respectively, and substitute the following approximations in (4.17)

$$b_r \frac{\partial V}{\partial r} \approx \frac{b_{r^h}^+}{\Delta r}\left(V(r^h+\Delta r,\phi^h,\alpha^h)-V(r^h,\phi^h,\alpha^h)\right) \tag{4.19}$$

$$-\frac{b_{r^h}^-}{\Delta r}\left(V(r^h,\phi^h,\alpha^h)-V(r^h-\Delta r,\phi^h,\alpha^h)\right) \tag{4.20}$$

$$b_\phi \frac{\partial V}{\partial \phi} \approx \frac{b_{\phi^h}^+}{\Delta \phi}\left(V(r^h,\phi^h+\Delta\phi,\alpha^h)-V(r^h,\phi^h,\alpha^h)\right) \tag{4.21}$$

$$-\frac{b_{\phi^h}^-}{\Delta \phi}\left(V(r^h,\phi^h,\alpha^h)-V(r^h,\phi^h-\Delta\phi,\alpha^h)\right) \tag{4.22}$$

$$b_\alpha \frac{\partial V}{\partial \alpha} \approx \frac{b_{\alpha^h}^+}{\Delta \alpha}\left(V(r^h,\phi^h,\alpha^h+\Delta\alpha)-V(r^h,\phi^h,\alpha^h)\right) \tag{4.23}$$

$$-\frac{b_{\alpha^h}^-}{\Delta \alpha}\left(V(r^h,\phi^h,\alpha^h)-V(r^h,\phi^h,\alpha^h-\Delta\alpha)\right) \tag{4.24}$$

$$\frac{\partial^2 V}{\partial(\alpha)^2} \approx \frac{\sigma_R^2}{(\Delta\alpha)^2}\left(V(r^h,\phi^h,\alpha^h+\Delta\alpha)-V(r^h,\phi^h,\alpha^h)\right) \tag{4.25}$$

$$-\frac{\sigma_R^2}{(\Delta\alpha)^2}\left(V(r^h,\phi^h,\alpha^h)-V(r^h,\phi^h,\alpha^h-\Delta\alpha)\right) \tag{4.26}$$

which are the so-called upwind derivative approximations, where $b_{r^h}^+ = \max[0, b_{r^h}]$, $b_{r^h}^- = \max[0, -b_{r^h}]$ and $b_{\phi^h}^+$, $b_{\phi^h}^-$, $b_{\alpha^h}^+$, $b_{\alpha^h}^-$ are defined in the same way. The superscript h indicates that the term is evaluated at the points of the discretized state space with $r^{h+1} - r^h = \Delta r$, $\alpha^{h+1} - \alpha^h = \Delta\alpha$ and $\phi^{h+1} - \phi^h = \Delta\phi$. After the substitution, we move all the terms that include $V(r^h, \phi^h, \alpha^h)$ to the left side of the expression (4.17)

approximation and define

$$\Delta t = \left(\frac{|b_{r^h}|}{\Delta r} + \frac{|b_{\phi^h}|}{\Delta \phi} + \frac{|b_{\alpha^h}|}{\Delta \alpha} + \frac{\sigma_R^2}{(\Delta \alpha)^2} \right)^{-1} \tag{4.27}$$

with $|b_{r^h}| = b^+_{r^h} + b^-_{r^h}$, $|b_{\phi^h}| = b^+_{\phi^h} + b^-_{\phi^h}$, and $|b_{\alpha^h}| = b^+_{\alpha^h} + b^-_{\alpha^h}$ to obtain $V^h = V(r^h, \phi^h, \alpha^h)$ as

$$\begin{aligned} V^h = \min_{u_B} \{ & \Delta t + p_{\Delta r^+} V(r^h + \Delta r, \phi^h, \alpha^h) \\ & + p_{\Delta r^-} V(r^h - \Delta r, \phi^h, \alpha^h) + p_{\Delta \phi^+} V(r^h, \phi^h + \Delta \phi, \alpha^h) \\ & + p_{\Delta \phi^-} V(r^h, \phi^h - \Delta \phi, \alpha^h) + p_{\Delta \alpha^+} V(r^h, \phi^h, \alpha^h + \Delta \alpha) \\ & + p_{\Delta \alpha^-} V(r^h, \phi^h, \alpha^h - \Delta \alpha) \} \end{aligned} \tag{4.28}$$

with $p_{\Delta r^\pm} = \Delta t b^\pm_{r^h} / \Delta r$, $p_{\Delta \phi^\pm} = \Delta t b^\pm_{\phi^h} / \Delta \phi$ and $p_{\Delta \alpha^\pm} = \Delta t \left(b^\pm_{\alpha^h} / \Delta \alpha + \sigma_R^2 / (2 \Delta \alpha^2) \right)$ that can be interpreted as discrete Markov-chain transition probabilities from the points denoted in the brackets of the value function on the right side of the point (r^h, ϕ^h, α^h). Note that Δt is the time interpolation interval defined by the other problem parameters; therefore, this type of discretization is called time implicit discretization [94].

Expression (4.28) is the discrete version of (4.17) and can be solved numerically using the value iteration [175] starting from an initial guess for the $V(r, \phi, \alpha)$ values. We define the computational domain as

$$\mathscr{K} = \{ [R_{min}, R_{max}] \times [-\pi, \pi - \Delta \phi) \times [-\pi, \pi - \Delta \alpha) \} \tag{4.29}$$

which is the set bounded by the minimal $R_{min} = r_{min}$ and maximal R_{max} distances. Since, in our problem formulation, the angles ϕ and α have full ranges, in our com-

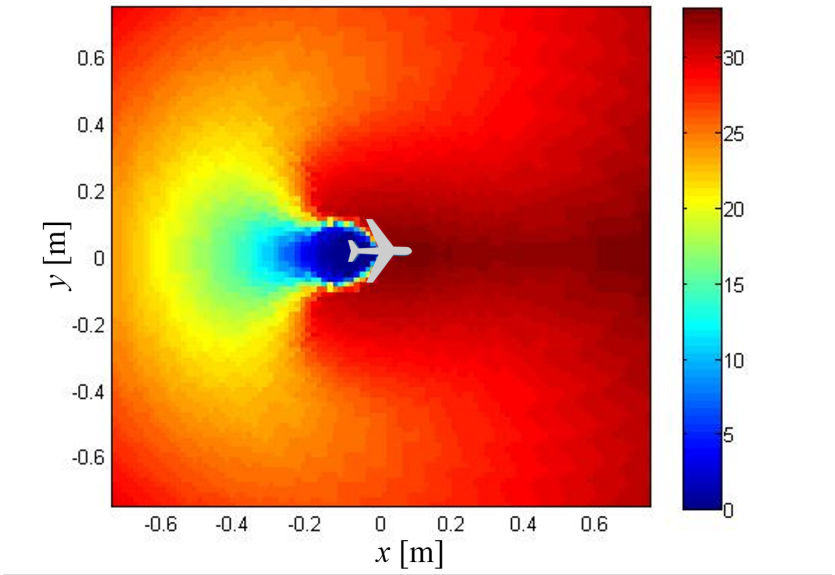


Figure 4.2: A plot of the expected time $T_B^{min}(x,y)$ for B to reach R using the optimal control u_B^*. The R vehicle is in the center pointing to the right. The B's and R's velocities are $v_B = 0.1$ and $v_R = 0.05$, respectively.

putational domain, the pairs of points $(r^h, -\pi, \alpha^h)$ and $(r^h, \pi - \Delta\phi, \alpha^h)$, as well as $(r^h, \phi^h, -\pi)$ and $(r^h, \phi^h, \pi - \Delta\alpha)$ are next to each other. For the boundary of the computational domain R_{max}, we use the reflective boundary conditions $V(R_{max}, \phi^h, \alpha^h) = V(R_{max} - \Delta r, \phi^h, \alpha^h)$. This way, we avoid specifying the value function at these boundaries, but actually incorporate in our solution that the stochastic process does not cross the boundaries. This is an approximation for the boundary $r = R_{max}$; therefore, we should use a large enough R_{max}.

Once the optimal control is computed, it can be also used to compute the expected time to reach the target set as

$$0 = \mathscr{L}^B \overline{T}_B(r, \alpha, \phi) + 1 \tag{4.30}$$

which is the so-called backward Kolmogorov partial differential equation. Since the solution should represent the expected time to reach the set $\mathscr{T}_{BT}$, the boundary condition for (4.30) is $\overline{T}_B(r, \alpha, \phi) = 0$, for $(r(t_B), \alpha(t_B), \phi(t_B)) \in \mathscr{T}_{BT}$. The solution can be computed using the value iterations (4.28), where V^h is substituted with $\overline{T}_B^h = \overline{T}_B(r^h, \alpha^h, \phi^h)$ and *min* is excluded, i.e., there is no update of control u_B^*, which is the reason why the variable u_B is omitted in (4.30). For the boundary of the computational domain R_{max}, we use the reflective boundary condition equivalent to the one used in the computations for V^h, i.e., $\overline{T}_B(R_{max}, \phi^h, \alpha^h) = \overline{T}_B(R_{max} - \Delta r, \phi^h, \alpha^h)$.

To illustrate the optimal control solution, we created a 2D map depicted in Fig. 4.2. The map coordinates are the x and y coordinates of B with respect to R, which is in the

center of the map and pointed to the right. The map depicts a discrete approximation of the expected time $T_B^{min}(x,y)$ defined as

$$T_B^{min}(x,y) = \min_{\alpha} \left\{ \overline{T}_B(r(x,y), \alpha, \phi(x,y,\alpha) \right\} \tag{4.31}$$

in which we use $\overline{T}_B^h \approx \overline{T}_B(r(x,y), \alpha, \phi(x,y,\alpha)$. In other words, for every point (x, y) and any heading angle of B at that point, i.e., the closest discrete point, we depict the minimum expected time to reach the target set. As expected, the figure shows that B, which is faster, reaches its target set in a shorter time if it approaches R from the "tail". If R is approached from the front, it takes B much longer because it has to avoid the "nose" of R.

4.4 Minimum Time Optimal Control for R

With the goal of formulating a set of relative positions that should be avoided by B, we here formulate the optimal control for R to reach an advantageous position with respect to B. The position is defined as the one in which the R is in the "tail" sector of B and with the heading angle aligned with that of B's.

While the "tail" sector can be depicted by a figure similar to Fig. 4.1, but with the reversed positions of B and R, for its formal definition, we need to introduce a triple of coordinate variables (r, ϕ_R, α_R) describing the position of B with respect to R. The distance r is given by (4.10), while the bearing angle ϕ_R and the difference of the

heading angles α_R with respect to R are

$$\phi_R = \arctan\left(\frac{y_B - y_R}{x_B - x_R}\right) - \theta_R \tag{4.32}$$

$$\alpha_R = \theta_B - \theta_R \tag{4.33}$$

Using these variables, the advantageous position is reached by R when the triple of the variables is in the set $\mathscr{T}_{RT}$ defined as

$$\mathscr{T}_{RT} = \left\{ (r, \phi_R, \alpha_R) \;\middle|\; \begin{array}{lll} r_{min} & < r < & r_{max}, \\ -\phi_m & < \phi_R < & \phi_m, \\ -\alpha_m & < \alpha_R < & \alpha_m \end{array} \right\} \tag{4.34}$$

which is the target set for R. The set that should be avoided by R due to the risk of collision with B is

$$\mathscr{T}_{RA} = \left\{ (r, \phi_R, \alpha_R) \;\middle|\; \begin{array}{c} r \leq r_{min}, \\ -\pi < \phi_R < -\phi_m \text{ or} \\ \phi_m < \phi_R < \pi \end{array} \right\} \tag{4.35}$$

and the terminal set $\mathscr{T}_R$ for R is the union of these two sets, i.e., $\mathscr{T}_R = \mathscr{T}_{RT} \cup \mathscr{T}_{RA}$.

Here we formulate the optimal control for R to reach the target set $\mathscr{T}_{RT}$ and avoid $\mathscr{T}_{RA}$, but the navigaion of B is unknown to R. Therefore, for the control of R, we use (4.1)-(4.3) and (4.4)-(4.6), in which expressions (4.3) and (4.6) are substituted with

$$d\theta_B^R = \sigma_B dw_B \text{ and } d\theta_R^R = u_R dt \tag{4.36}$$

respectively. Together with (4.10) and (4.32)-(4.33), this yields a stochastic evolution

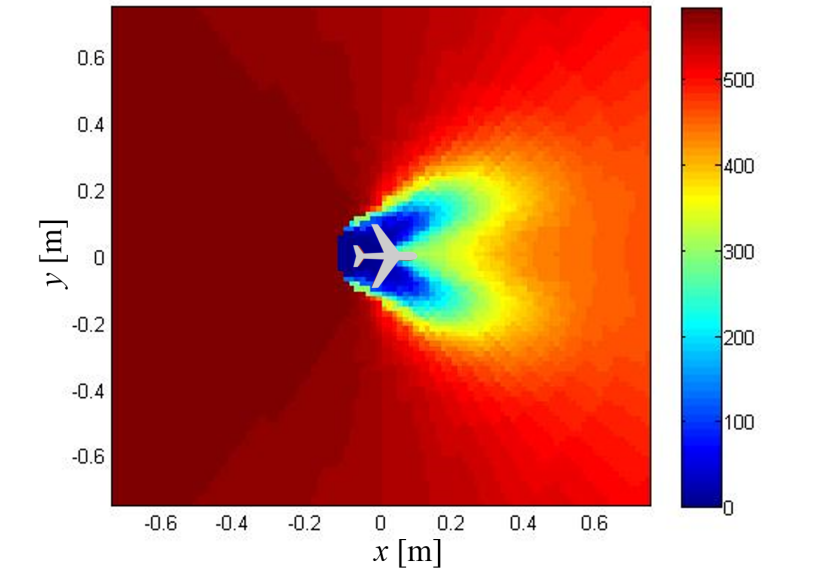


Figure 4.3: A plot of the expected time $T_R^{min}(x,y)$ for R to reach B using the optimal control u_R^*. The B vehicle is in the center pointing to the right. The B's and R's velocities are $v_B = 0.1$ and $v_R = 0.05$, respectively.

of the relative position of B with respect to R as

$$dr = b_r dt, \quad d\phi_R = b_\phi^R dt, \quad d\alpha_R = b_\alpha^R dt + \sigma_B dw_B \tag{4.37}$$

with b_r from (4.7) and

$$b_\phi^R = \left(-u_R + \tfrac{1}{r}\left(-v_R \sin(\phi^R - \alpha^R) + v_B \sin\phi^R\right)\right) \tag{4.38}$$

$$b_\alpha^R = -u_R \tag{4.39}$$

The expected time optimal control for R is based on the cost function

$$\mathscr{J}(u_R) = g(r(t_R), \phi_R(t_R), \alpha_R(t_R)) + \left\{ \int_0^{t_R} dt \right\} \tag{4.40}$$

where t_R is the time at which R reaches the terminal set $\mathscr{T}_R$. The terminal costs are $g = 0$ for $(r(t_R), \phi_R(t_R)\alpha_r(t_R)) \in \mathscr{T}_{RT}$ and $g = M$ for $(r(t_R), \phi_R(t_R)\alpha_r(t_R)) \in \mathscr{T}_{RA}$. The optimal control u_R^* can be computed from the corresponding HJB equation

$$0 = \min_{u_R} \left\{ \mathscr{L}^R V_R(r, \alpha_R, \phi_R, u_R) + 1 \right\} \tag{4.41}$$

where the differential operator $\mathscr{L}^R$ is

$$\mathscr{L}^R = b_r \frac{\partial}{\partial r} + b_\phi^R \frac{\partial}{\partial \phi^R} + b_\alpha^R \frac{\partial}{\partial \alpha^R} + \frac{1}{2}\sigma_B^2 \frac{\partial^2}{\partial (\alpha^R)^2} \tag{4.42}$$

and, due to the terminal costs, (4.41) satisfies the boundary conditions

$$V_R(r, \phi_R, \alpha_R) = \begin{cases} 0, \text{for } (r, \phi_R, \alpha_R) \in \mathscr{T}_{RT} \\ M, \text{for } (r, \phi_R, \alpha_R) \in \mathscr{T}_{RA} \end{cases} \tag{4.43}$$

reflecting that R has to avoid the noise of B. To evaluate the expected time to reach the target set $\mathscr{T}_{RT}$, we solve the backward Kolmogorov equation

$$0 = \mathscr{L}^R \overline{T}_R(r, \alpha_R, \phi_R) + 1 \tag{4.44}$$

with the boundary condition $\overline{T}_R(r, \alpha_R, \phi_R) = 0$ for $(r, \alpha_R, \phi_R) \in \mathscr{T}_{RT}$. This expected time together with the expected time $\overline{T}_B$ from the previous section is important for the definition of the safety region that should be avoided by B.

All computations of the value function V_R, expected time $\overline{T}_R$ and their discrete approximations V_R^h and $\overline{T}_R^h$ can be performed in the same way as in the case of the B vehicle. Similarly as in the previous sections, to illustrate the optimal control solution

for R, we create a map, see Fig. 4.3, which depicts the expected time $T_R^{min}(x,y)$ defined as

$$T_R^{min}(x,y) = \min_{\alpha} \left\{ \overline{T}_R(r(x,y), \alpha_R, \phi_R(x,y,\alpha_R) \right\} \tag{4.45}$$

with the x and y coordinates of R with respect to B, which is now in the center of the map and pointed to the right. For every point (x,y) and all heading angles of R at (x,y) we depict the minimum expected time to reach the target set. The shape of the map is dictated by the fact that R is slower than B. Because of that, the time to enter the tail of B from the tail direction is very long and the only chance for R to enter its target set $\mathscr{T}_{RT}$ is to get in a configuration in which B is in its proximity, but has to turn around to enter the tail of R.

An illustration of the configuration in which R can enter its target set is provided in Fig. 4.4. In Fig. 4.4*a* and 4.4*b*, B, which is faster than R, passes close to R and, in order to enter the tail of R, it has to make a full turn. While B maneuvers, there is a position from which R can sharply turn and enter the tail of B. While B is going farther from R, R is still in its tail and can adjust its heading angle to be aligned with the heading angle of B. Once the heading angles are aligned, R has reached its target set.

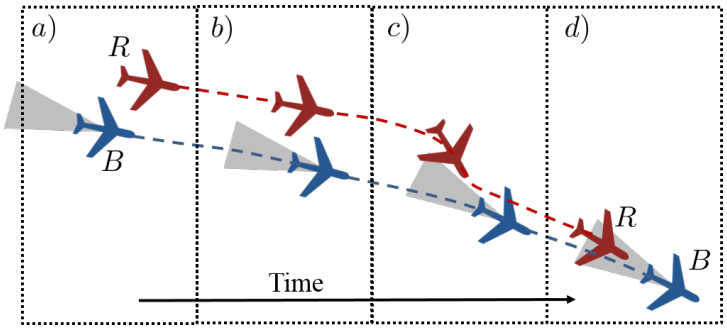


Figure 4.4: *a*) Configuration from which R can enter the tail of B; *b*) B is slightly in front of R and starts maneuvers to enter its tail; *c*) R performs a sharp maneuver to enter the tail of B (gray sector); *d*) R aligns its heading angle with B and reaches its target set.

4.5 Minimum Time Optimal Control that Avoids Unsafe Configurations

We should realize that the control for B defined in Section IV does not take into account that R may have the goal to enter the tail of B. This possibility should be addressed in the navigation strategy for B so that it avoids any configuration from which it can happen *before* B enters its target set. Because of the *before* condition, it is most natural to identify unsafe configurations not in terms of probability, but in terms of expected time.

To identify unsafe configurations for B, we find all relative positions (r, α, ϕ) in which the expected time of R to reach its target set is shorter than the expected time of B to reach its set. This can be expressed with the following inequality

$$\overline{T}_R(r, -\alpha, \pi + \phi - \alpha) < \overline{T}_B(r, \alpha, \phi) \tag{4.46}$$

Initialization :
$\mathscr{T}^{\mathscr{S}}{}_B = \mathscr{T}_{BT} \cup \mathscr{T}_{BA}$
$V(r^h, \alpha^h, \phi^h) = 0$ for all $(r^h, \alpha^h, \phi^h) \in \mathscr{T}_{BT}$
$V(r^h, \alpha^h, \phi^h) = M$ for all $(r^h, \alpha^h, \phi^h) \in \mathscr{T}_{BA}$
loop: for all (r^h, α^h, ϕ^h)
$\alpha_R^h := -\alpha^h$
$\phi_R^h := \pi + \phi^h - \alpha^h$
if $\overline{T}_R(r^h, \alpha_R^h, \phi_R^h) < \overline{T}_B(r^h, \alpha^h, \phi^h)$ *then*
$\mathscr{T}^{\mathscr{S}}{}_B = \mathscr{T}^{\mathscr{S}}{}_B \cup (r^h, \alpha^h, \phi^h)$
$V(r^h, \alpha^h, \phi^h) = M$
end if
end loop

Figure 4.5: Pseudo code for computing the terminal set and associated boundary condition. The symbol ":=" denotes the assignment of the closest discrete value on the right-hand side of the expression.

in which we exploit the relations $\alpha_R = -\alpha$ and $\phi_R = \pi + \phi - \alpha$, see Fig. 4.1. Once we find all of these positions, we include them in the terminal set for B and associate with them a high terminal cost. The pseudo code for computing the resulting terminal set $\mathscr{T}^{\mathscr{S}}{}_B$ and corresponding boundary condition for the discrete approximation of V in (4.17) is provided in Fig. 4.5.

Instead of $\mathscr{T}_B$ in (4.28), we use the computed terminal set $\mathscr{T}^{\mathscr{S}}{}_B$ and associated values for V to compute the optimal control $u_B^{\mathscr{S}}$ that avoids unsafe configurations. Our results are illustrated by the examples in the following section.

4.6 Results

The numerical simulation results of this section are based on the Dubins vehicle models (4.1)-(4.6) with the velocities $v_B = 0.1$ and $v_R = 0.05$ for B and R, respectively. These values are selected due to the verification of the results with robot experiments. We set the velocity of B to be twice as high as the velocity of R. The simulation parameters are in the ranges that are suitable for testing control algorithms using small-scale laboratory robot (e-puck) experiments and the speed of algorithms for real-time implementations.

The navigation uncertainties of R in (4.9) and of B in (4.37) are modeled with $\sigma_R = 10$ and $\sigma_B = 10$, respectively. The target set $\mathscr{T}_{BT}$ in (4.13) and the avoidance set $\mathscr{T}_{BA}$ in (4.14) are based on the parameters $r_{min} = 0.05$, $r_{min} = 0.15$, $\phi_m = 10$deg and $\alpha_m = 20$deg. The same parameters are used for the target set $\mathscr{T}_{RT}$ and the avoidance set $\mathscr{T}_{RA}$ in (4.34) and (4.35), respectively. The value which is used to penalize for state-space configurations that should be avoided is $M = 10^4$.

We first tested the pure time optimal control for B to enter the tail of R while R was moving along a straight or circular trajectory. Then we simulated B and R, while each of the vehicles was following its optimal control u_B^* and u_R^*, respectively. Once we verified both controllers, we searched for an initial condition from which R entered the tail of B. This result is presented in Fig. 4.6*a*. As it can be seen, while B tries to make a turn in a *greedy* manner to enter as soon as possible the "tail" of R, it opens space for

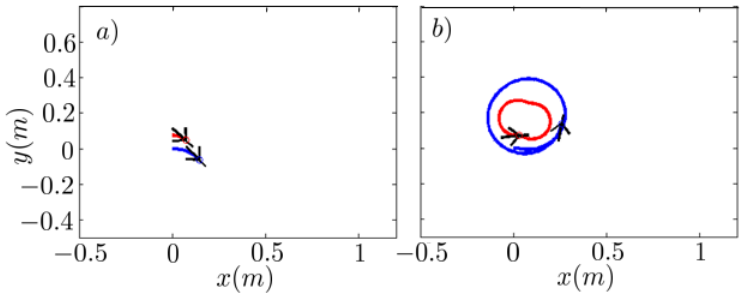


Figure 4.6: *a*) Simulation with the initial condition providing that R enters the tail of B while both R and B use their optimal controllers, u_R^* and u_B^*, respectively. *b*) Simulation for the same initial condition as in (*a*) with R using the optimal controller u_R^*, but with B using the controller $u_B^{\mathscr{S}}$ that avoids unsafe positions.

R to enter the tail of B. This scenario has been previously depicted in Fig. 4.4, but on a scale which better illustrates the maneuvers.

Fig. 4.6*b* shows the simulation with the same initial condition as in Fig. 4.6*a*, but now the B's controller $u_B^{\mathscr{S}}$, which avoids unsafe configurations, is applied. The outcome is that B turns in a less greedy manner and, before it starts turning, it first goes to a safe distance from R. Because it avoids unsafe configurations, B turns counter-clockwise (CCW) in 4.6*b*. Therefore, R cannot enter the "tail" of B, however, as the simulation shows, B also cannot enter the "tail" of R. Although this seems as a possible outcome, note that we set the initial condition in which B and R are close to each other. In reality, B will approach R from a distance and maneuver to avoid the position such as our initial condition in Fig. 4.6*a* and *b*. Most of our simulations show that B enters the tail of R and we present one simulation example in Fig. 4.7, in which B makes multiple turns before it reaches the "tail" of R.

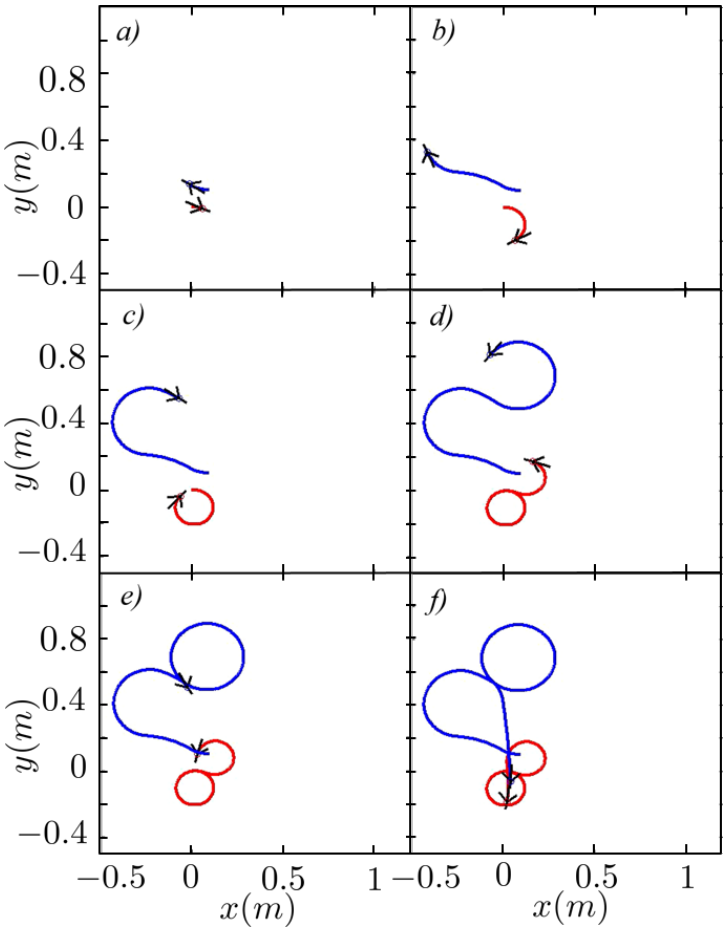


Figure 4.7: Simulation in which R uses the optimal control u_R^* and B uses the optimal control $u_B^{\mathscr{S}}$ that avoids unsafe configurations. The simulation time progresses from a to f. Panel f shows the configuration in which B enters the "tail" of R.

The simulation in Fig. 4.7 starts in the position in which B and R are facing each other. From Fig. 4.7a to c, it can be seen that both B and R turn clockwise (CW). From Fig. 4.7c to d, both of the vehicles switch the turning directions and, in Fig. 4.7e, it is clear that B is in the position to enter the tail of R.

4.6.1 Robot Experiments

After testing the stochastic optimal controllers in the numerical simulations, we implemented the numerically computed controllers in experiments with real e-puck robots. The experimental setup included two e-puck robots with infrared markers that were tracked by a motion capturing system composed of 4 Bonita 10 Vicon cameras. The reasons for the experiments are to show that the computed optimal controllers can be implemented on a real hardware platform and that the speed of control computations is sufficient for the real-time control.

Our software architecture for the real-time control is depicted in Fig. 4.8. Communication with the e-puck robots is over bluetooth. The Matlab script computes command values for the left *uL* and the right *uR* wheels of each robot and sends the values to the TCP server, which facilitates communication with the robots. Although all the values are computed by the single script, the computations are implemented so that the controls for *B* and *R* are computed independently. The Matlab script reads the mea-

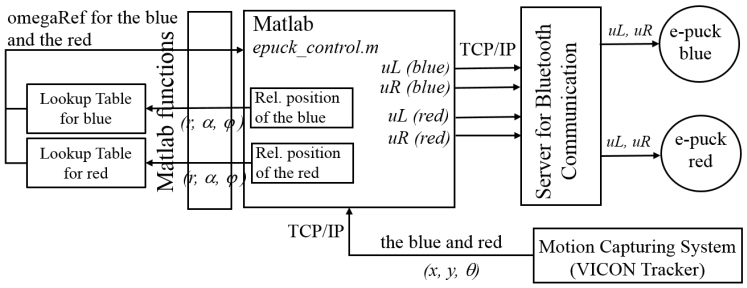


Figure 4.8: Software architecture for the robot experiments

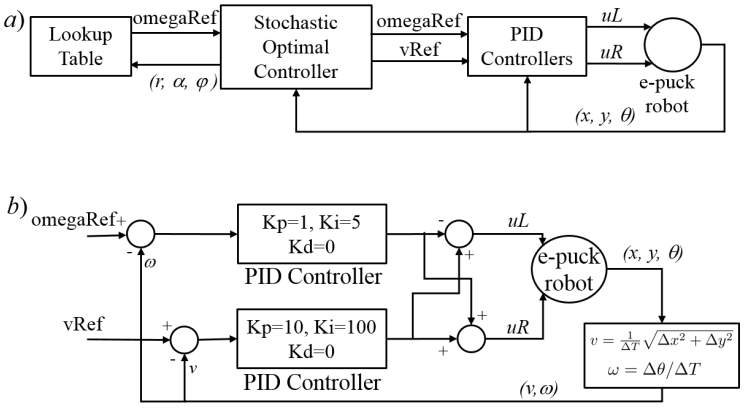


Figure 4.9: a) The robot's controller; b) Internal structure of the PID controllers implemented for each robot

surements from the motion capturing system, computes the relative position of B with respect to R and vice versa, and uses the computed values to read necessary control actions from the lookup tables that store optimal controls. The optimal controls are optimal turning rates and the Matlab script runs the PID control loops for the turning rates and fixed velocities of the B and R e-pucks.

The details of the Matlab-implemented control loops are illustrated in Fig. 4.9. For both B and R, we have the control loop presented in Fig. 4.9a. The figure shows the lookup table that stores optimal control actions and PID controllers for keeping the turning rate and velocity at the required levels. While the velocity is at a constant value of $v_B = 10cm/s$ for B and at a constant value of $v_B = 5cm/s$ for R, the block "Stochastic Optimal Controller" computes the relative position between the robots, locates the corresponding turning rate value in the lookup table and reads it so that it is set as a reference for the PID controller.

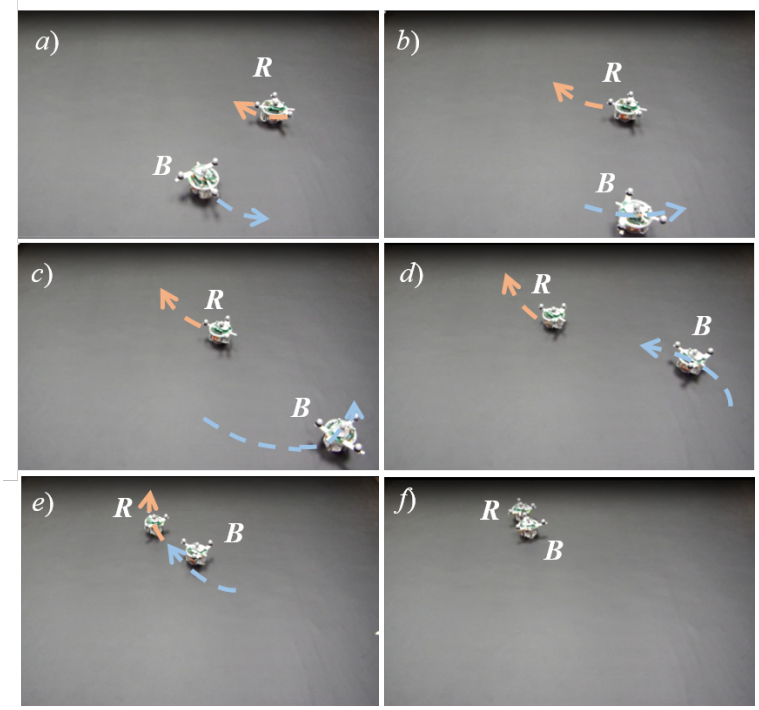


Figure 4.10: Results of the experiments with the e-puck robots. The time progresses from (a) to (f) until B reaches its target set in the "tail" of R.

Fig. 4.9*b* shows the internal structure of the couple of PID controllers implemented for each robot. The PID controller uses the motion capturing system coordinates and estimates the velocity and turning rate of the robot. These estimated values are used as measurements in the PID control loops.

With the presented software and control architecture, we were able to run stochastic optimal control with a sampling time of about 30ms on a PC computer with the Windows 7 operative system without any special real-time software, while the computer controlled both robots in separate control loops.

4.7 Conclusions

In this paper, we presented a control design for a vehicle (B) that has the goal to enter the "tail" of another vehicle (R). The navigation and intent of the R vehicle are unknown, therefore, the uncertainty of its navigation has to be anticipated in the navigation of B. Moreover, we identified unsafe configurations for the B vehicle and computed the controller that avoids them. The control design is based on infinite-time horizon stochastic optimal control and it was tested in numerical simulations. While it is commonly assumed that long-time planning horizons prevent real-time implementations of controllers, in our approach, it is not the case. To confirm it, we performed real-time robot experiments. These experiments are the first implementation of the real-time nonlinear stochastic optimal control design following from [6] and [12] to the navigation of nonholonomic vehicles. Our future work will be focused on the implementation of the control design on a high-fidelity flight simulator with a three-dimensional target set.

Chapter 5

Safe Navigation with Collision Avoidance of a Brownian Motion Obstacle

This chapter is a reprint of the paper

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. ”Safe navigation with the collision avoidance of a brownian motion obstacle.” In Dynamic Systems and Control Conference, vol. 58295, p. V003T39A009. American Society of Mechanical Engineers, 2017.

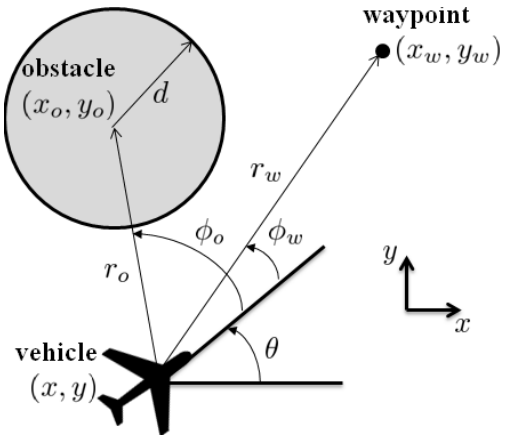


Figure 5.1: The relative position among the vehicle, obstacle and the waypoint. r_o is the distance between the vehicle and center of the obstacle. r_w is the distance between the vehicle and waypoint. θ is the heading angle of the vehicle. ϕ_w and ϕ_o are the bearing angles to the waypoint and center of the obstacle, Respectively. d is the obstacle radius.

5.1 Introduction

Collision avoidance is an integral part of vehicle navigation algorithms [100]. However, most of collision avoidance algorithms [40] are based on heuristic methods which are then integrated with navigation control algorithms without a discussion of a formal approach to the integration and optimality of the resulting controllers. Since the obstacle trajectory may not be known in advance, here we propose a stochastic optimal control approach to navigation and obstacle avoidance. The approach integrates a deterministic minimum time optimal control solution for navigation towards a waypoint and a stochastic optimal control solution to avoid an obstacle and account for its motion uncertainty. The integration is based on the value functions associated with each solution.

The problem of avoiding collision with obstacles, other mobile robots or unmanned aerial vehicles (UAV) while navigating to a location gives rise to a line of work called reach and avoid [13, 1, 122]. The problem is usually solved by two different controllers, the main controller for the path planning navigation and the secondary controller tasked to avoiding collisions [16]. There has been several variants of the secondary controller exploiting an optimal control value function [97], or using results from game theory [190]. Recently there has been work on incorporating collision avoidance as part of the main controller, where the robots [30] or UAVs [125, 73] navigate while avoid situations that lead to collisions. However, in [30] the mobile robots can change their velocities, and in [125, 73] a fixed velocity Dubins vehicle avoids collision with other Dubin vehicles.

In this paper we consider the navigation of a fixed velocity Dubins vehicle [47, 81, 133] to a prescribed waypoint. The vehicle is not subject to any disturbance such as a stochastic or deterministic wind [8], but there is the threat of a possible collision with an unpredictably moving circular obstacle as shown in Fig. 5.1. Therefore, the motion of the obstacle's center is modeled as a Brownian motion.

The contribution of this paper is that we integrate two optimal controllers into one controller that navigates towards a waypoint and avoids an obstacle. Solving the optimal control for the full 4-dimensional state space model associated with this problem is computationally challenging. Therefore, we sacrifice optimality to obtain a sub-optimal

solution which combines the two optimal solutions. The guarantee of the successful integration of the sub-problem solutions is achieved similarly to the auxiliary Markov decision problem (MDP) approach in [73], where the two sub-problems are combined together by their value functions. Although our approach does not use an approximate dynamic programming (ADP) for the navigation of UAVs [115, 138], it is based on the value function approximation, and similarly to the ADP, allows for real-time computations of control actions.

The rest of this paper is organized as follows. First we formulate the original optimal control problem, which includes the waypoint and obstacle. To cope with computationally demanding optimal control solution, we formulate two optimal control problems of (1) reaching the waypoint in a deterministic setting and (2) avoiding collision with the stochastic moving obstacle. This is followed by the section in which we combine these two optimal control solutions into a single sub-optimal controller. The work of the resulting controller is illustrated by numerical simulations. The last section of the paper presents conclusions.

5.2 Problem Formulation

The problem we consider here includes one vehicle, a waypoint and a circular obstacle. The goal of the vehicle is to reach the waypoint in a minimum time while avoiding collision with the randomly moving obstacle, see Fig. 5.1.

The fixed velocity Dubins vehicle [47, 81, 133] with a controllable turning rate $u \in [-u_{max}, u_{max}]$, $u_{max} > 0$, is given by

$$dx = v\cos\theta dt \tag{5.1}$$

$$dy = v\sin\theta dt \tag{5.2}$$

$$d\theta = udt \tag{5.3}$$

where x and y are the coordinates of the Dubins vehicle position and θ is the heading angle. The waypoint is fixed in space with coordinates (x_w, y_w), but the obstacle's motion is unknown. Therefore, we model it as a stochastic process

$$dx_o = \sigma dw_x \tag{5.4}$$

$$dy_o = \sigma dw_y \tag{5.5}$$

where dw_x and dw_y are mutually independent unit intensity Wiener process increments. Consequently, the obstacle position $(x_o(t), y_o(t))$ is a 2D Brownian motion.

We use the relative coordinates (see Fig. 5.1 and (5.22)-(5.23)) and Itô calculus (see Appendix) to derive a stochastic evolution of the relative position between the vehicle and the obstacle as

$$dr_o = \left(-v\cos\phi_o + \frac{\sigma^2}{2r_o}\right)dt + \sigma dw_{r_o} \qquad = b_{r_o}dt + n_{r_o}dw_{r_o} \tag{5.6}$$

$$d\phi_o = \left(\frac{v}{r_o}\sin\phi_o - u\right)dt + \left(\frac{\sigma}{r_o}\right)dw_{\phi_o} \qquad = b_{\phi_o}dt + n_{\phi_o}dw_{\phi_o} \tag{5.7}$$

where dw_{r_o} and dw_{ϕ_o} are mutually independent unit intensity Wiener processes.

While the vehicle navigates towards the waypoint (x_w, y_w), it has to avoid a set of relative positions

$$\mathscr{A} = \{(r_o, \phi_o) | r_o \leq d\} \tag{5.8}$$

that overlap with the obstacle, see Fig. 5.1. Now we can define the control u as the solution of the optimal control problem

$$J(\tilde{x}, u) = \min_u \left\{ E\{g(\tilde{x}(\tau))\} + \int_0^\tau dt \right\} \tag{5.9}$$

where $\tilde{x} = (r_w, \phi_w, r_o, \phi_o)$ is the four dimensional state that fully describes the relative positions among the vehicle, the waypoint and the obstacle. In expression (5.9), τ is the terminal time which corresponds to the time instant of the vehicle reaching either the waypoint or the obstacle. g is the terminal cost for hitting the obstacle defined as

$$g(\tilde{x}(\tau)) = M \text{ if } (r_o, \phi_o) \in \mathscr{A} \tag{5.10}$$

where M is a large positive cost to penalize reaching the set $\mathscr{A}$.

Notice that the optimal control solution is 4-dimensional, which is computationally challenging [94]. In order to reduce the computational cost, we propose to reduce the 4-dimensional single problem into two sub-problems. The first one is a 2-dimensional problem of reaching the waypoint without the obstacle and the second is a 2-dimensional problem of avoiding the obstacle. These two sub-problems can be solved independently of each other and then be combined to approximate the solution of the original 4-dimensional problem of navigating to the waypoint while avoiding the obsta-

cle. The first sub-problem of navigating to the waypoint (x_w, y_w) without any obstacle is given by the control u_w which solves the minimum time optimal control problem

$$J(\tilde{x}_w) = \min_{u_w} \left\{ \int_0^{\tau_w} dt \right\} \tag{5.11}$$

where $\tilde{x}_w = (r_w, \phi_w)$ is the relative position between the vehicle and the waypoint, and τ_w is the terminal time which corresponds to the time instant when the vehicle reaches the waypoint. The second sub-problem solution of avoiding the obstacle can be defined as the control u_o which solves the optimal control problem

$$J(\tilde{x}_o) = \min_{u_o} E\{g(\tilde{x}_o(\tau_o))\} \tag{5.12}$$

where $\tilde{x}_o(\tau_o) = (r_o(\tau_o), \phi_o(\tau_o))$ is the relative position between the vehicle at the terminal time point τ_o when the vehicle hits the obstacle. In the original problem (5.9), the terminal time $\tau = \min(\tau_w, \tau_o)$.

5.3 Deterministic Optimal Control to Reach a Waypoint

The first sub-problem is a deterministic optimal control problem where we minimize the time of the vehicle to reach the waypoint in the absence of the obstacle. This problem has been solved in [8, 21]. Here we introduce necessary variables, present the final results and their discretization in the discrete state space.

The relative position between the vehicle and target point is uniquely defined with

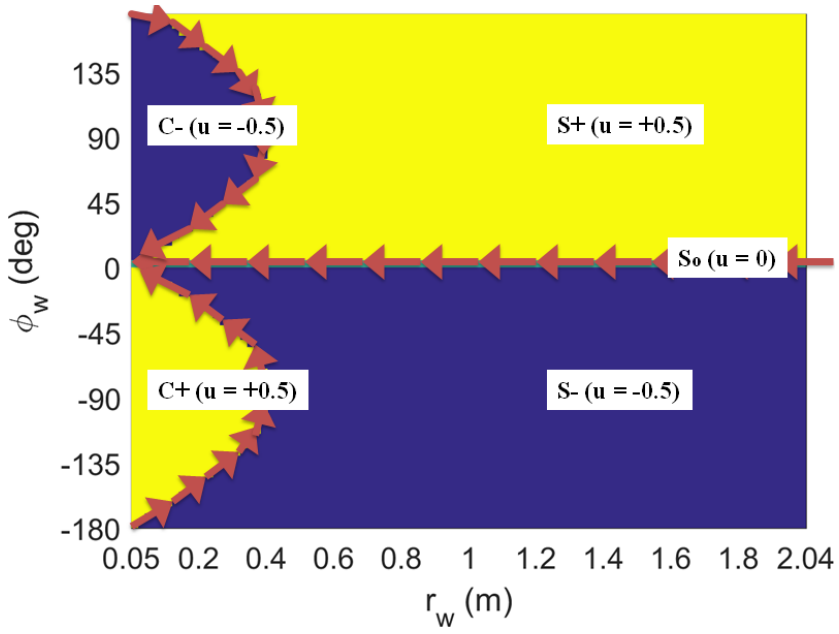


Figure 5.2: Time-Optimal partition of the state space and feedback control (5.15). The control values are in parenthesis to the left of each region. The arrows indicate the lines of characteristics.

the distance r_w and the bearing angle ϕ_w. From Fig. 5.1, it follows that

$$r_w^2 = (x_w - x)^2 + (y_w - y)^2 \tag{5.13}$$

$$\phi_w = \arctan\left(\frac{y_w - y}{x_w - x}\right) - \theta \tag{5.14}$$

The minimum time optimal control to reach the waypoint (x_w, y_w) can be formulated as a feedback control which is a function of r_w and ϕ_w. The optimal control which minimizes (5.11) is

$$u(r_w, \phi_w) = \begin{cases} -u_{max} & \text{if } (r_w, \phi_w) \in \{S_- \cup C_-\}, \\ 0 & \text{if } (r_w, \phi_w) \in S_0, \\ u_{max} & \text{if } (r_w, \phi_w) \in \{S_+ \cup C_+\} \end{cases} \tag{5.15}$$

where the five sets C_+, C_-, S_+, S_- and S_0 are

$$C_- = \{(r_w, \phi_w) : r_w \leq 2\rho_{min}\sin(\phi_w), \phi_w > 0\} \tag{5.16}$$

$$S_+ = \{(r_w, \phi_w) : r_w > 2\rho_{min}\sin(\phi_w), \phi_w > 0\} \tag{5.17}$$

$$C_+ = \{(r_w, \phi_w) : r_w \leq 2\rho_{min}\sin(-\phi_w), \phi_w < 0\} \tag{5.18}$$

$$S_- = \{(r_w, \phi_w) : r_w > 2\rho_{min}\sin(-\phi_w), \phi_w < 0\} \tag{5.19}$$

$$S_0 = \{(r_w, \phi_w) : \phi_w = 0\} \tag{5.20}$$

$\rho_{min} = v/u_{max}$ is the minimum turning radius of the Dubins vehicle. The boundaries between the sets C_+ and S_-, C_- and S_+, as well as S_- and S_+, are the so-called "switching surfaces" where the optimal control switches between the control values of each set.

Fig. 5.2 depicts all the sets, as well as the "switching surfaces" depicted as orange vector field arrows flowing along the boundaries to the location of the waypoint. The parameters of our Dubins vehicle are $u_{max} = 0.5$ rad/s and $v = 0.1$ m/s. The control values are indicated in parenthesis to the left of each set and the control along the "switching surface" S_0 is $u = 0$. The location of the waypoint corresponds to the line $r_w = 0$ and free terminal heading angle $\phi_w \in [-\pi/2, \pi/2)$.

We also discretize the analytical solution of navigating to the waypoint. To obtain the discretization, we run a simulation from each point of the discrete state space following the control (5.15) and record the time to reach the waypoint. We associate that time with the discrete cell from which the simulation started and repeat the same procedure for all points of the discrete space. For the computational domain $\mathscr{K}$ of the

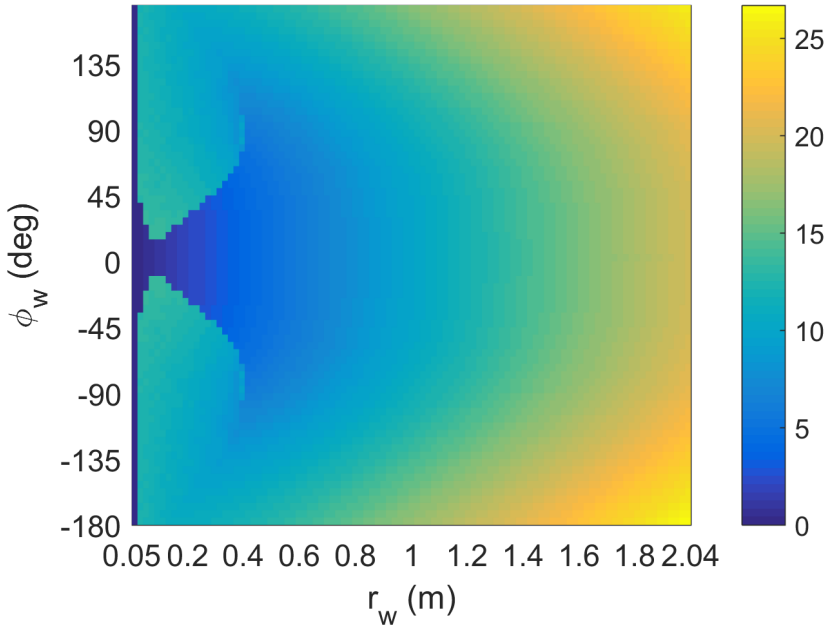


Figure 5.3: Time in seconds for a Dubinds vehicle with $v = 0.1$ m/s and $u_{max} = 0.5$ rad/s to reach the waypoint following the control (5.15).

solution we use

$$\mathscr{K} = \{[R_{min}, R_{max}] \times [-\pi, \pi - \Delta\phi_o]\} \tag{5.21}$$

which is the set bounded by the minimal $R_{min} = 0.05$ m and maximal $R_{max} = 2.04$ m distances. Since, in our problem formulation, the angles ϕ_w have a full 2π range, the computational domain is periodic, i.e., the pairs of points $(r_w^h, -\pi)$ and $(r_w^h, \pi - \Delta\phi_w)$ are next to each other. The discrete steps in $\Delta r_w = (R_{max} - R_{min})/100$ m and $\Delta\phi_w = 5$ *deg*. We stop the simulation when the Dubins vehicle has reached the minimal R_{min} distance allowed by our computational domain. The result of these computations is the time T^h presented in Fig. 5.3.

5.4 Stochastic Optimal Control to Avoid Collision with the Obstacle

This sub-problem is based on the stochastic kinematic model given by expressions (5.6) and (5.7). The model's state variables are the distance r_o and bearing angle ϕ_o, defined as

$$r_o^2 = (x_o - x)^2 + (y_o - y)^2 \tag{5.22}$$

$$\phi_o = \arctan\left(\frac{y_o - y}{x_o - x}\right) - \theta \tag{5.23}$$

where (x, y, θ) and (x_o, y_o) define a Dubin vehicle's pose and obstacle's position, respectively (see Fig. 5.1).

The set which the Dubins vehicle should avoid is $\mathscr{A}$ defined by (5.8). The optimal control u_o^* in (5.12) can be defined as the steady state solution of the Hamilton-Jacobi-Bellman (HJB) partial differential equation [94]

$$0 = \min_{u_o} \{\mathscr{L}^o H(r_o, \phi_o, u_o)\} \tag{5.24}$$

where H can be interpreted as the expected hazard to collide with the obstacle, and $\mathscr{L}^o$ is the differential operator

$$\mathscr{L}^o = b_{r_o}\frac{\partial}{\partial r_o} + \frac{1}{2}(n_{r_o}n_{\phi_o} + n_{r_o}^2)\frac{\partial^2}{\partial r_o^2} + b_{\phi_o}\frac{\partial}{\partial \phi_o} + \frac{1}{2}(n_{r_o}n_{\phi_o} + n_{\phi_o}^2)\frac{\partial^2}{\partial \phi_o^2} \tag{5.25}$$

where b_{r_o}, n_{r_o}, b_{ϕ_o} and n_{ϕ_o} are defined by (5.6) and (5.7), respectively. The solution of (5.24) can be computed using a locally consistent Markov chain discretization of

the HJB equation [94]. The discretization yields a Markov chain with control dependent transition probabilities and converts (5.24) into a dynamic programming problem which can be solved over a discrete space using so-called value iterations [175]. The value iterations result in a discrete approximation of the value function H^h and optimal control u_o^{*h}, where both are in the form of a two-dimensional lookup table. The superscript h indicates that the value function and control are computed for the discretized problem.

To discretize (5.24) in the state space, we use discrete steps Δr_o and $\Delta\phi_o$ for the discretization of r_o and ϕ_o, respectively, and the following upwind discrete approximations of derivatives in (5.24)

$$\begin{aligned}
\frac{\partial H}{\partial r_o} &\approx \frac{b^+_{r_o^h}}{\Delta r_o}\Big(H(r_o^h+\Delta r_o,\phi_o^h)-H(r_o^h,\phi_o^h)\Big)\\
&\quad-\frac{b^-_{r_o^h}}{\Delta r_o}\Big(H(r_o^h,\phi_o^h)-H(r_o^h-\Delta r_o,\phi_o^h)\Big) \qquad (5.26)\\
\frac{\partial^2 H}{\partial r_o^2} &\approx \frac{|n_{r_o}n_{\phi_o}+n_{r_o}^2|}{2\Delta r_o^2}\Big(H(r_o^h+\Delta r_o,\phi_o^h)-H(r_o^h,\phi_o^h)\Big)\\
&\quad-\frac{|n_{r_o}n_{\phi_o}+n_{r_o}^2|}{2\Delta r_o^2}\Big(H(r_o^h,\phi_o^h)-H(r_o^h-\Delta r_o,\phi_o^h)\Big) \qquad (5.27)\\
\frac{\partial H}{\partial \phi_o} &\approx \frac{b^+_{\phi_o^h}}{\Delta\phi_o}\Big(H(r_o^h,\phi_o^h+\Delta\phi_o)-H(r_o^h,\phi_o^h)\Big)\\
&\quad-\frac{b^-_{\phi_o^h}}{\Delta\phi_o}\Big(H(r_o^h,\phi_o^h)-H(r_o^h,\phi_o^h-\Delta\phi_o)\Big) \qquad (5.28)\\
\frac{\partial^2 H}{\partial \phi_o^2} &\approx \frac{|n_{r_o}n_{\phi_o}+n_{\phi_o}^2|}{2\Delta\phi_o^2}\Big(H(r_o^h,\phi_o^h+\Delta\phi_o)-H(r_o^h,\phi_o^h)\Big)\\
&\quad-\frac{|n_{r_o}n_{\phi_o}+n_{\phi_o}^2|}{2\Delta\phi_o^2}\Big(H(r_o^h,\phi_o^h)-H(r_o^h,\phi_o^h-\Delta\phi_o)\Big) \qquad (5.29)
\end{aligned}$$

where $b^+_{r^h_o} = \max[0, b_{r^h_o}]$, $b^-_{r^h_o} = \max[0, -b_{r^h_o}]$ and $b^+_{\phi^h_o}$, $b^-_{\phi^h_o}$ are defined in the same way. The superscript h indicate terms that are evaluated at the points of the discretized state space $r^{h+1}_o - r^h_o = \Delta r_o$ and $\phi^{h+1}_o - \phi^h_o = \Delta\phi_o$. After the substitution of (5.25)-(5.29) in (5.24), we move all the terms that include $H(r^h_o, \phi^h_o)$ to the left side of the expression (5.24) to obtain

$$H^h = \min_{u_o} \Big\{ p^+_{\Delta r_o} H^h(r_o + \Delta r_o, \phi_o, u_o) + p^-_{\Delta r_o} H^h(r_o - \Delta r_o, \phi_o, u_o) \\ + p^+_{\Delta\phi_o} H^h(r_o, \phi_o + \Delta\phi_o, u_o) + p^-_{\Delta\phi_o} H^h(r_o, \phi_w - \Delta\phi_o, u_o) \Big\} \tag{5.30}$$

where

$$p^\pm_{\Delta r_o} = \Delta t^h(\tilde{x}_o, u) \left(\frac{b^\pm_{r_o}}{\Delta r_o} + \frac{|n_{r_o} n_{\phi_o}| + n^2_{r_o}}{(2\Delta r^2_o)} \right) \tag{5.31}$$

$$p^\pm_{\Delta\phi_o} = \Delta t^h(\tilde{x}_o, u) \left(\frac{b^\pm_{\phi_o}}{\Delta\phi_o} + \frac{|n_{r_o} n_{\phi_o}| + n^2_{\phi_o}}{(2\Delta\phi^2_o)} \right) \tag{5.32}$$

can be interpreted as discrete Markov-chain transition probabilities from the points $(r^h_o \pm \Delta r_o, \phi^h_o \pm \Delta\phi_o)$ of the discrete space to the point (r^h_o, ϕ^h_o), and

$$\Delta t^h(\tilde{x}_o, u) = \left(\frac{|b^h_{r_o}|}{\Delta r_o} + \frac{|n_{r_o} n_{\phi_o}|}{(\Delta r_o)^2} + \frac{(n_{r_o})^2}{(\Delta r_o)^2} + \frac{|b^h_{\phi_w}|}{\Delta\phi_w} + \frac{|n_{r_o} n_{\phi_o}|}{(\Delta\phi_o)^2} + \frac{(n_{\phi_o})^2}{(\Delta\phi_o)^2} \right)^{-1} \tag{5.33}$$

where $|b^h_{r_o}| = b^+_{r_o} + b^-_{r_o}$ and $|b^h_{\phi_o}| = b^+_{\phi_o} + b^-_{\phi_o}$. $\Delta t^h(\tilde{x}_o, u)$ is the time interpolation interval which depends on the other parameters of the problem; therefore, this type of discretization is called time implicit discretization [94].

Expression (5.30) is the discrete version of (5.24) and the discrete approximation H^h of the value function H can be solved numerically using value iterations [175] starting from an initial guess for the $H^h(r_o, \phi_o)$ values.

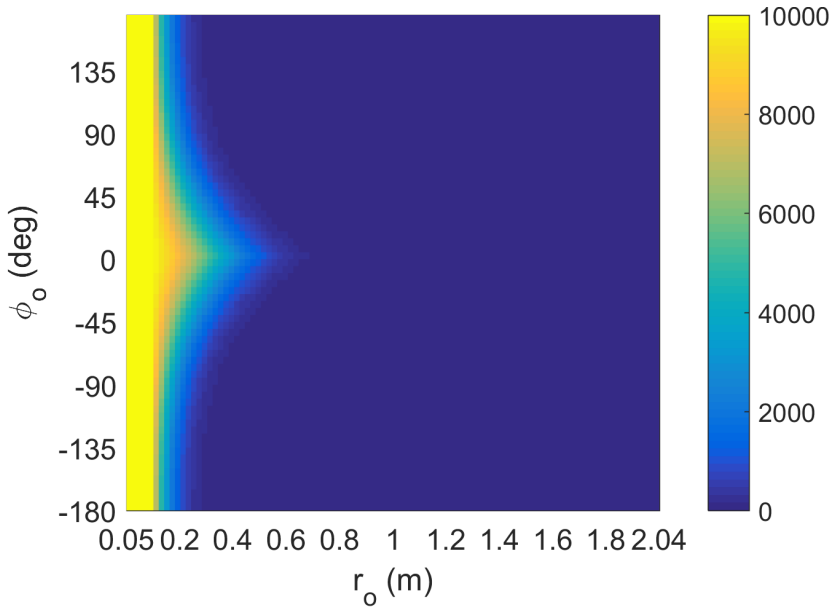


Figure 5.4: Expected Hazard for a Dubins vehicle with $v = 0.1$ m/s and $u_{max} = 0.5$ rad/s to avoid the obstacle with noise $\sigma = 0.039\ \sqrt{m/s}$ following the control (5.24).

The computational domain is (5.21) and it is discretized with the same steps as in the previous section, i.e., $\Delta r_o = \Delta r_w$ and $\Delta\phi_o = \Delta\phi_w$. For the boundary of the computational domain R_{max}, we use absorbing boundary conditions $H(R_{max}, \phi_o^h) = 0$. By this, we account for the terminal cost (5.10) of hitting the obstacle. This is an approximation since $H(r_o, \phi_o) = 0$ only for $r_o \to \infty$ therefore, we use a large enough R_{max}. The result of these computations is the expected hazard H^h presented in Fig. 5.4.

Based on the approximation, the probability of collision is

$$\mathbb{P}\{(r_o^h, \phi_o^h) \in \mathscr{A}\} = \frac{1}{M} H^h(r_o^h, \phi_o^h) \tag{5.34}$$

which is 0 for $r_o = R_{max}$.

5.5 Safe Navigation Towards a Waypoint

Once the solution for the two sub-problems has been defined, we combine these two solutions to obtain the solution of the original problem, which is to navigate towards the waypoint while avoiding collision with the obstacle.

We combine the value functions of out two optimal control solutions together to approximate the original problem's value function as

$$\tilde{V}^h(\tilde{x}) = T^h(\tilde{x}_w) + H^h(\tilde{x}_o) \tag{5.35}$$

where $\tilde{x} = (r_w, \phi_w, r_o, \phi_o)$, $\tilde{x}_w = (r_w, \phi_w)$, $\tilde{x}_o = (r_o, \phi_o)$, T^h is the time shown in Fig. 5.3 and H^h is the expected hazard shown in Fig. 5.4. Thereby, we obtain $\tilde{V}^h$ which is the approximate discrete solution to the original 4-dimensional problem of reaching the waypoint while avoiding collision with the Brownian moving obstacle.

To compute the control, we use the expected one-step look ahead. This method is similar to applying a gradient decent on the combined value functions (5.35), which gives the control:

$$u_c(\tilde{x}) = \underset{u}{\operatorname{argmin}} \left\{ \Delta t^h(\tilde{x}, u) + \sum_{\tilde{y} \in \mathcal{N}^h(\tilde{x})} p^h(\tilde{y}; \tilde{x}, u) V^h(\tilde{x}) \right\} \tag{5.36}$$

where $\tilde{y}$ corresponds to the one-step neighboring discrete state space cell locations around the current $\tilde{x}$ discrete state space cell location. The probability $p^h(\tilde{y}; \tilde{x}, u)$ corresponds to transitions from $\tilde{x}$ to $\tilde{y}$. Substituting the right-hand side of expression (5.35)

to expression (5.36), we obtain

$$u_c(\tilde{x}) = \underset{u}{\operatorname{argmin}} \left\{ \Delta t^h(\tilde{x},u) + T^h(\tilde{x}_w) + \sum_{\tilde{y}\in\mathcal{N}^h(\tilde{x})} p^h(\tilde{y};\tilde{x},u) H^h(\tilde{x}_o) \right\} \tag{5.37}$$

by noting $p^h(\tilde{y};\tilde{x},u) \approx p^h(\tilde{y}_o;\tilde{x}_o,u)\, p^h(\tilde{y}_w;\tilde{x}_w,u) \approx p^h(\tilde{y}_o;\tilde{x}_o,u)$, since the probability transitions $p^h(\tilde{y}_w;\tilde{x}_w,u) \approx 1$ in the direction where the Dubins vehicle will move in the next time step because it is a deterministic problem. Simplifying expression (5.37), we obtain

$$u_c(\tilde{x}) = \underset{u}{\operatorname{argmin}} \left\{ T^h(\tilde{y}_w) + \sum_{\tilde{y_o}\in\mathcal{N}_o^h(\tilde{x})} p^h(\tilde{y}_o;\tilde{x}_o,u) H^h(\tilde{x}_o) \right\} \tag{5.38}$$

where $p^h(\tilde{y}_o;\tilde{x}_o,u)$ corresponds to the discrete Markov-chain transition probabilities in (5.31)-(5.32), $T^h(\tilde{y}_w) = T^h(\tilde{x}_w) + \Delta t^h(\tilde{x},u)$ is the time cost at the next discrete cell location $\tilde{y}_w$ and $H^h(\tilde{x}_o)$ is the hazard of colliding with the obstacle at discrete cell location $\tilde{x}_o$.

5.6 Results

The numerical simulation results of this section are based on the Dubins vehicle model (5.1)-(5.3) and (5.4)-(5.5) with the velocity $v = 0.1$ m/s and maximum turning rate $u_{max} = 0.5$ rad/s. The noise intensity for the Brownian moving obstacle is $\sigma = 0.039\ \sqrt{m/s}$. Because of the discrete space approximations used in computing the control, we stop the simulation once the Dubins vehicle has reached the minimal distance allowed in our computational domain of 0.05 m to the waypoint (x_w, y_w), or

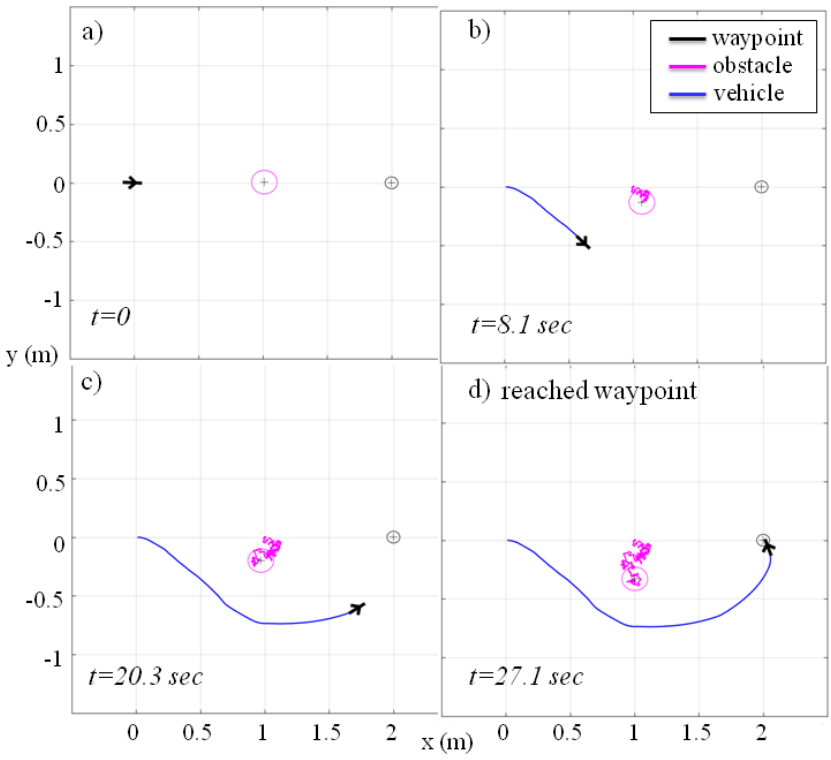


Figure 5.5: Simulation with the vehicle avoiding the stochastic obstacle to reach the waypoint.

has collided with the obstacle of the radius $d = 0.1$ m centered at (x_o, y_o). The sample time is $\Delta t = 0.1$ seconds. The parameters used for computing the expected hazard in Fig. 5.4 and the time in Fig. 5.3 are the same and are listed in expression (5.21).

Figure 5*a* depicts the initial positions of the vehicle and the obstacle. The waypoint is the black circle farther from the Dubins vehicle (aircraft symbol), whereas the obstacle is the lighter circle closer to the initial position of the Dubins vehicle. While the pure time optimal control would navigate the Dubins vehicle towards the obstacle, accounting for the expected hazard pushes the vehicle away from its path, as shown

in Fig. 5.5*b*. The Dubins vehicle navigates around the obstacle (see Fig. 5.5*c*) where under the threat of colliding with the obstacle, the Dubins vehicle keeps its distance from it and reaches the waypoint (see Fig. 5.5*d*).

To see how well this controller satisfies its criteria of avoiding collision with the obstacle and reaching the waypoint, Fig. 5.6 shows the result of a simulation in which the waypoint overlaps with the stationary obstacle, i.e. $\sigma = 0$. Figure Fig. 5.6*a* depicts the initial positions of the vehicle and obstacle and position of the waypoint, which is inside the obstacle. Figure 5.6*b* shows that the Dubins vehicle starts to circle the obstacle, but does not move away as the waypoint is attracting the vehicle closer as shown in Fig. 5.6*c*. However, the Dubins vehicle stops moving closer to the waypoint as the vehicle is moving closer to the obstacle, see Fig. 5.6*d*. The Dubins vehicle continues to circle around the obstacle and waypoint location at the distance where moving closer would increase the chance of collision and moving farther away would mean being farther from the waypoint, see Figs. 5.6*e* and f.

5.7 Conclusions

In this paper, we presented the controller for the Dubins vehicle which has the goal to reach a target location while simultaneously avoiding collision with the randomly moving circular obstacle. Since the Dubins vehicle does not know the obstacle's trajectory, this uncertainty has to be anticipated during the navigation. While this problem

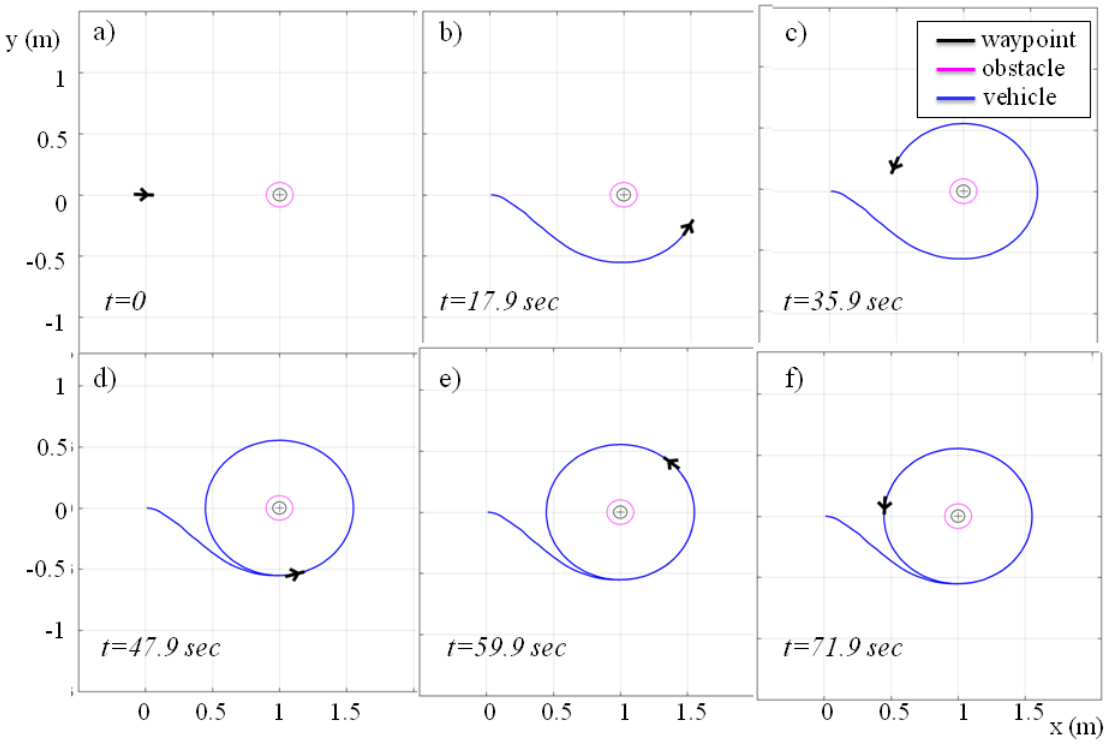


Figure 5.6: Simulation with the vehicle avoiding a stationary obstacle on top of the waypoint location.

can potentially be solved optimally using the method of value iterations, this approach is hindered by the number of state variables that should be considered. Therefore, we proposed the sub-optimal approach by dividing the original problem in two sub-problems, where each sub-problem is solved optimally. This approach has potential to scale well with the number of obstacles, which will be considered in future work.

Chapter 6

Time Efficient Inspection of Ground Vehicles by a UAV Team Using a Markov Inequality Based Rule

This chapter is a reprint of the paper

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. "Time Efficient Inspection of Ground Vehicles by a UAV Team Using a Markov Inequality Based Rule." In Distributed Autonomous Robotic Systems, pp. 95-108. Springer, Cham, 2019. (**one of three papers nominated for the best student paper award**)

6.1 Introduction

This paper is the result of an exploration to formulate the control of stochastic multi-agent systems by an integration of stochastic optimal control strategies that are designed for a pair of agents. To study such an integration, we consider a scenario with a team of N unmanned air vehicles (UAVs) that are tasked to inspect efficiently with respect to time a group of M ground vehicles. However, the navigation strategy of ground vehicles is unknown. To anticipate this uncertainty, the headings of ground vehicles are described as stochastic processes, and as the result, the UAV navigation has to be a solution of a feedback stochastic control problem.

A solution of the feedback stochastic optimal control problem for the two groups with N and M agents has to depend on the number of variables that increases quickly due to the number of combinations in which N agents can inspect M agents. This number should also account for solutions allowing that a single UAV may need to inspect multiple ground vehicles. In principle, we can formulate the minimum time stochastic optimal control using the Hamilton-Jacobi-Bellman equation and all necessary coordinates describing relative positions among vehicles. However, the computational complexity of such a solution goes quickly beyond the computational power of modern computers because of the combinatorial increase of the number of relative coordinates, as well as due to the so-called *curse-of-dimensionality* that results in the computational complexity which increases with the power of the number of relative coordinates. Con-

sequently, we propose an approach in which we first solve for the stochastic optimal control of one UAV inspecting one ground vehicle (*one-on-one*) and use this to formulate a time efficient solution for the problem of N UAVs inspecting M ground vehicles.

The minimum time stochastic optimal control of a single UAV entering the tail sector of another vehicle while safely navigating around it is presented in [125]. The computational method for this type of a *one-on-one* agent problem has been improved and used with a scalable value approximation [73] in a complex scenario of a single UAV safely intercepting a group of vehicles [124]. Here we consider a scenario of having multiple UAVs inspecting multiple ground vehicles and we assume that the UAVs are able to avoid each other, or more realistically, they are able to fly at different altitudes. Consequently, collision avoidance is not considered and the main problem lies in how to assign [34, 166] each UAV to ground vehicles, so that the inspection time for all ground vehicles is time efficient. Previously, the authors studied a related assignment problem [24], but in terms of path-planning [100].

Non path-planning, i.e., feedback-based navigations are frequently found in a line of work called sense and avoid [13], and game theoretic approaches to safe navigation [122]. They also appear in the context of pursuit-evasion games [81], including two car-like vehicles investigated in [64]. Scalable algorithms for multiple-pursuers/single-evader games have been considered in [54, 182]. Game theoretic problems for multi-pursuit and multi-evasion strategies using a hierarchical or decomposition algorithmic

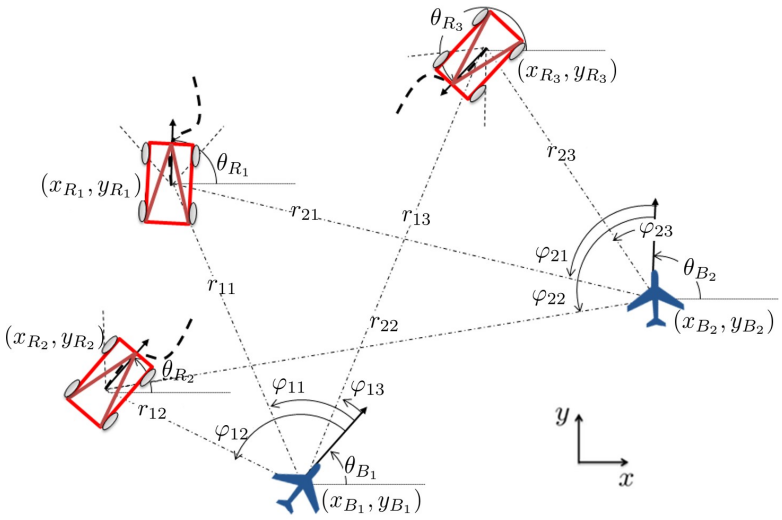


Figure 6.1: Geometry of the multi-agent/multi-target problem. θ_{B_i} and θ_{R_j} are the heading angles of the ith blue and jth red, respectfully. $\alpha_{ji} = \theta_{R_i} - \theta_{B_j}$ is the difference between the headings. ϕ_{ji} is the bearing angle of the jth blue to ith red.

approach [62, 102] are more similar to the multi-agent problem in this paper.

In this work, we approach the problem of N UAVs inspecting M ground vehicles as a Markov inequality-based switching problem which is inspired by the result [120] for one UAV inspecting multiple ground vehicles. Dealing here with N UAVs, the switching is among possible inspection assignments of UAVs to the ground vehicles to be inspected. No knowledge of the ground vehicles navigation strategy or intent is known by UAVs, therefore, each of the ground vehicles heading angles is anticipated to be a Brownian random walk. The contribution of this paper is that it solves the stochastic multi-agent problem using *one-on-one* stochastic optimal control solutions and the switching rule for the time-efficient navigation. The presented approach scales well with the number of agents and allows for real-time computations of control actions.

The paper is organized as follows. Section 6.2 discusses the problem, which stating succinctly is controlling a team of UAVs to inspect every ground vehicle in a minimum time. The stochastic optimal control of a single UAV to inspect a single ground vehicle in a minimum time is presented in Section 6.3. The time-efficient dynamic assignment problem approach is discussed in Section 6.4. Section 6.5 shows results for two UAVs and three ground vehicles, and Section 6.6 provides conclusions.

6.2 Problem Formulation

Let us consider a scenario with five agents depicted in Figure 6.1. Three of the agents, labeled with R_i, $i = 1,2,3$, are ground vehicles with equal speeds $v_{R_i} = v_R$, and we refer to them as red agents. The other two agents, B_j, $j = 1,2$, are fixed-wing aerial vehicles flying at different altitudes at equal speeds $v_{B_j} = v_B$, and we refer to them as blue agents. The constant speed assumption approximates that the UAVs cannot stop and that without energy constraints, the UAV will fly at maximum speeds to cover larger areas of interest. We assume that the speed of blue (aerial) agents is larger than the speed of red (ground) agents, $v_B > v_R$. The kinematics of the jth blue agent is described using the deterministic kinematics of a Dubins vehicle, which is

$$dx_{B_j} = v_B \cos(\theta_{B_j})dt \tag{6.1}$$

$$dy_{B_j} = v_B \sin(\theta_{B_j})dt \tag{6.2}$$

$$d\theta_{B_j} = u_{B_j}dt \tag{6.3}$$

where the couple (x_{B_j}, y_{B_j}) and θ_{B_j} describe the 2D agent position and heading angle, respectively. The control input for the vehicle is a bounded turning rate $u_{B_j} \in [-1, +1]$.

The relative position between each jth blue B_j and ith red R_i is uniquely defined based on the relative coordinates

$$r_{ji} = \sqrt{(x_{R_i} - x_{B_j})^2 + (y_{R_i} - y_{B_j})^2}, \tag{6.4}$$

$$\phi_{ji} = \arctan\left(\frac{y_{R_i} - y_{B_j}}{x_{R_i} - x_{B_j}}\right) - \theta_{B_j}, \phi \in [-\pi, \pi) \tag{6.5}$$

$$\alpha_{ji} = \theta_{R_i} - \theta_{B_j}, \alpha \in [-\pi, \pi) \tag{6.6}$$

where r_{ji} is the distance between B_j and R_i, ϕ_{ij} is the bearing angle from B_j to R_i, and α_{ji} is the difference between the R_i and B_j heading angles. The definitions of these coordinates are also illustrated in Figure 6.1. Based on these relative coordinates, we define the tail sector $\mathscr{T}_i(t)$ for each ith red agent as

$$\mathscr{T}_i(t) = \left\{ (r, \phi, \alpha) : \begin{array}{r} r \leq R_{min} \\ |\phi| \leq \phi_m \\ |\alpha| \leq \alpha_m \end{array} \right\} \tag{6.7}$$

where α_m and ϕ_m are the angles defining the tail sector width and alignments of the agents heading angles. A successful inspection is defined using (6.7) as $(r_{ji}, \varphi_{ji}, \alpha_{ji}) \in \mathscr{T}_i$, i.e., it occurs when agent B_j is in this sector behind R_i, and its heading is aligned with that of R_i.

The control problem in this paper is to find the control for the B agent team providing that the time of inspection of all ground vehicles is minimal, i.e., that the B team

visit all tail sectors of R vehicles in a minimum time. However, B agents *have no knowledge of the R ground vehicles' navigation strategy, i.e., the motion of the corresponding tail sectors*. To anticipate that uncertainty, the kinematics of each R_i, $i = 1,...M$ agent is modeled by the stochastic kinematics

$$dx_{R_i} = v_R \cos(\theta_{R_i})dt \tag{6.8}$$

$$dy_{R_I} = v_R \sin(\theta_{R_i})dt \tag{6.9}$$

$$d\theta_{R_i} = \sigma_R dw_i \tag{6.10}$$

where the position of R_i is given by x_{R_i}, y_{R_i} and the heading angle is $\theta_{R_i} = \int_0^t \sigma_R dw_i$. The latter describes that the heading angles of R agents are random walks since dw_i denotes the Wiener process increments. The scaling parameter σ_R is identical for all R agents.

A solution of the minimum time feedback optimal control problem for N blue (B) and M red (R) agents depends on the number of variables that increases quickly due to the number of combinations in which N blue agents can visit M tail sectors of R vehicles. This number should also account for solutions allowing that a single B agent may need to visit multiple R agent tail sectors. While in principle we can formulate the stochastic optimal control using the Hamilton Jacobi Bellman (HJB) equation and all necessary coordinates describing relative positions of Bs and Rs, the computational complexity of such a solution goes quickly beyond the computational power of modern computers because of the number of relative coordinates, as well as due to the so-called

curse-of-dimensionality.

We approach the problem by dividing it into (1) the minimum time stochastic optimal control problem of one B and one R agents (*one-on-one*), and (2) the problem of computing inspection assignments of B agents. This approach allows us to formulate the solution for the navigation strategy of Bs that scales well with the number of agents and guarantees that all R agents are inspected efficiently with respect to time. This guaranteed property and time efficiency are achieved by the optimality of the *one-on-one* problem and assignment updates when the assignment leads to a robust decrease of time for the inspection of all agents.

6.3 Minimum Time Stochastic Optimal Control

In this section, we deal with the scenario of a single B_j that enters the tail sector $\mathscr{T}_i$ of a single R_i in the minimum expected time, thus $i = j = 1$. To simplify the notation, in this section we will drop subscripts i and j from B_j, R_i and $\mathscr{T}_i$.

Using Îto calculus, the kinematic model of B (6.1)-(6.3), the kinematic model of R (6.8)-(6.10) and the definitions of relative coordinates between B and R (6.4)-(6.6), we can derive the following stochastic differential equations describing the evolutions of

relative positions between B and R

$$dr = (v_R \cos(\phi - \alpha) - v_B \cos(\phi))dt = b_r dt \tag{6.11}$$

$$d\phi = \left(-u_B + \frac{-v_R \sin(\phi - \alpha) + v_B \sin(\phi)}{r} \right) dt = b_\phi dt \tag{6.12}$$

$$d\alpha = -u_B dt + \sigma_R dw = b_\alpha dt + \sigma_R dw \tag{6.13}$$

The minimum expected time control u_B to reach the target set $\mathscr{T}(t)$ is the one that minimizes the cost

$$\mathscr{J}(u_B) = E\left\{ g(\tau) + \int_0^\tau 1 dt \right\} \tag{6.14}$$

with the terminal cost $g(\tau) = g(r(\tau), \phi(\tau), \alpha(\tau))$ defined as

$$g(\tau) = \left\{ \begin{array}{ll} 0 & \text{if } (r(\tau), \phi(\tau), \alpha(\tau)) \in \mathscr{T} \\ M & \text{if } (r(\tau), \phi(\tau), \alpha(\tau)) \in \mathscr{P} \end{array} \right\} \tag{6.15}$$

where the set $\mathscr{P}$ is defined as

$$\mathscr{P}(t) = \left\{ (r, \phi, \alpha): \begin{array}{r} r \leq R_{min} \\ |\phi| > \phi_m \\ |\alpha| > \alpha_m \end{array} \right\} \tag{6.16}$$

The cost function $\mathscr{J}$ is constructed to yield the optimal control u_B that minimizes the time for B to reach the target set $\mathscr{T}$ and avoids configurations in which B is in the proximity of R, but not in the target set $\mathscr{T}$. This is expressed by the definition of terminal cost $g(\tau)$ in which there is a large positive penalty $M >> 0$ for the set $\mathscr{P}$ and no penalty for reaching the set $\mathscr{T}$.

The cost function $\mathscr{J}(u_B)$ gives rise to the stochastic HJB equation defining the evolution of the cost-to-go function $U = U(r, \varphi, \alpha)$ for the optimal control

$$0 = \min_{u_B} \left\{ b_r \frac{\partial U}{\partial r} + b_\phi \frac{\partial U}{\partial \phi} + b_\alpha \frac{\partial U}{\partial \alpha} + \sigma^2 \frac{\partial^2 U}{\partial \alpha^2} + 1 \right\} \tag{6.17}$$

with the two boundary conditions $U = 0$ for all $(r, \phi, \alpha) \in \mathscr{T}$ and $U = M$ for all $(r, \phi, \alpha) \in \mathscr{A}$. The solution of the HJB equation yields the cost-to-go function U and the corresponding optimal state feedback control $u_B = u_B(r, \phi, \alpha)$. Once the optimal control is computed, it can be used to compute the expected time $V = V(r, \varphi, \alpha)$ to reach the target under the optimal control by computing the steady-state solution of the backward Kolmogorov (BK) equation

$$0 = b_r \frac{\partial V}{\partial r} + b_\phi \frac{\partial V}{\partial \phi} + b_\alpha \frac{\partial V}{\partial \alpha} + \sigma^2 \frac{\partial^2 V}{\partial \alpha^2} + 1 \tag{6.18}$$

with the boundary condition $V = 0$ for all $(r, \phi, \alpha) \in \mathscr{T}$ and reflective boundary condition elsewhere on the boundary of the solution domain.

The solution of the optimal control problem used in this paper is based on the so-called locally consistent Markov chain discretization of the HJB equation. The discretization yields a Markov chain with control u_B-dependent transition probabilities while the problem of solving the HJB equation is converted into a discrete state space dynamic programming problem that can be solved using value iterations [94]. Further details about the numerical method can be found in [124] and [125]. In both of these papers, the controllers have been implemented and tested with ground robots as a part

of different control problems. Similar controllers have been used for the navigation of a small UAV in the presence of stochastic winds [8]. A general explanation of the method for the control of nonholonomic vehicles is given in [38]. The method has also been used for target tracking problems [12] and flight tested with real UAVs [119].

Once the control is computed, we can use it to compute the solution of the Kolmogorov equation (6.18) to obtain expected time V, i.e., its discrete space numerical representation. Due to the similarity of the Kolmogorov equation (6.18) with (6.17), the computations are based on the same discretization scheme and value iterations [94], except that instead of *min* operator, the value iterations are based on already computed optimal control.

The units in (6.4)-(6.6), which we used to compute the numerical optimal control in the example of this paper, are normalized so that all the angles are in radians and the velocities $v_B = 0.1$ and $v_R = 0.05$. The noise scaling parameter $\sigma_R = 10\pi/180$ and the maximum turning rate of the blue $u_{max} = 0.5$. The tail sector (6.7) to be reached by the blue is defined by $R_{min} = 0.05$, $\phi_m = 10\pi/180$, and $\alpha_m = 20\pi/180$. The computational domain is

$$\mathscr{K} = \{[R_{min}, R_{max}] \times [-\pi, \pi - \Delta\phi] \times [-\pi, \pi - \Delta\alpha]\} \tag{6.19}$$

with $R_{max} = 2.04$, and is discretized with the steps $\Delta r = (R_{max} - R_{min})/99 \approx 0.0201$ and $\Delta\phi = \Delta\alpha = 5\pi/180$ in the direction of r, ϕ, α state space variables. Since the angles ϕ and α in our problem formulations have full ranges, in the discretized computational

domain, the pairs of points $(r^h, -\pi, \alpha^h)$ and $(r^h, \pi - \Delta\phi, \alpha^h)$, as well as $(r^h, \phi^h, -\pi)$ and $(r^h, \phi^h, \pi - \Delta\alpha)$ are next to each other, i.e., we use periodic boundary conditions along the ϕ and α state space variables. Other boundary conditions of the discrete approximation based on the locally consistent Markov chain approximation method take into account (6.15) and the boundary conditions of (6.7) and (6.16).

6.4 Time efficient dynamic assignment

A general formulation of the assignment problem of which B_j, $j = 1, ...N$ agent should inspect R_i, $i = 1, ...M$ should take into account that a single B_j may be assigned to a sequence of multiple R_i agents and that all R_i agents are assigned to at least one B_j agent. This type of a problem can be formulated as an optimization problem on a graph. In the most general case, the graph nodes would be B and R agents and edges between them would have an associated time to travel from any B agent to any R agents, as well as between all R agents, so that the inspection of multiple R agents can be accounted for. The solution would be a sequence of R agents that should be visited by each B_j, $j = 1, ...N$. However, in this paper we assume that R agents are moving stochastically and, consequently, the travel times between all nodes, including those corresponding to the edges among R_i agents are changing stochastically. These changes create a major challenge for the solution.

To address the assignment problem, we consider the optimization on a graph in

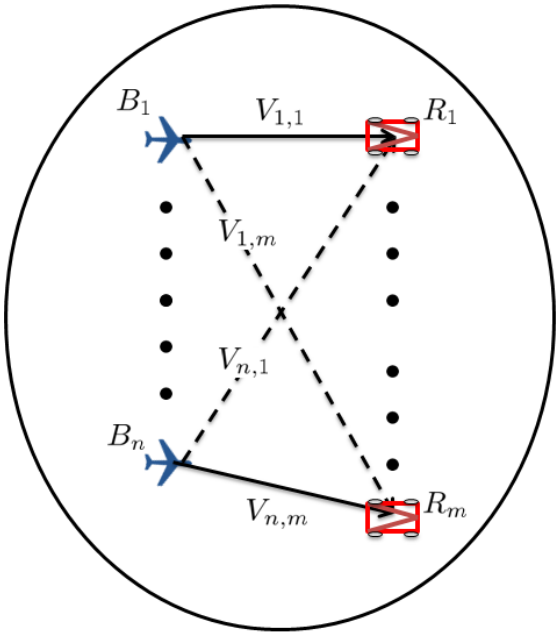


Figure 6.2: Assignment graph problem, where $V_{i,j}$ denotes the expected time of B_j inspecting R_i. The dashed lines denote a possible assignment and the solid lines denote an assigned agent-task pair.

which there are no edges among R_i agents. The graph is depicted in Fig. 6.2 and its edges are associated with the expected times $V_{j,i}(t) = V(r_{ji}(t), \varphi_{ji}(t), \alpha_{ji}(t))$ of B_j reaching $\mathscr{T}_i$, i.e., inspecting R_i agent. Note that the expected time V is obtained as the solution of the *one-on-one* problem in the previous section and that $V_{j,i}$ changes in time due to the change of agents relative positions. Given these expected times, we propose an assignment that at $t = 0$ minimizes the expected time to the inspection of the last R_i, i.e., the longest expected time $V_{j,i}$ of the assignment which is

$$C_0 = \min_{\mathscr{A}} \left\{ \max_{j \in \mathscr{N}_0} \{V_{j,i}(0) z_{j,i}(0)\} \right\}, \; z_{j,i} \in \{0,1\} \tag{6.20}$$

$$j \in \mathscr{N}_0 \subseteq \{1,2,...N\}, i \in \mathscr{M}_0 \subseteq \{1,2,...M\} \tag{6.21}$$

$$\text{subject to} \begin{cases} \sum_{i \in \mathscr{M}_0} z_{j,i}(0) = 1, \;\; \text{for all } i, \text{ if } |\mathscr{N}_0| \leq |\mathscr{M}_0| \\ \sum_{j \in \mathscr{N}_0} z_{j,i}(0) \geq 1 \;\; \text{for all } j, \text{ if } |\mathscr{N}_0| > |\mathscr{M}_0| \end{cases} \tag{6.22}$$

where the assignment variable $z_{j,i} = 1$ if B_j is assigned to inspect R_i, otherwise, $z_{j,i} = 0$. Sets $\mathscr{N}_0$ and $\mathscr{M}_0$ contain indexes of all B and R agents, respectively, that are included in the assignment, and $|\mathscr{N}_0|$, $|\mathscr{M}_0|$ denote cardinal numbers of these sets. While the constraints (6.22) are specific and define that each B agent has to be assigned to at least one R agent, the definition of sets $\mathscr{N}_0$ and $\mathscr{M}_0$ allow for a flexibility that will be used in the text below. The minimization is over a finite set of all possible assignments $\mathscr{A} = \{A_1, A_2, ...\}$, where each assignment A_k is uniquely defined by the assignment variables $z_{j,i}$ and can be depicted as the graph in Fig. 6.2 with all edges corresponding to $z_{j,i} = 0$ removed.

Let us define the cost $C_k(t)$ of the assignment A_k evaluated at time t

$$C_k(t) = \max_{j \in \mathcal{N}_0} \{V_{j,i}(t) z_{j,i}(t)\},\ i \in \mathcal{M}_0 \tag{6.23}$$

Since R agents are moving stochastically, it is obvious that the initial configuration with cost C_0 defined by the solution of (6.20) may be inferior to any other configuration A_k for which $C_k(t) < C_0$. To pursue the idea of optimality at every time instant t, one may think about solving the optimization (6.20) at every time instant (greedy approach). However, note that the navigation strategy is based on the minimization of expected times to the target sets $\mathcal{T}_i$, therefore, there is a non-zero probability for the increase of times to target sets. Because of that, although there may exist multiple assignments that result in the inspection of all R agents, once we start switching among them using the greedy approach, it can result in an infinite sequence of assignment switchings without ever inspecting any R agent. This is a well-known characteristic of the so-called *hybrid systems*.

To resolve the problem of an infinite sequence of switchings, here we propose to use the switching rule that was presented and analyzed in [120]. The rule is that if at time t

$$\frac{\min\{C_k(t), C_k(\tau^s)\} - C_m(t)}{\min\{C_k(t), C_k(\tau^s)\}} \geq p,\ \ p \in (0,1) \tag{6.24}$$

then the switching from the assignment A_k to A_m takes place. In this rule, k is the index of the current assignment, $\tau^s < t$ is the time at which the assignment becomes A_k and

s is the counter of switchings from $t = 0$, therefore, after the switching $\tau^{s+1} = t$ and s is incremented. If τ_F^m defines the time to inspect all R agents, then the above rule guarantees

$$Pr\{\tau_F^m < \min\{C_k(t), C_k(\tau^s)\}\} \geq p \tag{6.25}$$

which means that the switching happens only if it provides that the time τ_F^m is with a probability p shorter than the smallest of the current expected time to inspect all R agents and the expected time to inspect all R immediately after τ^s, i.e., the last switching of the assignment.

The above rule is introduced and analyzed in [120] for the case of a single B and multiple R agents until the first R agent is inspected and relies on the existence of a threshold $\underline{C} > 0$, so that the switching stops once $C(\tau^s) \leq \underline{C}$. The threshold $\underline{C}$ is the smallest expected time in the discrete space of our numerical stochastic optimal control solution. All these assumptions hold in the problem at hand, however, we have to deal with an additional feature of the problem, which is that at the time instants when a single B reaches the tail sector of a single R, other Bs are still trying to reach R based on their assignment. At this point, there are multiple ways the navigation of B can proceed. For example, B can be excluded from further considerations and be navigated towards a target at a far distance, or it can be assigned to another R. In the latter case, it may happen that multiple Bs can be assigned to a single R. All of these possibilities are covered by expressions (6.21) and (6.22) of the initial optimization. Therefore, at

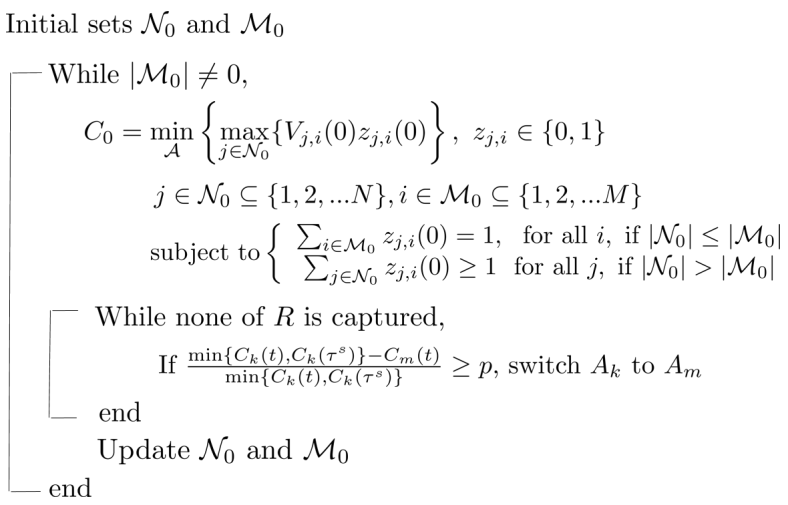


Figure 6.3: Assignment algorithm: Set $\mathscr{N}_0$ and $\mathscr{M}_0$ of available B and R agents, respectively; the number of agents in each set is $|\mathscr{N}_0|$ and $|\mathscr{M}_0|$; C_k is the cost of the assignment A_k and is defined by (6.23).

the time the first B_j reaches the target set $\mathscr{T}_i$ that corresponds to R_i, we propose to reset the time to $t = 0$, update $\mathscr{N}_0$ and $\mathscr{M}_0$, solve again (6.20) and proceed with the use of the switching rule. The rule guarantees that the taget set of the next R will be reached in a finite expected time. Summing the times until reaching $\mathscr{T}_i$ and the target set of the next R, we can conclude that two Rs are inspected in a finite expected time. Following this reasoning, it is clear that the proposed approach guarantees the inspection of all Rs in a finite expected time.

In summary, after every inspection of R we recompute the optimal assignment based on the current positions of agents (greedy approach). Following this, the assignment is updated based on the switching rule that guarantees the inspection of the next R agent in a finite expected time.

6.5 Example

To illustrate our results, we use an example with two UAVs (blue agents) and three ground vehicles (red agents). The velocity of UAVs is $v_B = 0.1$ and the velocity of ground vehicles is $v_R = 0.05$. These and all other problem parameters for the example are provided in Section 6.3, before and after (6.19), defining the computational domain of the *one-on-one* stochastic optimal control solution. Once the numerical solution and the expected time have been computed, we search over the discrete space of the solution to find the minimal non-zero expected time for a UAV to inspect a ground vehicle. This value defines the threshold $\underline{C} = 1.649$sec. Once the threshold is reached, we stop the switching assignments using the rule (6.24). The switching assignment rule parameter in the example is $p = 0.05$ and the rule defines switching among possible assignments.

In this example, we can identify six possible assignments that are labeled as A_i $i = 1,2...6$ and depicted in Fig 6.4. Beyond these assignments, the only other possible assignments are those in which a single ground vehicle is assigned to both UAVs, and we label all of them with the single label A_0. Given the small number of possible assignments, the optimization (6.20) can be performed by the evaluation of each assignment and the selection of the one with the smallest cost.

After running multiple numerical simulations, we selected one which illustrates well the approach presented in this paper.

The simulation in Fig.6.5A starts with the two blue UAVs behind the three red

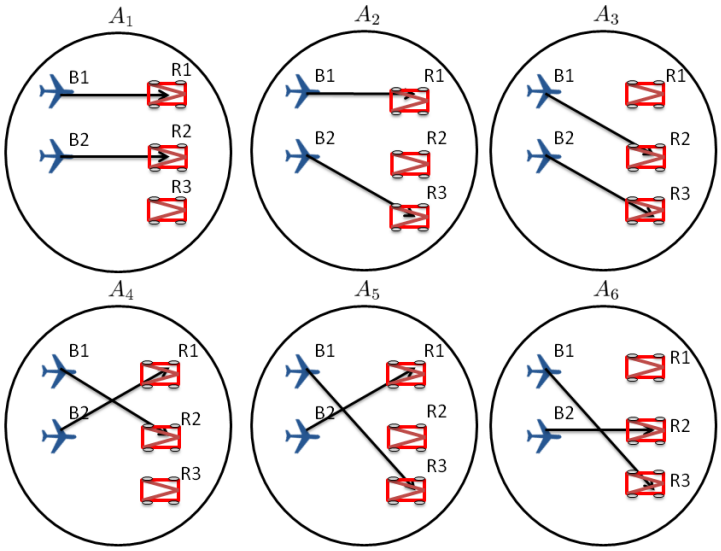


Figure 6.4: Possible assignments: each assignment is depicted by the lines connecting the UAVs ($B1$, $B2$) with the ground vehicles ($R1$, $R2$, $R3$). Beyond these assignments, those in which a single ground vehicle is assigned to both UAVs are labeled by the single label A_0.

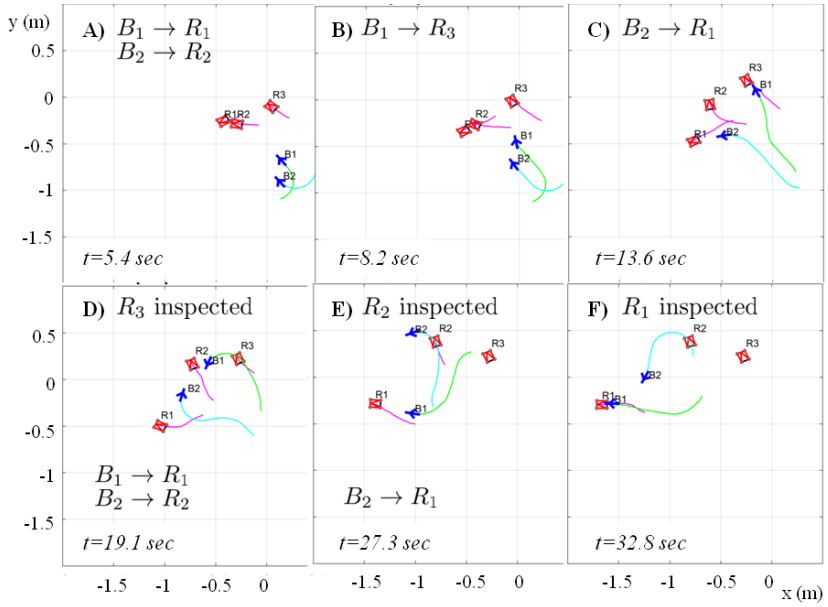


Figure 6.5: Simulation in which the two UAVs switch inspect the three ground vehicles. The progress of time is from A to F. The ground vehicles R_3, R_2, R_1 have been inspected at $t = 14.6$, $t = 23.7$, $t = 32.8$, respectively.

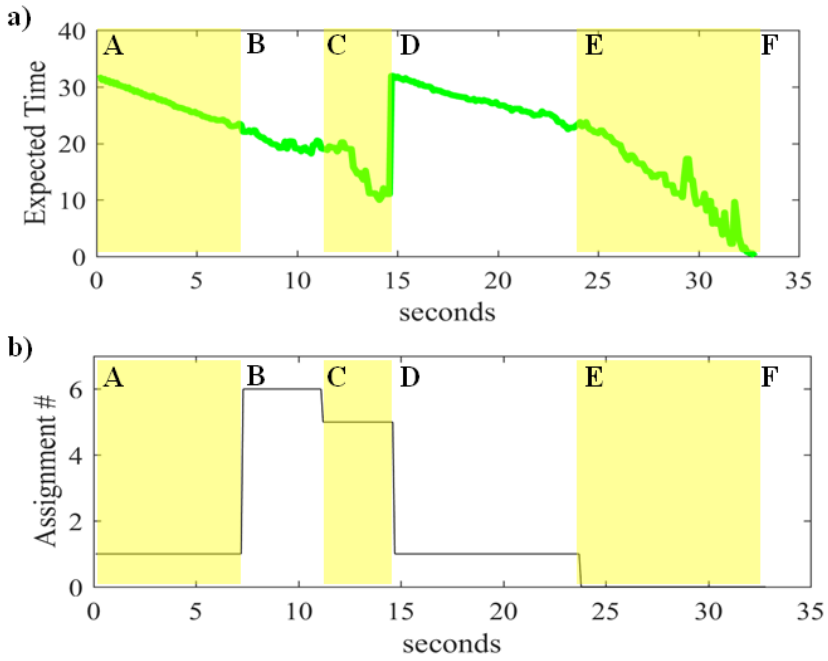


Figure 6.6: a) The cost of the current assignment A_i (see Fig. 4), which is the longest expected time of the assignment; b) the current assignment.

ground vehicles, where B_1 is assigned to inspect R_1 and B_2 to inspect R_2, i.e., the current assignment is A_1 as shown in Fig. 6.4. In Fig.6.5B, B_1 switches its assignment to R_3 and after this, B_2 switches to R_1 (see Fig. 6.5C). This assignment lasts until R_3 has been inspected by B_1 at $t = 14.6$sec (see Fig. 6.5D) and is no longer of interest to B_1 or B_2.

At the beginning of time interval D, a new optimal assignment is computed for B_1 and B_2. Because of that, we can observe a positive jump in the assignment cost, which is the longest expected time of the assignment. The new assignment is that B_1 inspects R_1 with the expected time to inspection of 31.7sec, and B_2 inspects R_2 with the expected time to inspection of 28.1sec; therefore, the cost of the assignment is 31.7sec.

This is the optimal assignment since the alternative assignment has the cost of 37.3sec, which results from the expected times of 37.3sec for B_1 inspecting R_2 and 11sec for B_2 inspecting R_1. Following the optimal assignment, R_2 has been inspected by B_2 at $t = 23.7$ (see Fig. 6.5E) without any switchings. Following this, the only assignment is A_0 in which both B_1 and B_2 inspect the remaining ground vehicle R_1. Figure 6.6*a* shows that R_1 has been inspected by B_1 at $t = 32.8$sec at the end of the time interval F.

6.6 Conclusions

In this paper we presented the control design for N UAVs tasked to perform the time efficient inspection of M ground moving vehicles. The navigation and intent of each ground vehicle are unknown, therefore, the uncertainty of its navigation has to be anticipated in the navigation of each UAV. The controller for each UAV to inspect each ground vehicle is based on the minimum time stochastic optimal control. This *one-on-one* vehicle optimal control solution is used to compute the expected time of the inspection. We further use that expected time to formulate the assignment problem of deciding what ground vehicle each UAV should inspect. We formulate it as the optimization problem of minimizing the expected time to inspect all ground vehicles. Since the ground vehicles have uncertain trajectories, the optimal assignment may need to be recomputed. However, the recomputing may result in an indefinite sequence of assignment updates without the UAVs ever inspecting all ground vehicles. To address

that, we update assignments with the Markov inequality rule. While the rule prevents the possibility of indefinite changes of assignments, it also updates an assignment if it leads to a statistically significant improvement of the expected time of the inspection. The proposed approach was illustrated by the numerical example with two UAVs and three ground vehicles. Our future work will try to address the complexity of the initial assignment optimization (6.20) and distributed implementation of the presented approach which will be able to deal with limited communication among UAVs.

Chapter 7

Scalable Markov Chain Approximation for a Safe Intercept Navigation in the Presence of Multiple Vehicles

This chapter is a reprint of the paper

- Munishkin, Alexey A., Araz Hashemi, David W. Casbeer, and Dejan Milutinović. "Scalable markov chain approximation for a safe intercept navigation in the presence of multiple vehicles." Autonomous Robots 43, no. 3 (2019): 575-588.

7.1 Introduction

In the context of this paper, safe navigation is the one providing that a vehicle that moves along its path (a) avoids collisions with other vehicles; (b) performs maneuvers to reach a safe configuration with regard to other vehicles; and (c) navigates while taking into account uncertainties in the trajectories of other vehicles.

To study the concept of safe navigation and propose a design method, we introduce in this paper an unmanned aerial vehicle scenario in which a single fixed wing vehicle is tasked to intercept one of multiple other vehicles, avoid collisions and any unsafe positions from which other vehicles can enter its regions of vulnerability. This scenario emphasizes that safety is not only about avoiding collisions, for example, in a car traffic case, the presence of a vehicle in the car driver's blind spot does not immediately lead to a collision, but creates a threat for the car safety, and for the safety reasons, the car driver would try to avoid such configurations. Our multiple-aerial vehicle scenario is related to work on collision-free navigation of mobile robots in complex cluttered environments [77], i.e., collision avoidance in multi-robot and swarm like systems [4, 131, 186], but it accounts as well for safety consideration that is beyond the risk of collision only.

The study of navigation against threats created by other vehicles is tightly interwoven with the development of game theory [82] and the two-target game problem [15], [63],[64]. The game includes two vehicles that navigate around each other until one of the vehicles, the winner of the game, enters its target set. A stochastic variant of

such two-target games is considered in [192] and [193]. These and other earlier theoretical works have been surveyed in [69]. Other lines of work have been focused on applications [50], as well as various other approaches to the problem [84], [115], [181].

In all these works, the two vehicles are opponents with the intent to harm the other vehicle and win the game. Obviously, the safe navigation inspired by the game theory would essentially mean that every vehicle in the surrounding is considered as an adversary, which is the worst case scenario, and it would result in conservative navigation strategies. Therefore, in this paper we propose a stochastic approach to safe navigation.

In the proposed stochastic approach, we anticipate that vehicles in the surroundings of a safely navigated vehicle *may, or may not* have bad intents. The lack of information about their navigation is modeled by a stochastic process and safety is addressed by a computationally defined avoidance set. In the deterministic problems [79] the boundaries between reachable and unreachable state space regions are sharp and avoidance set can be computed using the deterministic optimal control approach. However, in the stochastic problems, reachable and unreachable regions are defined in terms of probability and without sharp boundaries. In our preliminary work [125], we introduced the concept of avoidance set based on expected times. *The novelty of the work presented here is in an iterative algorithm for the avoidance set computations. Then we use the avoidance set to compute the solution for a safe vehicle navigation in the proximity of a single vehicle and propose an approach to apply this one-on-one vehicle safe naviga-*

tion to the multiple vehicle case, which scales well with the number of vehicles, and is therefore suitable for real time applications.

Extending the game theory or optimal control solution to multiple agents is difficult because of the so-called curse-of-dimensionality, due to the number of agents. In the discrete domain, there has been extensive work in solving game theory problems on graphs [2], which are traditionally called cops and robbers, and in [180] a scalable solution of the game was proposed. For the continuous domain, [143] provides a framework for combining the kinematic models of various pursuing agents for a real-time implementation of a chase and search problem. In [45], a hierarchical game extension with a finite time look ahead to the stochastic setting has been proposed, while in [54], a game theory problem is partitioned into smaller problems that are then solved separately, and the solution of the original problem is determined as the lower bound of the smaller problems. However, none of these works considered the two target problem in a multi-vehicle scenario.

Our attempt to deal with multi-agent, two-target problems and safe navigation is based on the stochastic optimal control solution of the one-on-one vehicle problem, which is to some extent similar to [131, 186] and other works that use Lyapunov functions to achieve a collision free motion. Instead of guessing a suitable Lyapunov function, in our approach [6, 12] we numerically compute it as the value function resulting from the solution of stochastic optimal control, which is tightly connected with the

nonholonomic kinematics of the vehicles, as well as anticipated uncertainties. In our preliminary work along this line in [73], we found that the value function of the one-on-one vehicle problem can be represented as a sum of the expected time and hazard function components. In the work presented here, the computed components are integrated in the navigation to replicate the performance of the one-on-one vehicle solution when the safely navigated vehicle is close to its goal and the position of other vehicles is irrelevant.

The paper is organized as follows. Section 7.2 discusses the problem we are solving with multiple Dubins vehicles. In Section 7.3 we present the iterative algorithm for computing the avoidance set and develop an optimal control strategy for the Blue agent to navigate in the presence of a single Red agent. Then, we extend in Section 7.4 the problem to multiple Red agents and a single faster Blue agent, which navigates to enter the tail sector of one of the Red agents while avoiding positions from which any of the Red agents can enter its tail. After our discussion on the optimal control, we discuss a method of simplifying the computations to provide scalability and preserving optimality in the limiting case of a single Blue chasing a single Red agent. Our results are illustrated by a simulation in Section 7.5, and a robot experiment in Section 7.6. Section 7.7 gives conclusions.

7.2 Problem Formulation

Consider a scenario with a single blue agent B and N red agents R_1, R_2, ... R_N, as depicted in Figure 7.1. *Our goal is to design a control policy for B, which will allow it to navigate into the tail (vulnerable position) of one of the red agents as quickly as possible while simultaneously avoiding collisions with the red agents.*

Agent B has Dubins vehicle kinematics described by

$$dx_B = v_B \cos\theta_B dt \tag{7.1}$$

$$dy_B = v_B \sin\theta_B dt \tag{7.2}$$

$$d\theta_B = u_B dt \tag{7.3}$$

where (x_B, y_B) is the vehicle's position, v_B is the velocity and θ_B is the heading angle. The control variable $u_B \in \mathscr{U}$ is the heading rate which is bounded and takes the value in the set $\mathscr{U} = [-u_{max}, u_{max}]$.

B is cognizant of its own state (x_B, y_B, θ_B), as well as each red agent's position and heading angle $(x_{R_i}, y_{R_i}, \theta_{R_i})$, $i = 1...N$. However, the intent of the red agents is unknown to B. To account for this uncertainty each red agent, R_i, is modeled as a Dubins vehicle with stochastic heading rate as follows:

$$dx_{R_i} = v_R \cos\theta_{R_i} dt \tag{7.4}$$

$$dy_{R_i} = v_R \sin\theta_{R_i} dt \tag{7.5}$$

$$d\theta_{R_i} = \sigma_R dw_i \tag{7.6}$$

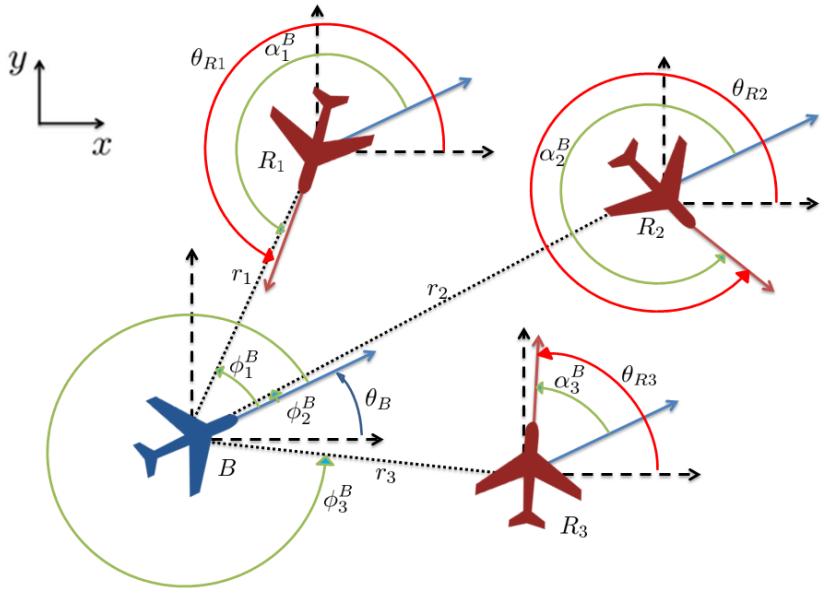


Figure 7.1: Geometry of the multi vehicle tail chase problem. θ_B, θ_{R_i} are the heading angles of the blue (faster) agent B and the red (slower) agents R_i. The relative coordinates are the distance r_i^B, the alignment angle $\alpha_i^B = \theta_{R_i} - \theta_B$ and the bearing angle $\phi_i^B = \psi_i^B - \theta_B$.

where v_R is the velocity ($v_R < v_B$), σ_R is the scaling parameter, and dw_i, $i = 1...N$, are standard, unit intensity, mutually independent Wiener processes. Hence, for an infinitesimal time-step dt, R_i's change in heading $\theta_{R_i}(t+dt) - \theta_{R_i}(t)$ has a normal distribution with a mean zero and variance $\sigma_R^2 dt$. Therefore, the parameter σ_R describes the agility of the vehicle.

The objective of B is more easily expressed mathematically using the relative coordinates $\mathbf{X}_i = (r_i^B, \phi_i^B, \alpha_i^B)^T$ for each R_i, where r_i^B is the distance, ϕ_i^B is the bearing angle and α_i^B is the alignment angle as depicted in Fig 7.1. These relative coordinates are

given by

$$r_i^B = \sqrt{(x_{R_i} - x_B)^2 + (y_{R_i} - y_B)^2} \tag{7.7}$$

$$\phi_i^B = \psi_i^B - \theta_B \tag{7.8}$$

$$\alpha_i^B = \theta_{R_i} - \theta_B \tag{7.9}$$

By applying the Itô's Lemma to the relative coordinates (7.7)-(7.8) and kinematic models (7.1)-(7.6), we obtain the dynamics of the relative coordinates as

$$dr_i^B = b_{r_i}dt;\;\; d\phi_i^B = b_{\phi_i}dt;\;\; d\alpha_i^B = b_{\alpha_i}dt + \sigma_R dw_i \tag{7.10}$$

where

$$b_{r_i} \;=\; v_R\cos(\phi_i^B - \alpha_i^B) - v_B\cos\phi_i^B;\;\; b_{\alpha_i} = -u_B \tag{7.11}$$

$$b_{\phi_i} \;=\; -u_B + \frac{1}{r_i^B}(v_B\sin\phi_i^B - v_R\sin(\phi_i^B - \alpha_i^B)). \tag{7.12}$$

Using the vector of relative coordinates $\mathbf{X}_i$ and $\mathbf{b}(\mathbf{X}_i, u_B) = (b_{r_i}, b_{\phi_i}, b_{\alpha_i})^T$, we can rewrite (7.10) in the vector form as

$$d\mathbf{X}_i = \mathbf{b}(\mathbf{X}_i, u_B)dt + (0,0,\sigma_R)^T dw_i. \tag{7.13}$$

With respect to each agent R_i, the goal of B is to reach the *target set*

$$\mathscr{T}_i = \left\{\mathbf{X}_i : \mathbf{X}_i = (r_i^B, \phi_i^B, \alpha_i^B)^T, r_i^B \in (\underline{r}, \bar{r}], \phi_i^B \in (-\bar{\phi}, \bar{\phi}], \alpha_i^B \in (-\bar{\alpha}, \bar{\alpha}]\right\}, \tag{7.14}$$

for some $\underline{r}, \bar{\phi}, \bar{\alpha} > 0$ while avoiding the collision set

$$\mathscr{C}_i = \{\mathbf{X}_i : \mathbf{X}_i = (r_i^B, \phi_i^B, \alpha_i^B)^T, r_i^B \leq \underline{r}\}. \tag{7.15}$$

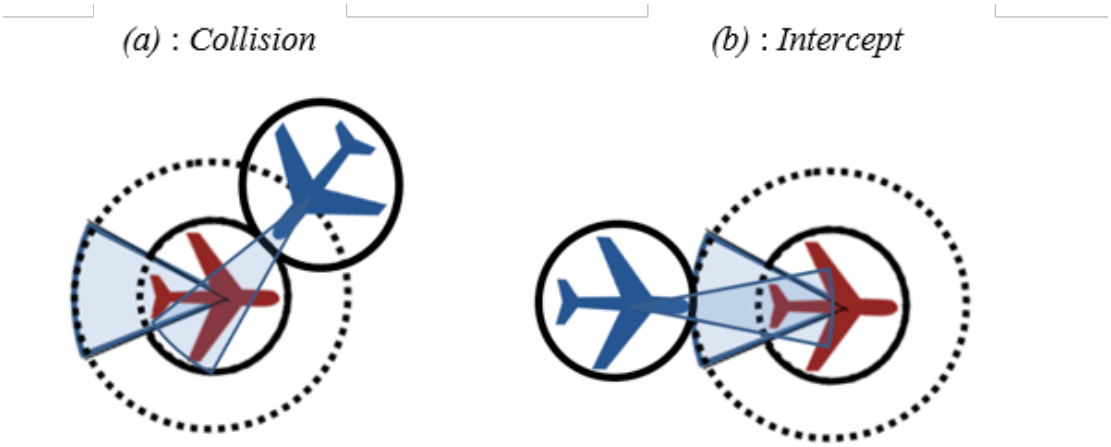


Figure 7.2: The tail sector of the red vehicle is the target tail sector of B (angular width ϕ). The sector in front of B (angular width α) indicates the range in which B and the red vehicle should be aligned. (a) A collision occurs when B reaches the proximity of the red vehicle (thick arc line). (b) An intercept occurs when B is in the tail sector of the red vehicle and aligned with the red vehicle.

The target set $\mathscr{T}_i$ requires not only that B is in the tail sector of R_i at the distance r_i^B between the *collision distance* $\underline{r}$ and the "length" of the tail $\bar{r}$, but also that its heading θ_B is aligned in the general direction with R_i's heading θ_{R_i}, as described by the relative heading bounds $\underline{\alpha}$ and $\bar{\alpha}$. The collision set $\mathscr{C}_i$ describes a ball about R_i which B must avoid. A depiction of these geometries is shown in Fig. 7.2.

Since the intent of R_i is unknown, and it *may, or may not* have the intent to enter the tail (vulnerable position) of B, it is also reasonable for B to avoid configurations which facilitate R_i reaching its tail. Any such configuration would be considered unsafe by B. We define the set of unsafe configurations $\bar{\mathscr{I}}_i$ with respect to R_i as the set of relative coordinates from which the expected time T_{B_i} for B to enter the tail of R_i is longer or equal to the expected time T_{R_i} for R_i to enter the tail of B, i.e., $\bar{\mathscr{I}}_i = \{\mathbf{X}_i : T_{B_i} \geq T_{R_i}\}$.

Therefore, with regard to the single R_i, the set that B should avoid, i.e., the *avoidance* set, is $\mathscr{A}_i = \mathscr{C}_i \cup \bar{\mathscr{T}}_i$.

Let us define the set $\mathscr{G}_i \subset \mathbb{R}^3$ of B's relative coordinates to the red agent R_i, which excludes interiors of the target $\mathscr{T}_i$ and avoidance set $\mathscr{A}_i$, i.e., $\text{int}(\mathscr{A}_i)$ and $\text{int}(\mathscr{T}_i)$, respectively. In other words,

$$\mathscr{G}_i = (\mathbb{R}_+ \times [-\pi, \pi) \times [-\pi, \pi)) \setminus (\text{int}(\mathscr{A}_i) \cup \text{int}(\mathscr{T}_i)) \tag{7.16}$$

Let τ_i be the time when B first enters the avoidance set $\mathscr{A}_i$ or target set $\mathscr{T}_i$. In other words, $\tau_i = \inf\{t : \mathbf{X}_i(t) \in \partial\mathscr{G}_i\}$ is the time when B reaches the boundary $\partial\mathscr{G}_i$.

For any admissible pure Markov control policy u_B and any initial condition $\mathbf{X}_i(0) \in \text{int}(\mathscr{G}_i)$, the drift and diffusion terms of (7.13) are Lipschitz for $\mathbf{X}_i \in \text{int}(\mathscr{G}_i)$ and thus (7.13) has a unique solution $\mathbf{X}_i(t) \in \text{int}(\mathscr{G}_i)$ (in the sense of distribution) until the time point τ_i when we can assume that the process stops; therefore, τ_i is also called the *stopping time*. The generator corresponding to the relative dynamics (7.13) for the control policy u_B is given by the differential operator

$$\mathscr{L}^{u_B} f(\mathbf{X}_i) = \mathbf{b}(\mathbf{X}_i, u_B)^T (\nabla_{\mathbf{X}_i} f) + \frac{\sigma_R^2}{2} \frac{\partial^2 f}{(\partial [\mathbf{X}_i]_3)^2} \tag{7.17}$$

where $[\cdot]_3$ denotes the third component of $\mathbf{X}_i$, which is α_i, $f : \mathscr{G}_i \to \mathbb{R}$ and $f \in C^2$ in its domain $\mathscr{G}_i$.

The domain $\mathscr{G} \subset \mathbb{R}^{3N}$ of the control policy u_B, which takes into account all red agents, is $\mathscr{G} = \prod_{i=1}^N \mathscr{G}_i$. The configuration of all agents at a time t is defined by a vector

$X(t) \in \mathscr{G}$ as $\mathbf{X}(t) = (\mathbf{X}_1(t)^T, \mathbf{X}_2(t)^T, ...\mathbf{X}_N^T(t))^T$, $\mathbf{X}_i \in \mathscr{G}_i$. If we introduce

$$\mathbf{b}(\mathbf{X}, u_B) = (\mathbf{b}(\mathbf{X}_1, u_B)^T, \dots \mathbf{b}(\mathbf{X}_N, u_B)^T)^T \tag{7.18}$$

then the dynamics of $\mathbf{X}$ is

$$d\mathbf{X} = \mathbf{b}(\mathbf{X}, u_B)dt + \sum_{i=1}^{N} (\mathbf{e}_i \otimes (0, 0, \sigma_R)^T) dw_i \tag{7.19}$$

where $\mathbf{e}_i = (0, 0, ...1, ...0)^T$ is the N-dimensional standard basis vector with zeros everywhere, except 1 in the ith component. The symbol $\otimes$ represents the Kronecker matrix product, and the result of the operation in the brackets between the two-column vectors of the dimensions $N \times 1$ and 3×1, respectively, is the column vector of the dimension $3N \times 1$. The generator corresponding to the dynamics (7.19) and control policy u_B is given by the differential operator

$$\mathscr{L}^{u_B} f(\mathbf{X}) = \sum_{i=1}^{N} \mathscr{L}_i^{u_B} f(\mathbf{X}) \tag{7.20}$$

where $f : \mathscr{G} \to \mathbb{R}$ and $f \in C^2$ in its domain $\mathscr{G} \subset \mathbb{R}^{3N}$.

Using the relative dynamics (7.19), we proceed to define a stochastic optimal control problem as in [57], [94], for any admissible, pure Markov control policy u_B and any initial condition $\mathbf{X} \in \mathscr{G}$. The time at which B enters for the first time any avoidance $\mathscr{A}_i$ or target set $\mathscr{T}_i$, $i = 1, ...N$, i.e., when it reaches the boundary $\partial\mathscr{G}$ of $\mathscr{G}$, is $\tau = min(\tau_1, ..., \tau_N) = \inf\{t : \mathbf{X}(t) \in \partial\mathscr{G}\}$. Therefore, our goal is to define the control policy $u_B(\mathbf{X}) : \mathscr{G} \to [-u_{max}, u_{max}]$ that minimizes the cost function

$$W(\mathbf{X}, u_B) = \mathbb{E}_{\mathbf{X}}^{u_B} \left\{ g(\mathbf{X}(\tau)) + \int_0^\tau dt \right\} \tag{7.21}$$

where $\mathbb{E}_{\mathbf{X}}^{u_B}\{\cdot\}$ is the expectation operator with respect to the probability distribution of the realization of the trajectory $\mathbf{X}(t)$ which starts at $\mathbf{X}(0) = \mathbf{X} \in \text{int}(\mathscr{G})$ and terminates in $\mathbf{X}(\tau) \in \partial\mathscr{G}$ under the control u_B, and the terminal cost

$$g(\mathbf{X}) = \begin{cases} M, & \text{if } \mathbf{X} \text{ is such that any of } \mathbf{X}_i \in \mathscr{A}_i \\ 0, & \text{otherwise} \end{cases} \tag{7.22}$$

where $M \gg 0$ is a large constant which applies a penalty for entering the boundary $\partial\mathscr{G}$ with any of $\mathbf{X}_i$ in the avoidance set $\mathscr{A}_i$, i.e., outside the target set $\mathscr{T}_i$. The optimal control policy u_B, which minimizes (7.21) for any initial condition $\mathbf{X} \in \text{int}(\mathscr{G})$, results in the optimal cost-to-go function $V(\mathbf{X}) = \inf_{u_B} W(\mathbf{X}, u_B)$, which is the solution of the dynamic programming (HJB) equation

$$\begin{cases} \inf_{u_B}\{\mathscr{L}^{u_B} V(\mathbf{X}) + 1 = 0\}, & \mathbf{X} \in \text{int}(\mathscr{G}) \\ V(\mathbf{X}) = g(\mathbf{X}), & \mathbf{X} \in \partial\mathscr{G} \end{cases} \tag{7.23}$$

While the HJB solution defines theoretically the optimal control u_B, finding its solution is practically difficult and we need to resort to approximate computational methods [94]. The approach we propose here is inspired by the form of (7.20), whose structure is a sum of $\mathscr{L}_i^{u_B}$. Therefore, we propose to solve a one-on-one problem involving B and a single red agent ($N = 1$), and use the solution to approximate the solution of (7.23) for multiple red vehicles ($N > 1$).

7.3 Intercept of a Single Vehicle with Avoidance of Unsafe Configurations (one-on-one solution)

In this section $N = 1$; therefore, without loosing generality, we set $i = 1$. To numerically solve the single-target stochastic tail chase problem, we utilize the locally consistent Markov chain approximation method [94]. In the first part of this section, we briefly describe this method.

7.3.1 Locally Consistent Markov Chain Approximation Method

We discretize the continuous domain $\mathscr{G}_i \subset \mathscr{R}^3$ with small discrete steps Δr, $\Delta\phi$, $\Delta\alpha$ for each component of $\mathbf{X}_i \in \mathscr{G}_i$. With this we obtain a discrete state space, denoted by $\mathscr{G}_i^h$, which defines the states of the discrete time Markov chain approximation $\mathbf{X}_i^h(n)$, where n is the index of discrete steps. The locally consistent Markov chain approximation $\mathbf{X}_i^h(n)$ of $\mathbf{X}_i(t)$ requires that increments $\Delta\mathbf{X}_i^h(n) = \mathbf{X}_i^h(n+1) - \mathbf{X}_i^h(n)$ satisfy

$$\mathbb{E}\left\{\Delta\mathbf{X}_i^h(n)\right\} = \mathbf{b}(\mathbf{X}_i^h, u_B)\Delta t^h + o(\Delta t^h) \tag{7.24}$$

$$Cov\left\{\Delta\mathbf{X}_i^h(n)\right\} = \mathbf{a}(\mathbf{X}_i^h)\Delta t^h + o(\Delta t^h) \tag{7.25}$$

where the discrete steps are separated by interpolation times Δt^h, $\mathbb{E}\{\cdot\}$ represents the conditional expectation given the discretization steps, control action u_B and state $\mathbf{X}_i^h$, $Cov\{\mathbf{X}\} = \mathbb{E}\{\mathbf{X}\mathbf{X}^T\}$, and the matrix $\mathbf{a}(\mathbf{X}_i^h) = diag(0,0,\sigma_R^2)$.

By defining the transition probabilities, $p^h(\mathbf{X}_i^h(n+1)|\mathbf{X}_i^h(n), u_B, \Delta t^h)$, in such a way

that the discrete chain $\mathbf{X}_i^h(n)$ is locally consistent [13, Sec. 4.1] with the original process $\mathbf{X}_i(t)$, it can be shown that the optimal cost and control of the discrete problem, which satisfy the discrete dynamic programming equation,

$$V_i^h(\mathbf{X}_i^h) = \min_{u_B \in \mathscr{U}} \left\{ \sum_{\mathbf{Y}_i^h \in \mathscr{N}(\mathbf{X}_i^h)} p^h(\mathbf{Y}_i^h | \mathbf{X}_i^h, u_B) V_i^h(\mathbf{Y}_i^h) + \Delta t^h \right\} \tag{7.26}$$

with boundary condition

$$V_i(\mathbf{X}_i^h) = g(\mathbf{X}_i^h), \quad \mathbf{X}_i^h \in \partial \mathscr{G}_i^h \tag{7.27}$$

converge, as $\Delta r, \Delta\phi, \Delta\alpha \to 0$, with the known rate of convergence [164] to the optimal cost and control of the continuous optimal control problem, given from the solution of (7.23).

In expression (7.26), $\mathscr{N}^h(\mathbf{X}_i^h)$ denotes the set of six neighbor states $\mathscr{N}^h(\mathbf{X}_i^h) = \{(\mathbf{X}_i^h \pm (\Delta r, 0, 0)^T), (\mathbf{X}_i^h \pm (0, \Delta\phi, 0))^T, (\mathbf{X}_i^h \pm (0, 0, \Delta\alpha))^T\}$ which are the only possible transition states. To abbreviate notation, we write the two transition probabilities along the r_i component as $p_r^{\pm}(\mathbf{X}_i, u_B) = p^h(\mathbf{X}_i^h \pm (\Delta r, 0, 0)^T | \mathbf{X}_i^h, u_B)$ and we do it similarly for $p_\phi^{\pm}(\mathbf{X}_i^h, u_B)$ and $p_\alpha^{\pm}(\mathbf{X}_i^h, u_B)$. Based on the locally consistent Markov chain approximation, the transition probabilities are

$$p_r^{\pm}(\mathbf{X}_i^h, u_B) = t^h(\mathbf{X}_i, u_B) b_r^{\pm}(\mathbf{X}_i) / \Delta r \tag{7.28}$$

$$p_\phi^{\pm}(\mathbf{X}_i^h, u_B) = t^h(\mathbf{X}_i, u_B) b_\phi^{\pm}(\mathbf{X}_i, u_B) / \Delta\phi \tag{7.29}$$

$$p_\alpha^{\pm}(\mathbf{X}_i^h, u_B) = t^h(\mathbf{X}_i, u_B) b_\alpha^{\pm}(u_B) / \Delta\alpha + \sigma_R^2 / (\Delta\alpha)^2 \tag{7.30}$$

where $b_r^{\pm}(\mathbf{X}_i) = \max\{0, \pm b_r(\mathbf{X}_i)\}$ is used with the '+' sign for the step Δr and with the '$-$' sign for the step $-\Delta r$. The values $b_\phi^{\pm}(\mathbf{X}_i, u_B)$ and $b_\alpha^{\pm}(u_B)$ are defined in the same way. The transition probabilities are based on the state and control dependent interpolation time

$$\Delta t^h(\mathbf{X}_i, u_B) = \left(\frac{|b_{r_i}(\mathbf{X}_i)|}{\Delta r} + \frac{|b_{\phi_i}(\mathbf{X}_i, u_B)|}{\Delta \phi} + \frac{|b_{\alpha_i}(u_B)|}{\Delta \alpha} + \frac{\sigma_R^2}{(\Delta \alpha)^2} \right)^{-1} \tag{7.31}$$

Note that the discretization scheme is based on fixed steps Δr, $\Delta \phi$, $\Delta \alpha$, while the interpolation interval is defined by the problem parameters; therefore, this type of discretization is called *time implicit discretization* [94].

Using the discrete approximation (7.26)-(7.27) and (7.28)-(7.31), and the so-called value iterations, we can obtain an approximate solution of (7.23) for a single target. For brevity, we will denote this numerical method as

$$(u_B^h, V_i^h) \leftarrow HJBSolution(v_B, v_R, \sigma_R, \mathscr{T}_1, \mathscr{A}_i, M) \tag{7.32}$$

where the superscript h indicates that the results of the computation are in the form of a lookup table corresponding to the discrete computational domain and the problem input parameters are listed in the brackets.

The expected time to reach the target, $\mathscr{T}_i$, can be computed [61] from the backwards Kolmogorov equation,

$$\begin{cases} \mathscr{L}^{u_B} T_{B_i}(\mathbf{X}_i) + 1 = 0\}, & \mathbf{X}_i \in \text{int}(\mathscr{G}_i) \\ T_{B_i}(\mathbf{X}_i) = 0, & \mathbf{X}_i \in \mathscr{T}_i \end{cases} \tag{7.33}$$

Comparing the above with (7.23), we see that once the optimal u_B is fixed (is computed), the cost-to-go function $V_i(\mathbf{X}_i)$ and expected time $T_{B_i}(\mathbf{X}_i)$ solve the same PDE with different boundary conditions. Therefore, once we numerically solve (7.26) to find the optimal cost and policy, we can apply the same discretization and compute the expected time as

$$T^h_{B_i}(\mathbf{X}^h_i) = \min_{u_B \in \mathscr{U}} \left\{ \sum_{\mathbf{Y}^h_i \in \mathscr{N}(X^h_i)} p^h(\mathbf{Y}^h_i | \mathbf{X}^h_i, u_B) T^h_{B_i}(\mathbf{Y}^h_i) + \Delta t^h \right\} \tag{7.34}$$

$$T_{B_i}(\mathbf{X}^h_i) = 0, \quad \mathbf{X}^h_i \in \partial \mathscr{T}^h_i \tag{7.35}$$

We denote the numerical method of computing the expected time as

$$T^h_{B_i} \leftarrow BKGSolution(v_B, v_R, \sigma_R, \mathscr{T}_i, \mathscr{A}_i, u_B) \tag{7.36}$$

7.3.2 Avoidance of Unsafe Configurations

Under our problem formulation, the red agent *may*, or *may not* have the goal to enter the tail of B, which is defined in the same way as the tail of the red agent. The threat from the red agent entering the tail of B should be anticipated by the navigation of B. We account for this threat by assuming that if R_i has the goal to enter the tail of B, then it wants to achieve it without colliding with B and in the shortest possible time. Since R_i does not know the navigation strategy of B, the problem of defining its navigation is similar to the one posed in Section 7.3.1, with the roles of R_i and B reversed. In this case, the relative coordinates of B relative to R_i, (r^R, ϕ^R, α^R), are the distance, the

bearing angle and alignment angle, respectively, defined from the perspective of R. At any time point, their relation to $(r^B_{R_i}, \phi^B_{R_i}, \alpha^B_{R_i})$ is

$$r^R = r^B_{R_i};\ \ \phi^R = \pi + \phi^B_{R_i} - \alpha^B_{R_i};\ \ \alpha^R = -\alpha^B_{R_i}. \tag{7.37}$$

For the purpose of computing the control u^R, the dynamics of these relative coordinates have to account that u_R is the control variable, i.e., $d\theta_R = u_R$, and that control u_B is unknown, i.e., $d\theta_B = \sigma_B dw_B$. Based on the fact that these "reversed role" relative dynamics are similar to (7.10), we can compute the control u_R using the same technique described in Section 7.3. However, we set $\sigma_B = \sigma_R$ stating that the amount of uncertainty that B and R have about each other's control is the same. With the tail of B defined in the same way as the tail of R, the optimal control policy u_R can be computed as

$$(u^h_R, V^h_i) \leftarrow HJBSolution(v_R, v_B, \sigma_R, \mathscr{T}_i, \mathscr{C}_i, M) \tag{7.38}$$

which is different from (7.32) only because of the switched order of v_R and v_B. We can also evaluate the expected time to reach the target

$$T^h_R \leftarrow BKGSolution(v_R, v_B, \sigma_R, \mathscr{T}_i, \mathscr{C}_i, u_R) \tag{7.39}$$

which is the information we need to compute the avoidance set $\mathscr{A}_i$.

The iterative procedure for computing B's avoidance set, which takes into account the threat of hostile red agents, is provided in Figure 7.3. The first line computes the initial optimal control u_B taking into account only the collision set $\mathscr{C}_i$; therefore, initially

the avoidance set $\mathscr{A}_i = \mathscr{C}_i$. The expressions (7.38) and (7.39) are included as the second and the third lines of the algorithm. Inside the *repeat* loop we evaluate the expected time of reaching the tail of R, and any point with $T_R^h < T_{B_i}^h$ is included in the unsafe set $\bar{\mathscr{S}}_i$, which is then included in the avoidance set $\mathscr{A}_i^k$. The avoidance set is updated in each iteration and, after each update, the optimal control is re-computed. The iterations stop once the set of unsafe configurations is empty and the last updated avoidance set is returned as the result.

Let us denote with $Card(\cdot)$ the number of elements in a set. Therefore, the iterations stop when $Card(\bar{\mathscr{S}}_i) = 0$ and through the iterations $Card(\mathscr{A}_i^k) = Card(\mathscr{A}_i^{k-1}) + Card(\bar{\mathscr{S}}_i)$, which implies

$$0 \leq Card(\mathscr{A}_i^{k-1}) \leq Card(\mathscr{A}_i^k) \leq Card(\mathscr{G}_i) \tag{7.40}$$

The sequence $Card(\mathscr{A}_i^k)$ is monotically increasing and limited from above. Therefore, only two outcomes are possible. First, the iterations stop for $\mathscr{A}_i^k$ corresponding to the whole space, in which case, we conclude that the control u_B does not exist since B has to avoid the whole space, i.e., $\mathscr{G}_i$. Second, $\mathscr{A}_i^k \subseteq Card(\mathscr{G}_i)$.

In the second case, the control can be computed using (7.32) and the corresponding expected time can be evaluated by (7.34).

Algorithm AvoidanceSet $(v_B, v_R, \sigma_R, \mathscr{T}_i, \mathscr{C}_i)$

$(u_B^0, V_i) \leftarrow HJBSolution(v_B, v_R, \sigma_R, \mathscr{T}_i, \mathscr{C}_i, M)$
$(u_R, V_i) \leftarrow HJBSolution(v_R, v_B, \sigma_R, \mathscr{T}_i, \mathscr{C}_i, M)$
$T_R \leftarrow BKGSolution(v_R, v_B, \sigma_R, \mathscr{T}_i, u_R)$
$\mathscr{A}_i^0 = \mathscr{C}_i$, $k = 0$
repeat:
  $k = k+1$, $\bar{\mathscr{S}}_i = \emptyset$
  $T_B^{k-1} \leftarrow BKGSolution(v_B, v_R, \sigma_R, \mathscr{T}_i, u_B^{k-1})$
  for all: (r^h, α^h, ϕ^h)
    $\alpha_R^h = -\alpha^h$
    $\phi_R^h = \pi + \phi^h - \alpha^h$
    if $\overline{T}_R(r^h, \alpha_R^h, \phi_R^h) < \overline{T}_B^{k-1}(r^h, \alpha^h, \phi^h)$ *then*
      $\bar{\mathscr{S}}_i = \bar{\mathscr{S}}_i \cup (r^h, \alpha^h, \phi^h)$
    end if
  end for
  $\mathscr{A}_i^k = \mathscr{A}_i^{k-1} \cup \bar{\mathscr{S}}_i$
  $(u_B^k, V_1) \leftarrow HJBSolution(v_B, v_R, \sigma_R, \mathscr{T}_1, \mathscr{A}_i^k, M)$
until $\bar{\mathscr{S}}_i == \emptyset$
$\mathscr{A}_i = \mathscr{A}_i^k$
return $\mathscr{A}_i$

Figure 7.3: Pseudocode for computing the avoidance set $\mathscr{A}_i$, which is a union of the collision $\mathscr{C}_i$ and unsafe configuration $\bar{\mathscr{S}}_i$ sets. The optimal control u_B^k and the avoidance set $\mathscr{A}_i^k$ are updated in each iteration until the set of unsafe configurations $\bar{\mathscr{S}}_i$ is empty.

7.3.3 Hazard and Expected Time

Let us denote the optimal control defined by (7.23) as $u^*_{B_i}$. Therefore, when $u_B = u^*_{B_i}$, the two partial differential equations (7.23) and (7.33) are the same, except for their difference in the boundary conditions. Since the differential operator $\mathscr{L}_i^{u_B}$ is linear, we can define the hazard $H(\mathbf{X}_i)$ as

$$V(\mathbf{X}_i) = H(\mathbf{X}_i) + T_{B_i}(\mathbf{X}_i) \tag{7.41}$$

where both $V(\mathbf{X}_i)$ and $T_{B_i}(\mathbf{X}_i)$ are the solutions of (7.23) and (7.33), respectively, corresponding to the optimal control, i.e., $u_{B_i} = u^*_{B_i}$. If we substitute $V(\mathbf{X}_i)$ from (7.41) into (7.23), we obtain that $H(\mathbf{X}_i)$ satisfies the same partial differential equation as (7.23), i.e., (7.33) with the boundary condition $H(\mathbf{X}_i) = M$ for $\mathbf{X}_i \in \partial\mathscr{G}_i$. The state X_i dependent probability $\mathbb{P}^{u_{B_i}}(\mathbf{X}_i)$ of reaching the boundary under the feedback control u_{B_i} can be computed as $\mathbb{P}^{u_{B_i}}(\mathbf{X}_i) = \frac{1}{M}H(\mathbf{X}_i)$. This can be verified by the fact that $\mathbb{P}^{u_{B_i}}$ also satisfies (7.33) with the boundary condition $\mathbb{P}^{u_B}(\mathbf{X}_i) = 1$ for $\mathbf{X}_i \in \partial\mathscr{G}_i$. This short analysis also indicates that the effect of a large M is to reduce the probability of reaching the boundary $\mathscr{G}_i$.

7.4 Scalable Navigation Strategy

Theoretically, one can apply the same discretized value iteration scheme used in the previous section to the optimal control with multiple red vehicles ($N > 1$). However,

the dimension of the state space $\mathscr{G} \subset \mathbb{R}^{3N}$ is large in the sense that if we discretize each dimension with D_s discrete steps, the number of grid cells over which we have to compute the value iterations is D_s^{3N}. Furthermore, the control computed that way would not be able to address the change in the number of red vehicles. To circumvent this challenge in this section, we propose a control based on the one-on-one solution (see Section 7.3.2) which requires the value iteration computations with only D_s^3 grid cells. To achieve that, we introduce an auxiliary Markov decision problem which is locally consistent with the original problem and derive the form of its value function. Then we use its one-step look-ahead approximation which approximates its value function via lookup tables for the expected time and the hazard of the one-on-one solution. These lookup tables have to be computed only once and stored in the memory, and allow the computation of a scalable navigation strategy.

7.4.1 Auxiliary Markov Decision Problem

Instead of dealing with the discretized state space $\mathscr{G}^h$ of $\mathscr{G} \subset \mathbb{R}^{3N}$ and the corresponding locally consistent Markov chain $\mathbf{X}^h(n)$ we could obtain by the discretization from (7.23), here we deal with the discrete state space $\tilde{\mathscr{G}}^h = \prod_{i=1}^{N} \mathscr{G}_i^h$, where $i = 1, ...N$. The state space is illustrated in Fig. 7.4 in which each plane represents $\mathscr{G}_i^h$. Let us denote with $p^h(\tilde{\mathbf{X}}^h(n+1)|\tilde{\mathbf{X}}^h(n), u_B)$ the transition probability of the Markov chain $\tilde{\mathbf{X}}(n)$ over the discretized space $\tilde{\mathscr{G}}^h$. Since our model of R_i for the control B design is

(7.4)-(7.6), i.e., the motion of the red agents R_i is independent, we conclude that

$$p^h(\tilde{\mathbf{X}}^h(n+1)|\tilde{\mathbf{X}}^h(n),u_B)=\prod_{i=1}^{N}p^h(\mathbf{X}_i^h(n+1)|\mathbf{X}_i^h(n),u_B) \tag{7.42}$$

where $\tilde{\mathbf{X}}(n)=(\mathbf{X}_1^h(n),...\mathbf{X}_N^h(n))^T$ with component evolutions resulting from the discretization scheme based on (7.26) and applied for each $i=1,...N$. Note that every state of a Markov chain $\mathbf{X}^h(n)$ resulting from (7.23) would have $6N$ adjacent states, 2 along each of $3N$ dimensions of the state space $\mathscr{G}^h$. However, in our case, each state of the auxiliary Markov chain $\tilde{\mathbf{X}}(n)$ has 6^N, 2 along each of 3 relative dimensions in each subspace $\mathscr{G}_i^h$. The reason for constructing the auxiliary chain is that, given a control, the transitions in each subspace are independent of each other. Now, one may question whether the auxiliary Markov chain $\tilde{\mathbf{X}}(n)$ is locally consistent with the original continuous process $\mathbf{X}(t)$. The following theorem resolves this issue.

Theorem 1. *For* $\widetilde{\mathbf{X}}^h=(\mathbf{X}_1^h,\mathbf{X}_2^h,\dots\mathbf{X}_N^h)$ *and* u_B*, define the multi-target interpolation interval* $\widetilde{\Delta t}^h$ *as*

$$\widetilde{\Delta t}^h(\widetilde{\mathbf{X}},u_B)=\min_{i=1,\dots,N}\Delta t^h(\mathbf{X}_i^h,u_B) \tag{7.43}$$

where $\Delta t^h(\mathbf{X}_i^h,u_B)$ *is defined by* (7.31)*. Then* $\widetilde{\mathbf{X}}^h(n)$ *is locally consistent with the process* $\mathbf{X}(t)$ *in* (7.19) *so that*

$$\begin{aligned}\mathbb{E}\left\{\Delta\widetilde{\mathbf{X}}^h(n)\right\}&=\mathbf{b}(\widetilde{\mathbf{X}}^h(n),u_B)\widetilde{\Delta t}^h+o(\widetilde{\Delta t}^h)\\ \mathbb{C}\mathrm{ov}\left\{\Delta\widetilde{\mathbf{X}}^h(n),\Delta\widetilde{\mathbf{X}}^h(n)\right\}&=\widetilde{\mathbf{a}}\widetilde{\Delta t}^h+o(\widetilde{\Delta t}^h)\end{aligned} \tag{7.44}$$

where $\Delta\widetilde{\mathbf{X}}(n)=\widetilde{\mathbf{X}}(n)-\widetilde{\mathbf{X}}(n-1)$*,* $\mathbf{b}$ *is defined by* (7.18)*,* $\widetilde{\mathbf{a}}=diag(\mathbf{1}_N\otimes(0,0,\sigma_R)^T$ *and* $\mathbf{1}_N$ *is the column vector of 1.*

Proof. Let $\hat{1}_i \in \mathbb{R}^{3N\times 3}$ be the block-stacked matrix with the 3×3 identity matrix in block i and zeros elsewhere. Then $\mathbb{E}\left\{\Delta\widetilde{\mathbf{X}}^h(n)\right\} = \mathbb{E}\left\{\sum_{i=1}^N \hat{1}_j \Delta\mathbf{X}_i^h(n)\right\} = \sum_{i=1}^N \hat{1}_i \mathbb{E}\left\{\Delta\mathbf{X}_i^h(n)\right\}$ since, by construction, the subchain $\mathbf{X}_i^h(n)$ is independent of $\mathbf{X}_j^h(n)$ for $i \neq j$.

Based on Lemma X from Sec. 5.3 of [94], we have that $\mathbb{E}\{\Delta\mathbf{X}_i(n)\} = \mathbf{b}(\mathbf{X}_i,u)$ $\Delta t^h(\mathbf{X}_i,u)$ where $\Delta t^h(\mathbf{X}_i,u)$ is as in (7.31). Without losing generality, we will as in [94] assume that discretization steps are expressed based on a mesh size h, e.g., Δr, $\Delta\phi$ and $\Delta\alpha$ are expressed as $m_k h$, $k = 1,2,3$, respectively. With mesh-size h, and $\mathbf{b}_{\{1,2,3\}}(\mathbf{X}_j,u_B)$ which corresponds to $\mathbf{b}_{\{r_j,\phi_j,\alpha_j\}}(\mathbf{X}_j,u_B)$ we have $Q^h(\mathbf{X}_i,u_B) = \sigma_R^2 + h\sum_{k=1}^3 m_k|b_k(\mathbf{X}_j,u_B)|$, where therefore, $\Delta t^h(\mathbf{X}_i,u_B) = m_3^2h^2/Q^h(\mathbf{X}_i,u_B)$ is

$$\Delta t^h(\mathbf{X}_i,u_B) = \frac{m_3^2h^2/\sigma_R^2}{1+\frac{h}{\sigma_R^2}\sum_{k=1}^3 m_k|b_k(\mathbf{X}_i,u_B)|}$$

and

$$\widetilde{\Delta t}^h(\widetilde{\mathbf{X}},u_B) = \frac{m_3^2h^2/\sigma_R^2}{1+\frac{h}{\sigma_R^2}\max_j\{\sum_{k=1}^3 m_k|b_k(\mathbf{X}_j,u_B)|\}}$$

We can now exploit a general inequality $|\frac{1}{1+x}-\frac{1}{1+y}| \leq |x-y|$, $x,y>0$ and by the boundedness of $\mathbf{b}(\cdot)$ on the domain $\mathscr{G}_i^h$ obtain

$$|\Delta t^h(\mathbf{X}_i,u_B) - \widetilde{\Delta t}^h(\widetilde{\mathbf{X}},u_B)| \leq K\frac{m_3^2h^3}{\sigma_R^4} \tag{7.45}$$

with
$$K \geq |\sum_{k=1}^3 m_k|b_k(\mathbf{X}_i,u_B)| - \max_j\{\sum_{k=1}^3 m_k|b_k(\mathbf{X}_j,u_B)|\}| \tag{7.46}$$

It follows that $|\Delta t^h(\mathbf{X}_i,u_B) - \widetilde{\Delta t}^h(\widetilde{\mathbf{X}},u_B)|/\widetilde{\Delta t}^h(\widetilde{\mathbf{X}},u_B) = O(h)$, so the difference (7.45) is $o(\widetilde{\Delta t}^h(\widetilde{\mathbf{X}},u_B))$ for all i.

Putting the above together, we have

$$\mathbb{E}\{\Delta\widetilde{\mathbf{X}}^h(n)\} = \sum_{i=1}^{N} \hat{1}_j \mathbf{b}(\mathbf{X}_j, u_B)\Delta t^h(\mathbf{X}_j, u_B) = \sum_{i=1}^{N} \hat{1}_j \Big\{$$
$$\mathbf{b}(\mathbf{X}_j, u_B)\left[\widetilde{\Delta t}^h(\widetilde{\mathbf{X}}, u_B) + \left(\Delta t^h(\mathbf{X}_j, u_B) - \widetilde{\Delta t}^h(\widetilde{\mathbf{X}}, u_B)\right)\right]\Big\}$$
$$= \mathbf{b}(\widetilde{\mathbf{X}}, u_B)\widetilde{\Delta t}^h(\widetilde{\mathbf{X}}, u_B) + o\left(\widetilde{\Delta t}^h(\widetilde{\mathbf{X}}, u_B)\right).$$

The covariance follows in a similar manner. Writing $\overline{\Delta\widetilde{\mathbf{X}}^h}(n) = \Delta\widetilde{\mathbf{X}}^h(n) - \mathbb{E}\{\Delta\widetilde{\mathbf{X}}^h(n)\}$ for the (zero-mean) centered increment, the covariance from the theorem statement is

$$\mathbb{E}\{\overline{\Delta\widetilde{\mathbf{X}}^h}(n)\overline{\Delta\widetilde{\mathbf{X}}^h}(n)^T\} = \mathbb{E}\left\{\sum_{i=1}^{N}\sum_{j=1}^{N} \hat{1}_i[\overline{\Delta\widetilde{\mathbf{X}}}_i][\overline{\Delta\widetilde{\mathbf{X}}}_j]^T \hat{1}_j^T\right\}$$
$$= \sum_{j=1}^{N} \hat{1}_j\left[\mathbf{a}(X_j)\Delta t^h(\mathbf{X}_j, u_B) + o(\Delta t^h(\mathbf{X}_j, u_B))\right]\hat{1}_j'$$
$$= \widetilde{\mathbf{a}}(\widetilde{\mathbf{X}})\widetilde{\Delta t}^h(\widetilde{\mathbf{X}}, u_B) + o\left(\widetilde{\Delta t}^h(\widetilde{\mathbf{X}}, u_B)\right).$$

□

From Theorem 1, one can apply the results of [94] to show the convergence of the auxiliary MDP to the original continuous-time problem associated with (7.18)-(7.21). Consequently, the original problem can be solved in the discrete space $\widetilde{\mathscr{G}}^h$ using

$$\widetilde{V}^h(\widetilde{\mathbf{X}}^h) = \min_{u_B \in \mathscr{U}} \left\{ \sum_{\widetilde{\mathbf{Y}}^h \in \mathscr{N}(\widetilde{\mathbf{X}}^h)} \widetilde{p}^h(\widetilde{\mathbf{Y}}^h; \widetilde{\mathbf{X}}^h, u_B)\widetilde{V}^h(\widetilde{\mathbf{Y}}^h) + \widetilde{\Delta t}^h \right\} \tag{7.47}$$

for $\widetilde{\mathbf{X}} \in \widetilde{\mathscr{G}}^h$ and $\widetilde{V}^h(\widetilde{\mathbf{X}}) = g(\widetilde{\mathbf{X}})$ on $\partial\widetilde{\mathscr{G}}^h$.

7.4.2 One-Step Look-Ahead Cost Approximation

While the transition probabilities (7.42) define a locally consistent chain, solving the discrete HJB (7.47) for $\widetilde{V}^h(\widetilde{\mathbf{X}})$ (and the corresponding optimal control u_B via value iteration still explodes in computation time as the number of targets increases.

Instead, we approximate these values via a one-step look-ahead cost approximation

$$\widehat{V} = \min_{u_B} \left\{ \min_i \{ \mathscr{T}_{B_i}(\mathbf{X}_i, u_B) \} + \sum_{j=1}^{N} \mathscr{H}(\mathbf{X}_j, u_B) \right\} \tag{7.48}$$

composed of the total expected risk with respect to all red vehicles

$$\mathscr{H}(\mathbf{X}_i, u_B) = \sum_{\mathbf{Y}^h \in \mathscr{N}(\mathbf{X}_i)} p^h(\mathbf{Y}^h | \mathbf{X}_i, u_B) H(\mathbf{Y}^h, u_B) \tag{7.49}$$

and the minimum expected time to the target i

$$\mathscr{T}_{B_i}(\mathbf{X}_i, u_B) = \Delta t_i^h + \sum_{\mathbf{Y}^h \in \mathscr{N}(\mathbf{X}_i)} p^h(\mathbf{Y}^h | \mathbf{X}_i, u_B) T_{B_i}(\mathbf{Y}^h, u_B)$$

where $\Delta t_i^h = \Delta t^h(\mathbf{X}_i, u_B)$ and $\mathscr{N}(\mathbf{X}_i)$ are the 6^N neighbor states of $\mathbf{X}_i$. Based on this approximation, the control u_B minimizes the expression in the braces of (7.48). The approximation is scalable because all the values used in the approximations can be pre-computed and stored as lookup tables, and the number of computations linearly increases with the number of agents. Another important property of the approximation is that for $N = 1$, it is not an approximation, but an exact match to the optimal control solution. This method can be thought of as a form of approximate dynamic programming [138].

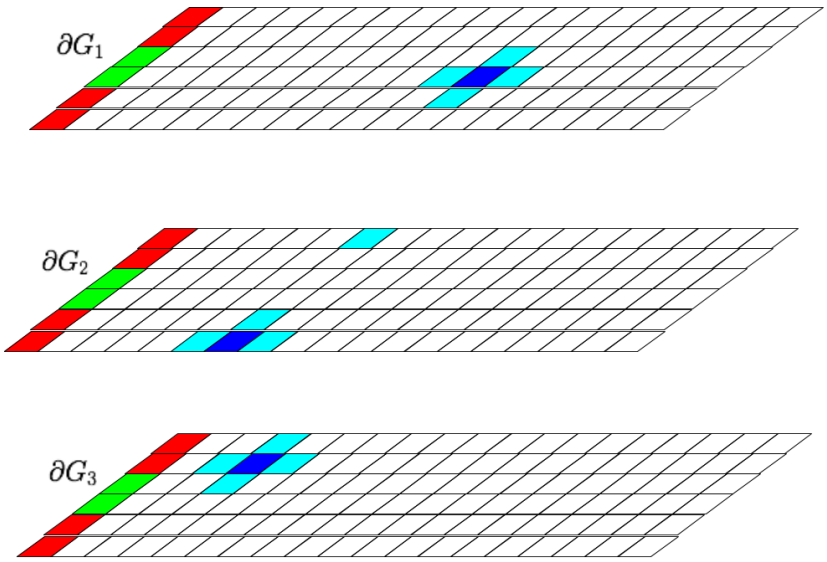


Figure 7.4: The discretized auxiliary Markov decision problem state space. Each plane represents a (simplified, $\phi \equiv 0$) domain for the relative dynamics with respect to each boundary $\partial \mathcal{G}_i^h$ with the target set states colored green and the states to be avoided colored red. The relative position between B and R_i is represented in the ith plane and colored blue. The neighbor states in which the relative position can be in the next step are colored light blue.

For multiple red vehicles, the approximation includes the sum of expected risks as $\mathscr{H}(\mathbf{X}_i, u_B)$. If the second part of the expression were the sum of the expected times, then our approximation could be written in terms of the value functions $V_i^h(\mathbf{X}_i, u_B)$, $i = 1, ...N$. However, the sum would include the expected times to the tails of all red agents and farther agents would contribute more to the sum. This is clearly unacceptable; at the state in which B is close to the tail sector of a specific red vehicle, the control should mainly depend on the expected time proximity to the tail sector and not on the expected times to the tails of distant vehicles. Consequently, in our approximation, we use the min operator for the expected times.

7.5 Results

The numerical simulation results of this section are based on the Dubins vehicle models (7.1)-(7.6) with the velocities $v_B = 0.1$ and $v_R = 0.05$ for B and R, respectively. The navigation uncertainties of R in (7.6) and of B in (7.38) are modeled with $\sigma_R = \sigma_B = \sqrt{10\pi/180}$ resulting in a standard deviation of the heading angle change of $10deg/sec$. The target set $\mathscr{T}_i$ in (7.14) and the collision set $\mathscr{C}_i$ in (7.15) are based on the parameters $\underline{r} = 0.05$, $\bar{r} = 0.15$, $\bar{\phi} = 10deg$ and $\bar{\alpha} = 20deg$. The discretization steps in our computations are $\Delta r = (\underline{r} - \bar{r})/100$, $\Delta\phi = \Delta\alpha = 5deg$. The M value, which is used in the iterative algorithm (see Fig. 7.4) for computing the avoidance set, is $M = 10^4$. Since the heading angle of B is the integral of u_B, i.e., $d\theta_B = u_B dt$, in our computations of optimal control for one-on-one solutions, we used the discrete values $u_{B_i} \in \{-0.5, 0, 0.5\}$, $i = 1,2,3,4$ to speed up the computations. In the computation of u_B based on our algorithm, we allowed for values between -0.5 and 0.5 with the step of 0.1.

To illustrate the avoidance set, we created a 2D map depicted in Fig. 7.5. The map coordinates are the x and y coordinates of B, relative to R, which is in the center of the map and pointed to the right as depicted in the figure together with the radius $\underline{r}$ and the tail sector in the B's target set. For every point (x,y) and for any heading angle of B at that point, i.e., the closest discrete point, we check if the value function $V_i^k = M$. If it is, then we know that the relative position is in the avoidance set and we mark that $x - y$

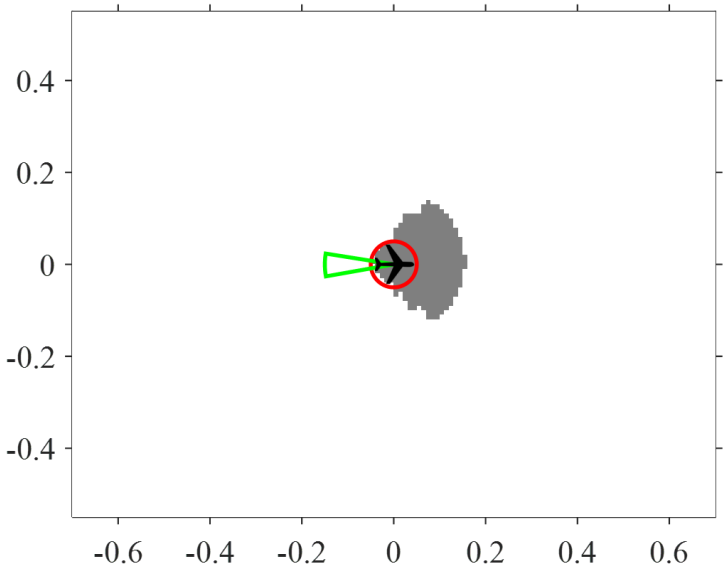


Figure 7.5: The avoidance set for $v_B = 0.1$ and $v_R = 0.05$ is depicted by gray points. The center of the map depicts the red agent, its tail sector (green) and circular collision set (red).

coordinate with gray color. Although the map is a projection of the avoidance set, it is likely that most of the positions to be avoided by B are in front of R and that the safest positions for B are behind R.

The avoidance set in Fig. 7.5 is computed after *two* iterations of the algorithm in Fig. 7.3. For the purpose of testing, we also computed the avoidance set for the velocity ratio between v_B and v_R that is closer to 1 ($v_B = 0.1$ and $v_R = 0.075$), and found that it required *four* iterations. That confirmed our expectation that the more competitive agents are, the more iterations it takes to compute the avoidance set. Once we reduced the discretization steps to $\Delta\phi = \Delta\alpha = 2deg$, the algorithm took *six* iterations until the convergence, which suggests that the size of the discrete space also impacts the number

of iterations.

In Fig. 7.6, we present an example of the simulation resulting from our navigation control. Figure 7.7 depicts the expected time toward each R agent and the corresponding hazard values. At the beginning, depicted in Fig. 7.6*a*, the B agent is in the position in which one of the red agents (R_4, gray) is the one towards which B has the smallest expected time, but it faces the other three red agents and the risk of collision. Therefore, the hazard values for R_1, R_2, R_3 are high in the time interval corresponding to the shaded segment A (see Fig. 7.7). To reduce the hazard values, the B agent maneuvers around the three R agents (see Fig. 7.6*b*), which results in the drop of the hazard values by the end of the time interval covered by the segment A. In the time interval between the shaded segments A and B, the expected time is reduced while the hazard to R_1 moderately increases (see Fig. 7.7). The position around which the hazard peaks is in Fig. 7.6*b*. Now B maneuvers to reduce the hazard, but at the cost of the steep increase of the expected times, which corresponds to the time covered by the shaded segment B. After the segment B, the expected time towards R_2 (green) is the smallest for the first time. It is the moment at which B starts cutting in front of R_1 and R_3. During the time interval covered by the shaded interval C, the expected time to R_2 is the shortest simultaneously with the highest hazard since B has already passed R_3 and is started cutting in front of R_2 as depicted in Fig. 7.6*c*. Shortly after the segment C time interval, the agent R_4 is out of range, the expected time for R_4 (gray) is too large to be plotted

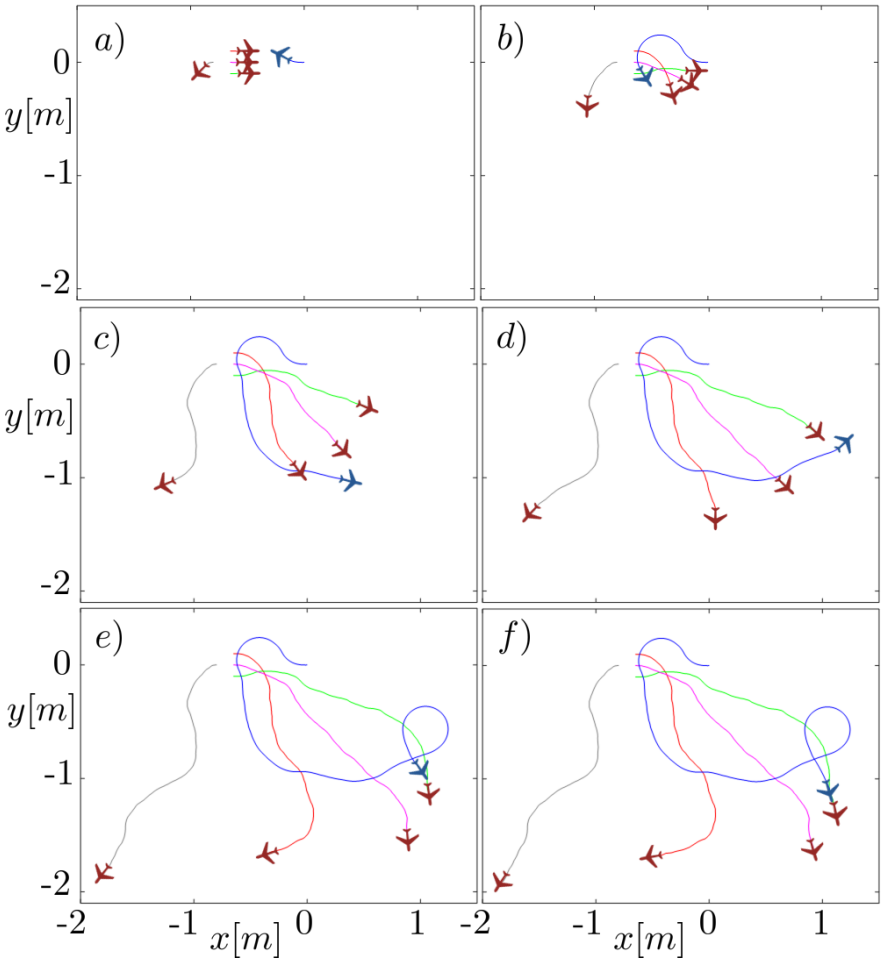


Figure 7.6: Simulation result. Each panel corresponds to the following simulation times: (a) 1.32s; (b) 9.32s; (c) 25.00s ; (d) 34.00s; (e) 45.29s; and (f) 47.73s, which is also the time at which B enters the tail sector of R_2. The B agent trajectory is colored blue and the R_1 to R_4 trajectories are colored red, green, magenta and gray, respectively (see Fig. C.1 in Appendix C).

and the corresponding hazard drops to 0. At that point, the shortest expected time does not correspond to R_2 and it remains like that until the time point towards the middle between the segments C and D, i.e., the position depicted Fig. 7.6*d*. In the time interval covered by the segment D, the expected time towards R_2 continues to decrease, but there is also a steep decrease in the corresponding hazard value (see Fig. 7.6*e*). At the time point of the intercept depicted in Fig. 7.6*f*, both the hazard, as well as the expected time are 0.

The optimal control of B towards each R, which results from the solution of the one-on-one problem, as well as the optimal control u_B used in the navigation in our scenario are plotted in Fig. 7.8. The values of the control are in the range of minimal and maximal values of the turning rate. As we can see from the diagram, there are time instances when the control u_B looks like as if it were obtained by the control that is optimal towards the majority of R agents. The best examples are time intervals covered by the shaded segments, E and H. In both of these segments $u_{B_1} = u_{B_2} = u_{B_3}$; therefore, independently of the value u_{B_4}, the value u_B is defined by the previous values. Note that in the segment H, all individual u_{B_i}, $i = 1,2,3,4$, are equal and it would be very strange to have u_B with a different value.

In the shaded segment F, we find $u_{B_1} = u_{B_2}$ and $u_{B_3} = u_{B_4}$. Since there is no majority of control values that are the same, the value u_B switches between these two values, or takes some value in between as may be expected by a *majority rule*. A clear example

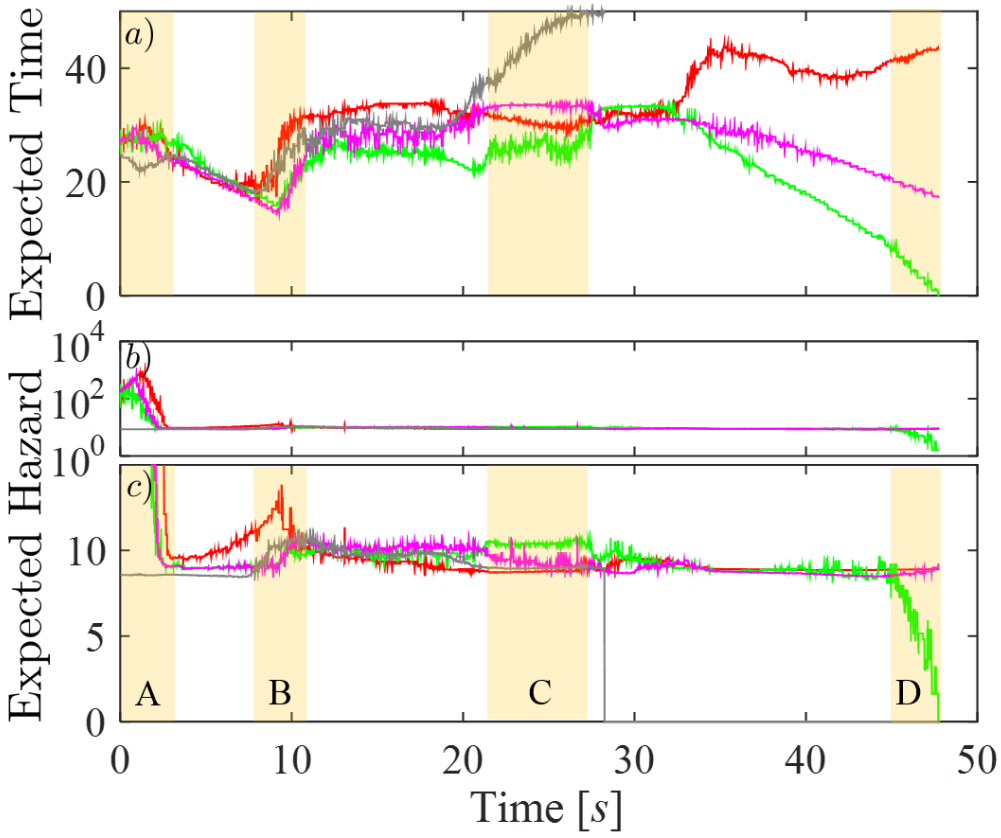


Figure 7.7: *a*) Expected time *b*) Hazard values on the log scale *c*) Hazard values on the linear scale. The diagrams corresponding to R_1-R_4 are colored red, green, magenta and gray, respectively.

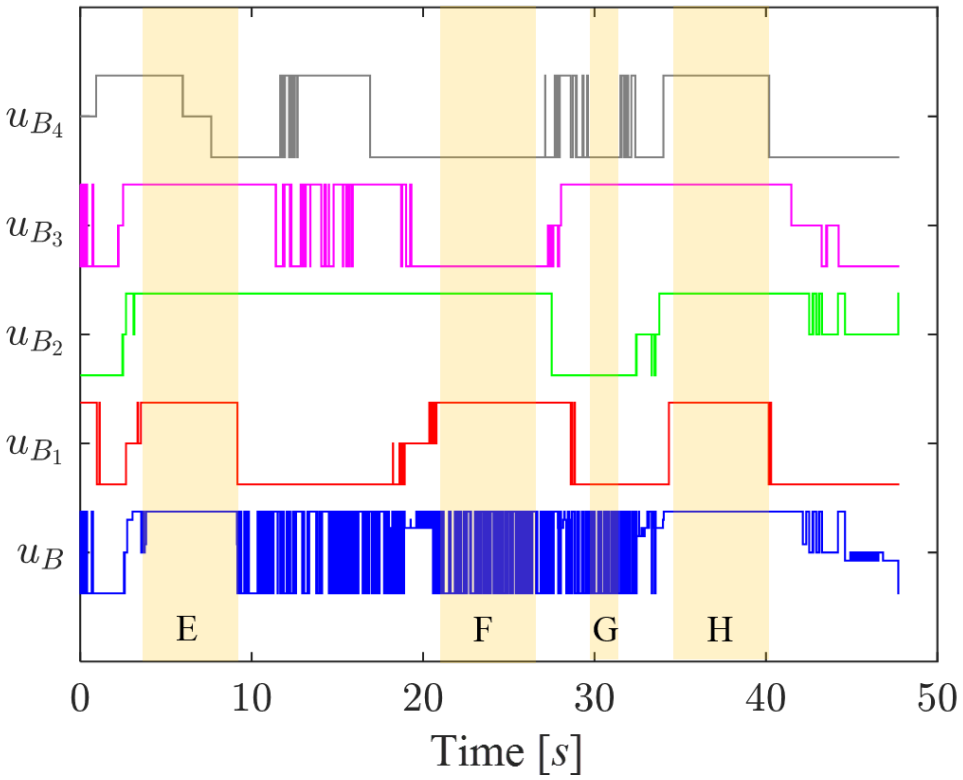


Figure 7.8: The control u_B and control variables u_{B_i}, $i = 1,2,3,4$ resulting from the solution of the one-on-one problem and relative positions between B and R_i at every time point of the simulation. All values are in the range $[-0.5, 0.5]$.

that our control does not yield such a rule is the control in the time interval covered by the shaded segment G. In the segment, $u_{B_1} = u_{B_2} = u_{B_4}$ and the control u_B still switches and takes some value in between the u_{B_1} and u_{B_3} values.

7.6 Robot Experiment Results

The experimental setup included five e-puck robots with infrared markers which were tracked by a motion capturing system composed of four Bonita 10 Vicon cameras. For each e-puck robot, we implemented proportional-integral (PI) controllers for the velocity and turning rate of the robot. These low-level control loops, software and hardware architecture supporting the experiment have been presented in [125].

Out of five e-puck robots, one had the role of B and four had the roles of R_1-R_4. The velocity of B was set to a constant $10cm/s$. The control for B used motion capturing system measurements to compute relative positions between B and each R_i, $i = 1,2,...4$. The relative positions were used to read lookup tables for the expected time, hazard and optimal control u_{B_i}, and compute the turning rate u_B. This turning rate was sent to the low-level PI controller that controlled the motion of B.

The four R_1-R_4 robots were controlled to follow trajectories that were similar to those from the simulation. However, in dealing with the experimental setup, we had to adjust the trajectories to avoid collisions among R_1-R_4 due to their size and fit them within the confined space of the experiment, see Fig. 7.9. As a result, the velocities of

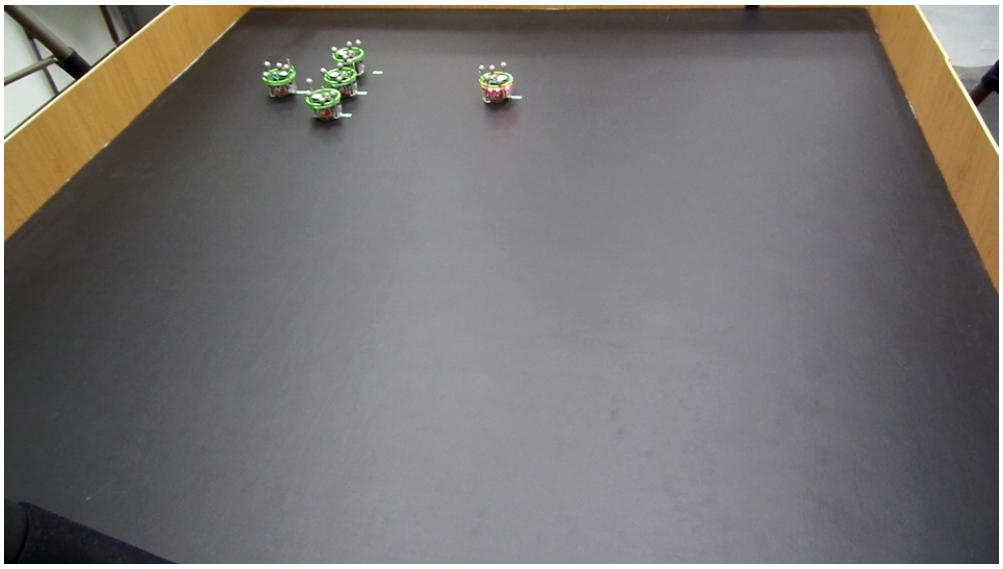


Figure 7.9: Robot experiment setup with five e-puck robots. Each robot has a unique configuration of infrared reflecting markers (green rings with silver spheres) tracked by a motion capturing system with four Bonita 10 Vicon cameras. The robots are at their initial position (see Fig. C.2 in Appendix C).

R_i, $i = 1,2,...4$, were set to $v_R = 3.25cm/s$. Although, this velocity was smaller then the one used in our stochastic control design ($v_R = 5m/s$), we did not update the design with this value. Our experience in the experiments [125] and flight tests [119] is that the type of stochastic optimal control we use does not require a perfect matching of parameters with the reality.

The initial configuration of B and R_i robots, $i = 1,2,...4$, in the experiment was similar to the one used in the simulation and encircled in Fig. 7.10. The figure shows the complete trajectories until B enters the tail of R_3. This outcome is different from the simulation and can be explained by a higher hazard with respect to R_2(green) than to R_3(magenta) towards the end of the experiment around the time point $40s$. The

trajectories that are confined to the dimensions of the experimental setup also result into smaller expected times than in the simulation, specially for R_4 (gray) in Fig. 7.11.

Fig. 7.12 depicts the optimal control u_i computed with respect to each R_i, $i = 1,2,...4$, as well as the control u_B which results from the one-step look-ahead cost approximation (7.48). This control, just like in the simulation, accounts both for the hazard and expected times to the tail of R_i, $i = 1,2...4$, which are defined by the one-on-one solution and plotted in Fig. 7.11. Any time u_{B_i}, $i = 1,2...4$, are the same for all i, the control $u_B = u_{B_i}$. For the time intervals in which at least one of u_{B_i} is different from the others, we can observe that u_B alternates its value. Since the vehicle's heading angle is the integral of u_B, the impact of fast alterations of u_B to the heading angle is smoothed out. The PI controller that controls the turning rate of the vehicle additionally contributes to the smoothing. Towards the end of the experiment, $u_B = u_{B_3}$ since B is close to the tail of R_3 and has to navigate optimally to it.

7.7 Conclusions

In this paper, we presented an approach to the safe navigation of the nonholonomic fixed wing unmanned aerial vehicle surrounded by multiple other vehicles. The approach is based on the stochastic optimal control which better addresses the uncertainty of the other vehicles' trajectories without assuming a worst case scenario, which is classically considered by the game theory approaches to similar problems. The sce-

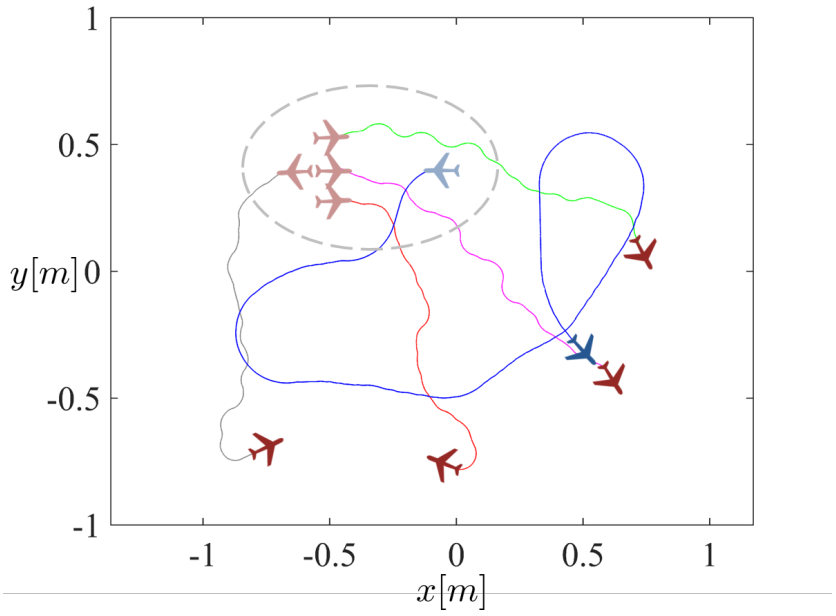


Figure 7.10: Robot experiment trajectories. The initial condition and the trajectories of R_1-R_4 are similar to those in the simulation. The B robot trajectory is colored blue and the R_1-R_4 trajectories are colored red, green, magenta and gray, respectively.

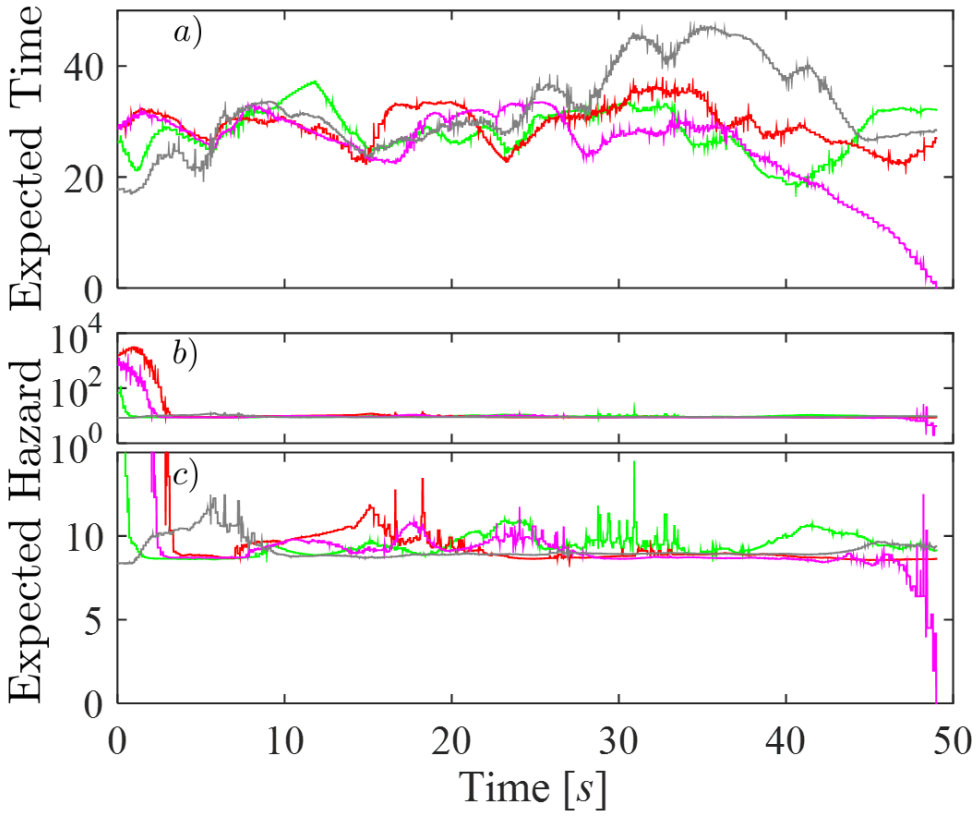


Figure 7.11: Robot experiment expected times and hazards. *a*) Expected time *b*) Hazard values on the log scale *c*) Hazard values on the linear scale. The diagrams corresponding to R_1-R_4 are colored red, green, magenta and gray, respectively.

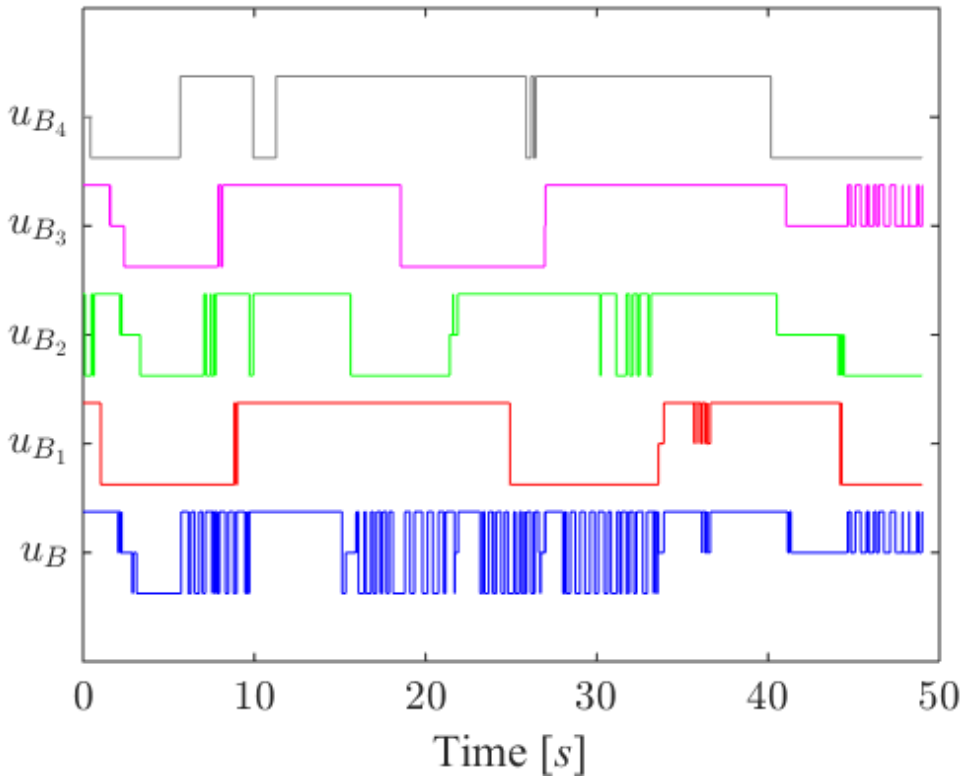


Figure 7.12: Robot experiment control. The control u_B and control variables u_{B_i}, $i = 1,2,3,4$, resulting from the solution of the one-on-one problem and relative positions between B and R_i at every time point of the simulation. All values are in the range $[-0.5, 0.5]$ (see the supplementary video).

nario that is used in our paper was selected to illustrate that safety is not only about collision avoidance.

The problem of the safe navigation in the presence of multiple vehicles is addressed based on the solution of the one-on-one vehicle problem. Therefore, we first explained the stochastic optimal control for that case and introduced the iterative algorithm for computing the avoidance set. The distinguishing property of our approach is that the avoidance set is computed taking into account the vehicle kinematics.

For the multiple vehicles, we derived the result based on the locally consistent Markov chain approximation. This result serves well to illustrate the complexity of the problem and as the basis for proposing the scalable method for computing the control. The scalable method is based on the expected times and hazards resulting from the one-on-one vehicle stochastic optimal control solution. Our scalable approach to the navigation in the multiple vehicle scenario was illustrated by the numerical simulation. We compared the trajectories of the vehicles with the expected times and hazards from our simulation. In addition, we compared the control action of computed solutions with the control actions resulting from the one-on-one solution.

The presented approach is suitable for real-time implementations, therefore, we presented the results from the experiment with small-scale laboratory e-puck robots. The navigation relies on the measurements of relative positions between B and red vehicles ($R_1 - R_4$) to be intercepted. While the measurements were corrupted by the

noise and robot motion model could be more complex, we did not face any significant problem in the implementation.

The approach addresses the uncertainty in the relative positions through the kinematic uncertainty in the motion of red vehicles. The presence of any additional uncertainty would likely require the introduction of an additional stochastic process in the problem formulation. Unless the additional uncertainty in the problem cannot be addressed by an increased intensity of stochasticity in the kinematic model of red vehicles, the problem may be significantly different from the one presented in this paper.

Chapter 8

Stochastic Optimal Control Approach to Navigation with Multi-Obstacle Avoidance

This chapter is a reprint of the paper

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. ”Navigation with Multi-obstacle Avoidance Composed of Stochastic Optimal Controllers.” In 2019 American Control Conference (ACC), pp. 2239-2244. IEEE, 2019.

8.1 Introduction

Obstacle avoidance is one of the fundamental problems in robotics [16]. Here we study a particular obstacle avoidance approach to navigation towards a desired waypoint in a obstacle cluttered dynamically moving environment. In the proposed design we anticipate uncertainty in motion of the obstacles and use stochastic control to reach a waypoint in a time efficient manner.

Obstacle avoidance methods can be roughly grouped into those that are based on path-planning schemes [100], and those that are based on sense and avoid schemes [13]. Our proposed approach is closer to the latter, and examples of other approaches in that group are based on game theory [190, 52], barrier functions [30], reachable sets [1, 122], and potential functions [178, 110].

In this paper we consider the navigation of a fixed velocity UAV flying at a constant altitude, which can be modeled as Dubins vehicle [133]. The UAV navigates towards a waypoint and is subject to a possible collision with unpredictable moving circular obstacles as shown in Fig. 8.1. The optimal control solution to this problem is computationally challenging.

To overcome the computationally challenging optimal control problem, we use the same idea as in a previous work [126] by decomposing the problem into separate smaller state space computable stochastic optimal control problems. Where now in this paper we develop a scalable control strategy for avoiding multiple Brownian moving

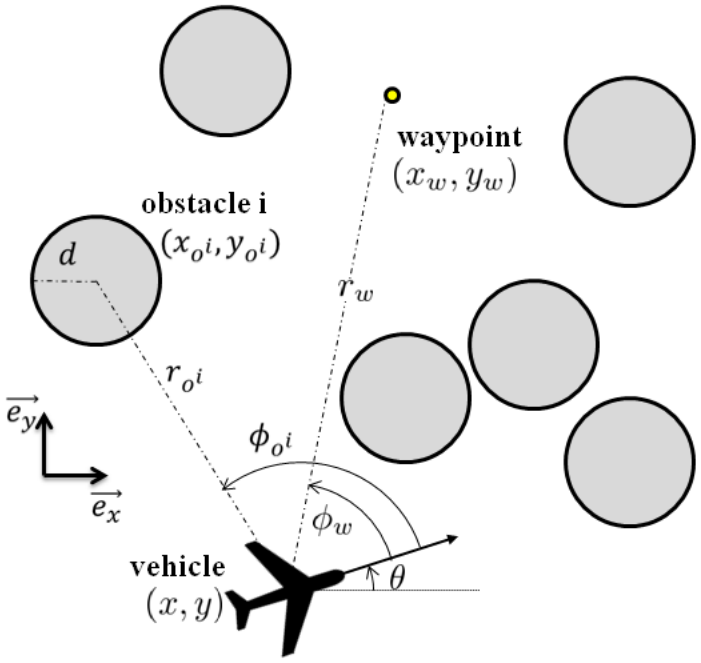


Figure 8.1: Geometry of the UAV navigating towards a waypoint while avoiding obstacles. θ is the heading angles of the UAV. ϕ_{o^i} and ϕ_w are bearing angles of the UAV towards *i*th obstacle and waypoint, respectfully. r_{o^i} and r_w are distances from UAV to *i*th obstacle and waypoint, respectfully.

obstacles. We provide statistical analysis that shows the controller reduces the expected probability of collision in this dynamic obstacle environment while navigating towards a waypoint.

The rest of this paper is organized as follows: First we formulate the optimal control problem for the full state-space of reaching the waypoint while avoiding N obstacles. Then to cope with the computationally demanding optimal control solution for multiple moving obstacles, we show that the multi-obstacle avoidance problem can be simplified to reaching a safe configuration from which the UAV naturally avoids collision with a stationary obstacle. From this we see that we need to solve two fundamental problems that is navigating to a waypoint with no obstacles and second reaching a safe configuration from an obstacle. These two sub-problem solutions are combined via their value functions, and the work of the resulting controller is illustrated through numerical simulations.

8.2 Problem Formulation

Consider the scenario in Fig. 8.1, depicting the UAV navigating in a cluttered obstacle environment towards a waypoint fixed at position (x_w, y_w). The UAV has to navigate to the waypoint while avoiding collision with all obstacles. The UAV flies at constant altitude and is subject to a stochastic wind [8] which can be modeled as Dubins vehicle

with a stochastic wind disturbance

$$dx = v\cos\theta dt + \sigma_w dw_{x_w} \tag{8.1}$$

$$dy = v\sin\theta dt + \sigma_w dw_{y_w} \tag{8.2}$$

$$d\theta = udt \tag{8.3}$$

where the triple (x, y, θ) denotes the UAV's position and heading angle. The Wiener process increments dw_{x_w} and dw_{y_w} are along the x- and y-axis directions, respectfully. The UAV's velocity, $v > 0$, is a known constant, and the control variable is a bounded heading rate $u \in [-u_{max}, u_{max}]$. The parameter $\sigma_w > 0$ is the intensity of the wind disturbance.

Since the UAV *has no knowledge of the obstacles' motion*, the UAV navigation has to anticipate that uncertainty, thereby the kinematics of each ith obstacle for $i = 1..N$ is modeled as 2D Brownian motion, i.e.

$$dx_o^i = \sigma_o dw_{x_o^i} \tag{8.4}$$

$$dy_o^i = \sigma_o dw_{y_o^i} \tag{8.5}$$

where each ith obstacle's position is given by x_o^i, y_o^i, and $dw_{x_o^i}$ and $dw_{y_o^i}$ denote the Wiener process increments along the x- and y-axis directions. The scaling parameter $\sigma_o > 0$ is identical for all obstacles.

8.3 Optimal Control Problem Formulation

The relative position between the UAV and each ith obstacle and waypoint is uniquely defined based on the coordinates $(r_w, \phi_w, r_o^i, \phi_o^i)$ as shown in Fig. 8.1, and to find the relative kinematics between the UAV, waypoint and N obstacles we use Îto calculus similarly as shown in the Appendix of [125] to obtain

$$dr_w = \left(-v\cos(\phi_w) + \frac{\sigma_w^2}{2r_w}\right)dt + \sigma dw_{r_w} \qquad = b_{r_w}dt + n_{r_w}dw_{r_w} \tag{8.6}$$

$$d\phi_w = \left(\frac{v}{r_w}\sin(\phi_w) - u\right)dt + \frac{\sigma_w}{r_w}dw_{(\phi_w+\theta)} \qquad = b_{\phi_w}dt + n_{\phi_w}dw_{(\phi_w+\theta)} \tag{8.7}$$

$$dr_o^i = \left(-v\cos(\phi_o^i) + \frac{\sigma_{ow}^2}{2r_o^i}\right)dt + \sigma_{ow}dw_{r_o^i} \qquad = b_{r_o^i}dt + n_{r_o^i}dw_{r_o^i} \tag{8.8}$$

$$d\phi_o^i = \left(\frac{v}{r_o^i}\sin(\phi_o^i) - u\right)dt + \frac{\sigma_{ow}}{r_o^i}dw_{(\phi_o^i+\theta)} \qquad = b_{\phi_o^i}dt + n_{\phi_o^i}dw_{(\phi_o^i+\theta)} \tag{8.9}$$

where $\sigma_{ow} = \sigma_o + \sigma_w$.

Reaching the waypoint (x_w, y_w) for the UAV is defined by reaching the set

$$\mathscr{T} = \{(r_w, \phi_w) | r_w \leq R_{min}\} \tag{8.10}$$

and collision with an obstacle is defined as the UAV reaching any of the sets

$$\mathscr{A}^i = \{(r_o^i, \phi_o^i) | r_o^i \leq d\} \tag{8.11}$$

that overlap with each ith obstacle. Now we can define the control u as the solution of the optimal control problem with the cost function

$$J(\tilde{x}, u) = \min_u E\left\{g(\tilde{x}(\tau)) + \int_0^\tau dt\right\} \tag{8.12}$$

where $\tilde{x} = (r_w, \phi_w, r_o^i, \phi_o^i)^T$ for $i = 1..N$ is the $2(N+1)$ dimensional state that fully describes the relative positions among the UAV, waypoint and each ith obstacle. The terminal time τ corresponds to the time instant the vehicle reaches either the waypoint or any of the ith obstacles. Notice that τ is the optimal time corresponding for the UAV following the optimal path to reach the waypoint while avoiding all the obstacles. Later we will discuss an approximate optimal control scheme which results in the UAV not necessarily following the optimal path to the waypoint while avoiding collision with all obstacles.

The terminal cost for hitting an obstacle is defined as

$$g(\tilde{x}(\tau)) = \begin{cases} M & \text{if } (r_o^i, \phi_o^i) \in \mathscr{A}^i \\ 0 & \text{if } (r_w, \phi_w) \in \mathscr{T} \end{cases} \tag{8.13}$$

where $M \gg 0$ is a large positive constant to penalize reaching the set $\mathscr{A}^i$ for any of the ith obstacles.

Notice that the optimal control solution is $2(N+1)$ dimensional, which is computationally challenging [138] for large N, i.e. very cluttered obstacles environments. Thus to reduce the computational complexity of solving the optimal control solution for large number of obstacles, we instead reduce the problem into two sub-problems. The first sub-problem is a two-dimensional stochastic optimal control problem for reaching the waypoint without any obstacles. The second is a two-dimensional stochastic optimal control problem for avoiding collision with an obstacle by reaching the configuration

from which the UAV can safely ignore that obstacle. These two sub-problems can be solved independently of each other and then combined together to achieve an approximate optimal control solution for the collision-free navigation problem with multiple obstacles.

The first sub-problem of navigating to the waypoint (x_w, y_w) without any obstacles is given by the control u_w which solves the minimum time optimal control

$$J(\tilde{x}_w, u_w) = \min_{u_w} E\left\{\int_0^{\tau_w} dt\right\} \tag{8.14}$$

where $\tilde{x}_w = (r_w, \phi_w)^T$ is the two-dimensional state that describes the relative positions among the UAV and waypoint, and τ_w is the terminal time which corresponds to the time instant when the UAV reaches the waypoint.

The second sub-problem solution of avoiding collision with the ith obstacle by reaching the configuration from which the UAV can safely ignore that ith obstacle can be defined as the control u_o^i which solves the minimum time optimal control

$$J(\tilde{x}_o^i, u_o^i) = \min_{u_o^i} E\left\{g(\tilde{x}_o^i(\tau_o^i)) + \int_0^{\tau_o^i} dt\right\} \tag{8.15}$$

where $\tilde{x}_o^i = (r_o^i, \phi_o^i)^T$ is the two-dimensional state that describes the relative positions among the UAV and the ith obstacle. The terminal time τ_o^i corresponds to the time instant when the UAV reaches configuration from which the UAV can safely ignore that ith obstacle, and $g(\tilde{x}_o^i) = g(\tilde{x})$ is expressed in expression (8.13). In the original problem (8.12) τ is the lower bound time that can be achieved compared to solving the combined result of these two sub-problems, i.e. $\tau \leq \tau_w + \sum_{i=1}^{N} \tau_o^i$.

8.4 Reaching Waypoint under Brownian Wind Disturbance

This sub-problem is based on the stochastic kinematic model given by expressions (8.6) and (8.7). The model's state variables are the distance r_w and bearing angle ϕ_w, defined as

$$r_w^2 = (x_w - x)^2 + (y_w - y)^2 \tag{8.16}$$

$$\phi_w = \arctan\left(\frac{y_w - y}{x_w - x}\right) - \theta \tag{8.17}$$

where (x, y, θ) and (x_w, y_w) define the UAV's position and heading angle and waypoint's position, respectively (see Fig. 8.1). The optimal control solution u_w^* that achieves the minimum cost in (8.14) can be defined as the same control that finds the steady state solution to the Hamilton-Jacobi-Bellman (HJB) partial differential equation [94] in

$$0 = \min_{u_w}\{\mathscr{L}_{u_w}^w V_w(\tilde{x}_w) + 1\} \tag{8.18}$$

with boundary condition $V_w(\tilde{x}_w) = 0$ for $\tilde{x}_w \in \mathscr{T}$, and V_w can be interpreted as the expected time to reach the waypoint defined by reaching the set (8.10), and $\mathscr{L}_{u_w}^w$ is the differential operator

$$\mathscr{L}_{u_w}^w = b_{r_w}\frac{\partial}{\partial r_w} + \frac{1}{2}(n_{r_w}n_{\phi_w} + n_{r_w}^2)\frac{\partial^2}{\partial r_w^2} + b_{\phi_w}\frac{\partial}{\partial \phi_w} + \frac{1}{2}(n_{r_w}n_{\phi_w} + n_{\phi_w}^2)\frac{\partial^2}{\partial \phi_w^2} \tag{8.19}$$

where b_{r_w}, n_{r_w}, b_{ϕ_w} and n_{ϕ_w} are defined by (8.6) and (8.7), respectively. The solution of (8.18) can be computed using a locally consistent Markov chain discretization of the

HJB equation [94]. The discretization yields a Markov chain with control dependent transition probabilities and converts (8.18) into a dynamic programming problem which can be solved over a discrete space using so-called value iterations [175]. The value iterations result in a discrete approximation of the value function V_w^h and optimal control $u_w^{*\,h}$, where both are in the form of a two-dimensional lookup table. The superscript h indicates that the value function and control are computed for the discretized problem.

To discretize (8.18) in the state space, we use discrete steps Δr_w and $\Delta\phi_w$ for the discretization of r_w and ϕ_w, respectively, and the following upwind discrete approximations of derivatives in (8.18)

$$\frac{\partial V_w}{\partial r_w} \approx \frac{b^+_{r_w^h}}{\Delta r_w}\left(V_w(\tilde{x}_w^h + \Delta r_w) - V_w(\tilde{x}_w^h)\right) - \frac{b^-_{r_w^h}}{\Delta r_w}\left(V_w(\tilde{x}_w^h) - V_w(\tilde{x}_w^h - \Delta r_w)\right) \tag{8.20}$$

$$\frac{\partial^2 V_w}{\partial r_w^2} \approx \frac{n_r}{2\Delta r_w^2}\left(V_w(\tilde{x}_w^h + \Delta r_w) - V_w(\tilde{x}_w^h)\right) - \frac{n_r}{2\Delta r_w^2}\left(V_w(\tilde{x}_w^h) - V_w(\tilde{x}_w^h - \Delta r_w)\right) \tag{8.21}$$

$$\frac{\partial V_w}{\partial \phi_w} \approx \frac{b^+_{\phi_w^h}}{\Delta \phi_w}\left(V_w(\tilde{x}_w^h + \Delta \phi_w) - V_w(\tilde{x}_w^h)\right) - \frac{b^-_{\phi_w^h}}{\Delta \phi_w}\left(V_w(\tilde{x}_w^h) - V_w(\tilde{x}_w^h - \Delta \phi_w)\right) \tag{8.22}$$

$$\frac{\partial^2 V_w}{\partial \phi_w^2} \approx \frac{n_\phi}{2\Delta \phi_w^2}\left(V_w(\tilde{x}_w^h + \Delta \phi_w) - V_w(\tilde{x}_w^h)\right) - \frac{n_\phi}{2\Delta \phi_w^2}\left(V_w(\tilde{x}_w^h) - V_w(\tilde{x}_w^h - \Delta \phi_w)\right) \tag{8.23}$$

where $n_r = |n_{r_w} n_{\phi_w} + n_{r_w}^2|$ and $n_\phi = |n_{r_w} n_{\phi_w} + n_{\phi_w}^2|$. $b^+_{r_w^h} = \max[0, b_{r_w^h}]$, $b^-_{r_w^h} = \max[0, -b_{r_w^h}]$ and $b^+_{\phi_w^h}$, $b^-_{\phi_w^h}$ are defined in the same way. The superscript h indicate terms that are evaluated at the points of the discretized state space $r_w^{h+1} - r_w^h = \Delta r_w$ and $\phi_w^{h+1} - \phi_w^h = \Delta\phi_w$. After the substitution of (8.19)-(8.23) in (8.18), we move all the terms that include

$V_w(r_w^h, \phi_w^h)$ to the left side of the expression (8.18) to obtain

$$V_w^h = \min_{u_w} \Big\{ \Delta t_{u_w}^h + p_{\Delta r_w, u_w}^{+} V_w^h(\tilde{x}_w + \Delta r_w) + p_{\Delta r_w, u_w}^{-} V_w^h(\tilde{x}_w - \Delta r) \\ + p_{\Delta \phi_w, u_w}^{+} V_w^h(\tilde{x}_w + \Delta \phi_w) + p_{\Delta \phi_w, u_w}^{-} V_w^h(\tilde{x}_w - \Delta \phi_w) \Big\} \tag{8.24}$$

where

$$p_{\Delta r_w, u_w}^{\pm} = \Delta t_{u_w}^h \left(\frac{b_{r_w}^{\pm}}{\Delta r_w} + \frac{n_r}{(2\Delta r_w^2)} \right) \tag{8.25}$$

$$p_{\Delta \phi_w, u_w}^{\pm} = \Delta t_{u_w}^h \left(\frac{b_{\phi_w}^{\pm}}{\Delta \phi_w} + \frac{n_\phi}{(2\Delta \phi_w^2)} \right) \tag{8.26}$$

can be interpreted as discrete Markov-chain transition probabilities from the points $(r_w^h \pm \Delta r_w, \phi_w^h \pm \Delta \phi_w)$ of the discrete space to the point (r_w^h, ϕ_w^h), and

$$\Delta t_{u_w}^h = \left(\frac{|b_{r_w}^h|}{\Delta r_w} + \frac{|n_{r_w} n_{\phi_w}|}{(\Delta r_w)^2} + \frac{(n_{r_w})^2}{(\Delta r_w)^2} + \frac{|b_{\phi_w}^h|}{\Delta \phi_w} + \frac{|n_{r_w} n_{\phi_w}|}{(\Delta \phi_w)^2} + \frac{(n_{\phi_w})^2}{(\Delta \phi_w)^2} \right)^{-1} \tag{8.27}$$

where $|b_{r_w}^h| = b_{r_w}^{+} + b_{r_w}^{-}$, $|b_{\phi_w}^h| = b_{\phi_w}^{+} + b_{\phi_w}^{-}$, and $\Delta t_{u_w}^h$ is the implicit time interpolation interval [94].

Expression (8.24) is the discrete version of (8.18) and the discrete approximation V_w^h of the value function V_w can be solved numerically using value iterations [175] starting from an initial guess for the $V_w^h(r_w, \phi_w)$ values.

For the computational domain $\mathscr{K}$ of the solution

$$\mathscr{K} = \{[R_{min}, R_{max}] \times [-\pi, \pi - \Delta \phi_w]\} \tag{8.28}$$

which is the set bounded by the minimal R_{min} and maximal R_{max} distances. Since, in our problem formulation, the angles ϕ_w have a full 2π range, the computational domain

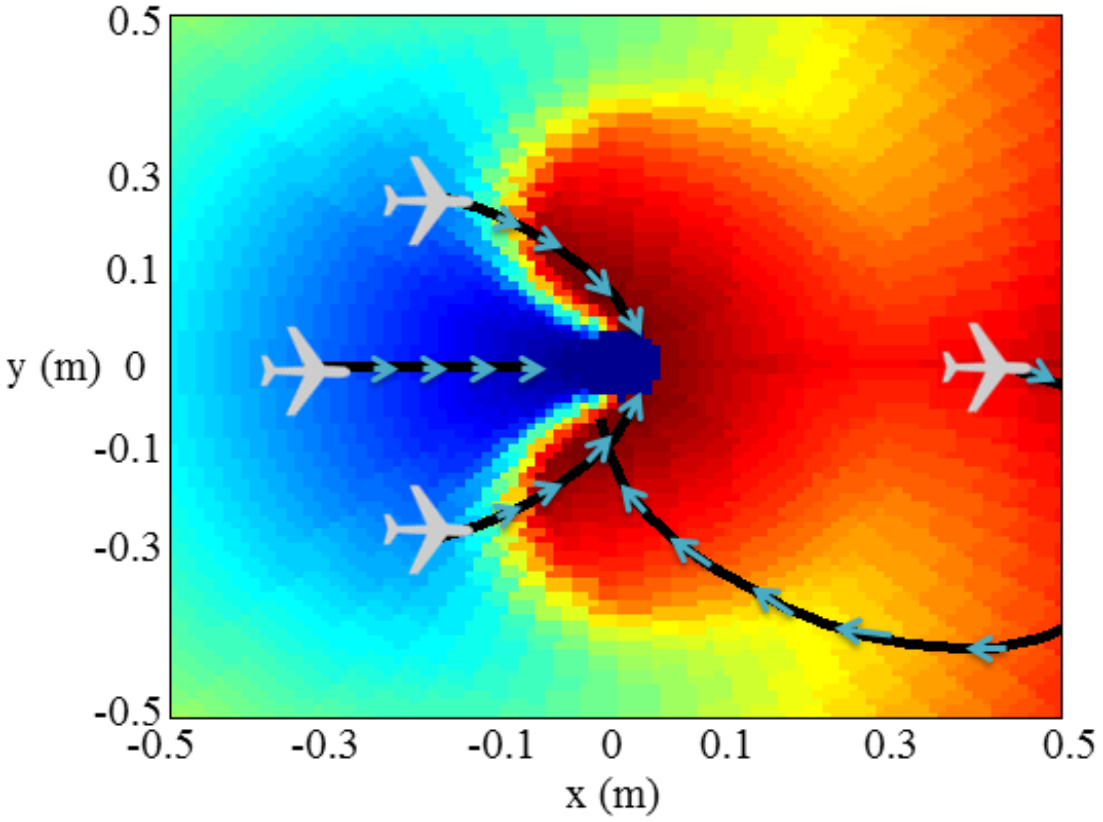


Figure 8.2: The value function for reaching the waypoint shown for various UAVs pointing towards the right. The three UAVs to the left of the waypoint location $(0,0)$ are able to reach the waypoint faster than the UAV that starts to the right of the waypoint.

is periodic, i.e., the pairs of points $(r_w^h, -\pi)$ and $(r_w^h, \pi - \Delta\phi_w)$ are next to each other. Fig. 8.2 presents a projection of the value function on the x- and y-axis computed with the parameters listed in the results section. In Fig. 8.2, the waypoint is placed in the center $(0,0)$ and the UAV has reached the waypoint once it has reached the minimal distance allowed in our computational domain.

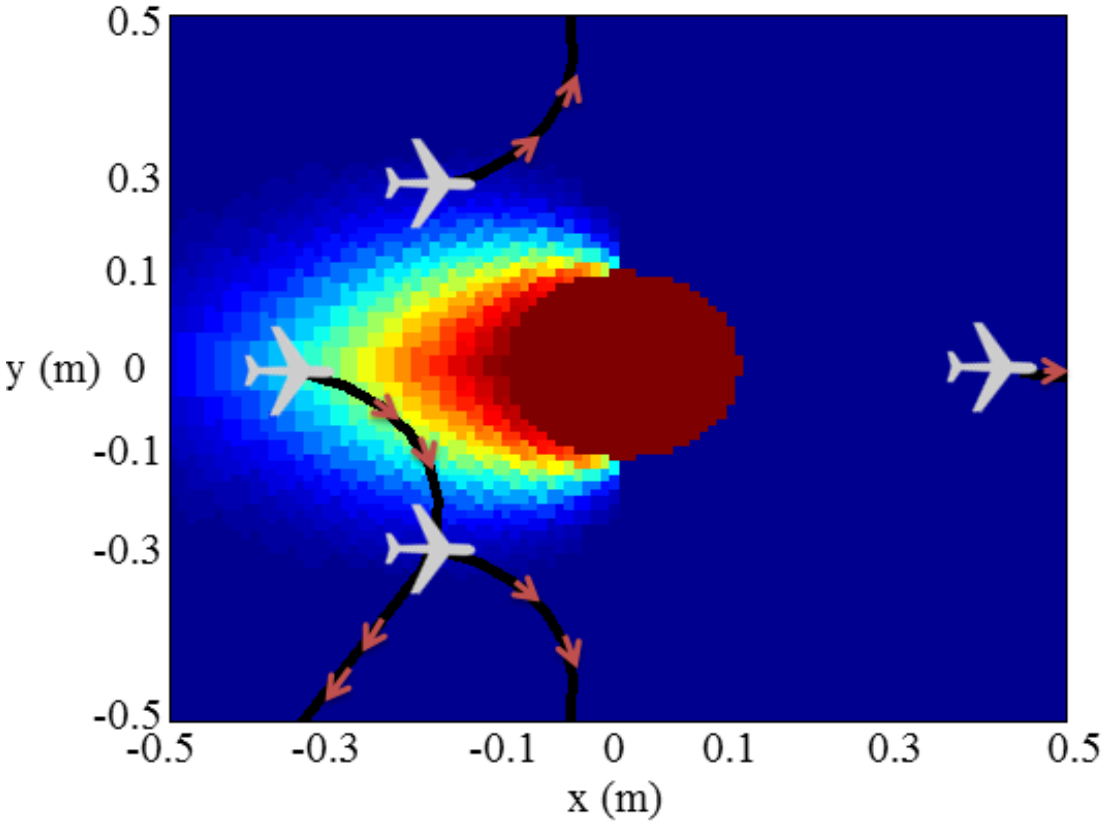


Figure 8.3: The value function for avoiding the obstacle shown for various UAVs pointing towards the right. The three UAVs to the left of the obstacle location $(0,0)$ are in a worse position than the UAV that starts to the right of the obstacle since the rightmost UAV just needs to continue to move towards the right.

8.5 Reaching Avoidance Configuration for a Brownian Moving Obstacle

This sub-problem is based on the stochastic kinematic model given by expressions (8.8) and (8.9). The model's state variables are the distance r_o^i and bearing angle ϕ_o^i for

the ith obstacle which we denote as r_o and ϕ_o for simplicity

$$r_o^2 = (x_o - x)^2 + (y_o - y)^2 \tag{8.29}$$

$$\phi_o = \arctan\left(\frac{y_o - y}{x_o - x}\right) - \theta \tag{8.30}$$

where (x, y, θ) and (x_o, y_o) define the UAV's position and heading angle and obstacle's position, respectively (see Fig. 8.1). The optimal control solution u_o^* that achieves the minimum cost in (8.15) can be defined as the same control that finds the steady state solution to the HJB partial differential equation [94] in

$$0 = \min_{u_o}\{\mathscr{L}_{u_o}^o V_o(\tilde{x}_o) + 1\} \tag{8.31}$$

with boundary conditions $V_o(\tilde{x}_w) = g(\tilde{x}_w)$ defined in (8.13) and $V_o(\tilde{x}_w) = 0$ for $\tilde{x}_w \in \mathscr{T}_o$, which is the set

$$\mathscr{T}_o = \{(r_o, \phi_o) | \phi_o \notin (-\pi/2, \pi/2)\} \tag{8.32}$$

where the value function V_o can be interpreted as the expected cost to reach that set, which is configuration from which the UAV can safely ignore that obstacle while avoiding collision with the obstacle, which is the set defined in (8.11). The differential operator is

$$\mathscr{L}_{u_o}^o = b_{r_o}\frac{\partial}{\partial r_o} + \frac{1}{2}(n_{r_o} n_{\phi_o} + n_{r_o}^2)\frac{\partial^2}{\partial r_o^2} + b_{\phi_o}\frac{\partial}{\partial \phi_o} + \frac{1}{2}(n_{r_o} n_{\phi_o} + n_{\phi_o}^2)\frac{\partial^2}{\partial \phi_o^2} \tag{8.33}$$

where $b_{r_o^i}$, $n_{r_o^i}$, $b_{\phi_o^i}$ and $n_{\phi_o^i}$ are defined by (8.8) and (8.9), respectively, for the ith obstacle. In the same way as the previous section, the solution of (8.31) can be computed

using a locally consistent Markov chain discretization of the HJB equation [94], where we will obtain

$$p^{\pm}_{\Delta r_o,u_o} = \Delta t^h_{u_o}\left(\frac{b^{\pm}_{r_o}}{\Delta r_o} + \frac{n_r}{(2\Delta r_o^2)}\right) \tag{8.34}$$

$$p^{\pm}_{\Delta \phi_o,u_o} = \Delta t^h_{u_o}\left(\frac{b^{\pm}_{\phi_o}}{\Delta \phi_o} + \frac{n_\phi}{(2\Delta \phi_o^2)}\right) \tag{8.35}$$

as the discrete Markov-chain transition probabilities from the points $(r^h_o \pm \Delta r_o, \phi^h_o \pm \Delta\phi_o)$ of the discrete space to the point (r^h_o, ϕ^h_o), and time interval as

$$\Delta t^h_{u_o} = \left(\frac{|b^h_{r_o}|}{\Delta r_o} + \frac{|n_{r_o} n_{\phi_o}|}{(\Delta r_o)^2} + \frac{(n_{r_o})^2}{(\Delta r_o)^2} + \frac{|b^h_{\phi_o}|}{\Delta \phi_o} + \frac{|n_{r_o} n_{\phi_o}|}{(\Delta \phi_o)^2} + \frac{(n_{\phi_o})^2}{(\Delta \phi_o)^2}\right)^{-1} \tag{8.36}$$

where $|b^h_{r_o}| = b^+_{r_o} + b^-_{r_o}$ and $|b^h_{\phi_o}| = b^+_{\phi_o} + b^-_{\phi_o}$.

Fig. 8.3 presents a projection of the value function for avoiding collision with Brownian moving obstacle by reaching safe configuration from that obstacle on the x- and y-axis. The obstacle has radius $d = 0.1$ m and is placed in the center $(0,0)$. The UAV has reached the safe configuration when it is at the set (8.32).

8.6 Solution Composition

The composition of these value functions is inspired by how the effect of harmonic functions [88] add together to create another harmonic potential function without any local minimum. Thus to create a similar effect, we saw our obstacle avoidance value functions V_{o^i} "naturally" forces the UAV to move around the obstacles with the addition

of the waypoint reaching value function V_w, thus we can combine them as

$$V_w + \sum_{i=1}^{N} V_{o^i} \tag{8.37}$$

and by applying a one-step look ahead approach [73] to (8.37, we obtain

$$u_c = \arg\min_u \left\{ \hat{V}_w + \sum_{i=1}^{N} \hat{V}_{o^i} \right\} \tag{8.38}$$

$$\text{subject to } \hat{V}_{o^i} \leq m < M \text{ for } i = 1..N \tag{8.39}$$

where u_c is the one-step look ahead control and $0 < m < M$ is small buffer area from each obstacle such that the UAV does not approach an obstacle arbitrary close. The respective one-step look ahead $\hat{V}_w$ and $\hat{V}_o$ to the value functions V_w and V_{o^i} are expressed as

$$\hat{V}_w = \Delta t_w + \sum_{y_w} p_w(y_w|\tilde{x}_w, u) V_w^h(y_w) \tag{8.40}$$

$$\hat{V}_{o^i} = \Delta t_o^i + \sum_{y_o^i} p_o^i(y_o^i|\tilde{x}_o^i, u) V_{o^i}^h(y_o) \tag{8.41}$$

for all the values functions $V_{o^i}^h$ associated with each obstacle $i = 1..N$ and the value function associated with reaching the waypoint V_w^h. Both y_w and y_o^i are the neighboring states of current state location $\tilde{x}_w$ and $\tilde{x}_o^i$, respectively. The probability transitions p_w and p_o^i are defined in (8.25)-(8.26) and (8.34)-(8.35), respectfully. The time interpolation intervals Δt_w and Δt_o^i are defined in (8.27), (8.36), respectfully.

8.7 Results

The numerical simulation results of this section are based on the Dubins vehicle model (8.1)-(8.3) and obstacle model (8.5) with Dubin's velocity $v = 0.1$ m/s and maximum turning rate $u_{max} = 0.5$ rad/s. The noise intensity for the Brownian moving obstacle is $\sigma_o = 0.038\ \sqrt{m/s}$, and the noise intensity for the stochastic wind is $\sigma_w = 0.001\ \sqrt{m/s}$. Because of the discrete space approximations used in computing the control, we stop the simulation once the Dubins vehicle has reached the minimal distance allowed in our computational domain of $R_{min} = 0.05$ m to the waypoint (x_w, y_w), or has collided with any of the circular obstacles centered at (x_o^i, y_o^i) with radius $d = 0.1$ m for $i = 1..N$. The sample time is $\Delta T = 0.5$ seconds.

The parameters used for computing the computational domain are expressed in (8.28) for both the waypoint and obstacle sub-problems, which are $R_{max} = 2.04$ m and $R_{min} = 0.05$ m. The discrete step sizes are $\Delta r_w = \Delta r_o = (R_{max} - R_{min})/100$ m and $\Delta\phi_w = \Delta\phi_o = 5\pi/180$. The penalty is $M = 10^4$ when the Dubins has collided with an obstacle defined in (8.13).

Example: Fig. 8.4 shows an example simulation run for the scenario depicted in Fig. 8.1, where there are 6 obstacles between the UAV and waypoint in Fig. 8.4*A*. The UAV continues to fly towards the right in Fig 8.4*B* while avoiding the nearest 3 obstacles and reaches the waypoint in Fig. 8.4*C*.

Statistical Analysis: From the Rayleigh distribution, the 2D Brownian moving ob-

stacle with kinematics (8.5) and noise intensity σ_o along both the x- and y-axis can be used to find the expected velocity of the obstacle as

$$\alpha v \Delta T = \sigma_o \sqrt{\Delta T} \sqrt{\frac{\pi}{2}} \tag{8.42}$$

$$\alpha = \sigma_o \frac{\sqrt{\pi}}{v\sqrt{2\Delta T}} \tag{8.43}$$

where α is the ratio of the 2D Brownian moving obstacle's expected velocity compared to the UAV's velocity v. Thus for our parameters $\sigma_o = 0.038\ \sqrt{m/s}$, $v = 0.1$ m/s and $\Delta T = 0.5$ s, we get that the obstacle's expected velocity ratio is $\alpha \approx 0.7$, thus the obstacle's expected velocity is 0.07 m/s.

For a N obstacle scenario within a $20 \times 20\ m^2$ planar region. The probability of collision [67] in this environment assuming that our UAV moves as another circular obstacle without control is

$$\overline{\delta A_{coil}/A} = N\pi \frac{\left((R_{min} + d) + \bar{v}_{12}\Delta T\right)^2}{A} \tag{8.44}$$

where the total area is $A = 400\ m^2$ and the relative velocity between an obstacle and UAV is $\bar{v}_{12} = 0.17$ m/s. For different N shown in Fig. 8.5 we get the expected probability of collision for each obstacle amount.

To compare how well our controller performs against the expected probability of collision in a $20 \times 20\ m^2$ planar region, we run 1000 simulations for the three different obstacle scenarios shown in Fig. 8.5*A*-*C* where the UAV is initially placed in the center of the region with heading angle $\theta = 0$. The Brownian moving waypoint and obstacles

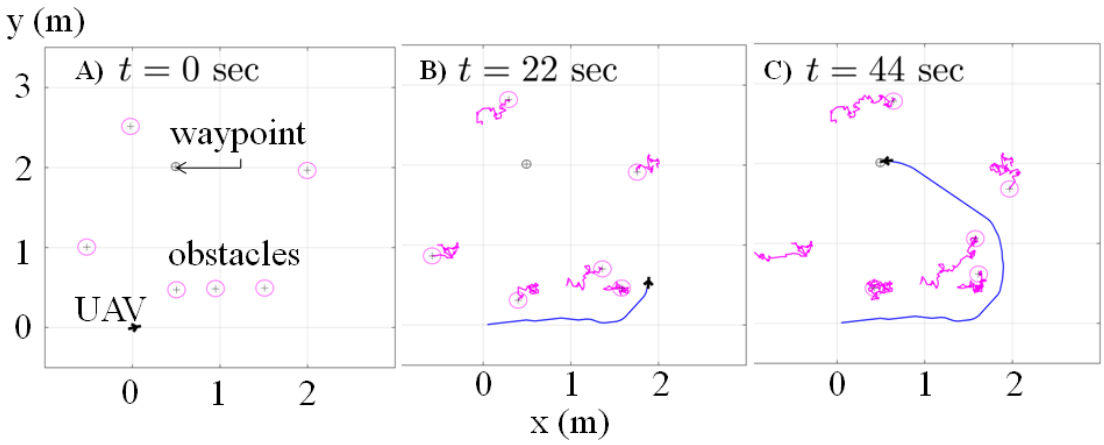
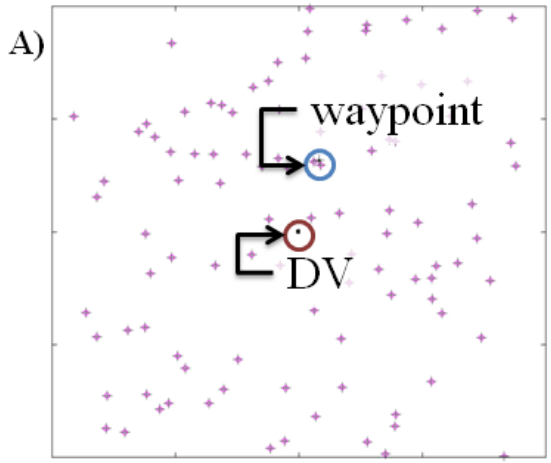


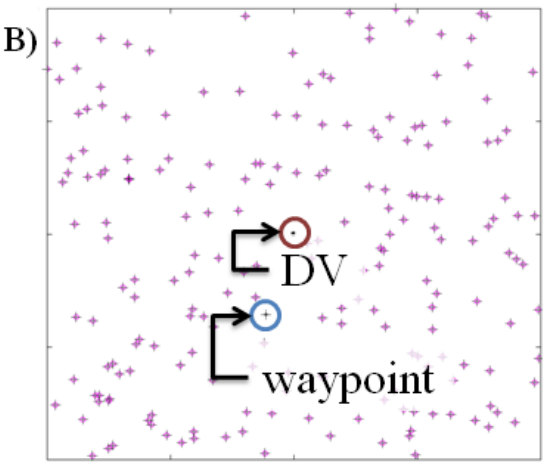
Figure 8.4: Simulation for a 6 obstacle avoidance scenario. Time progresses from A to C, where A) the initial placement of the UAV, waypoint and obstacles and C) UAV reaches the waypoint.

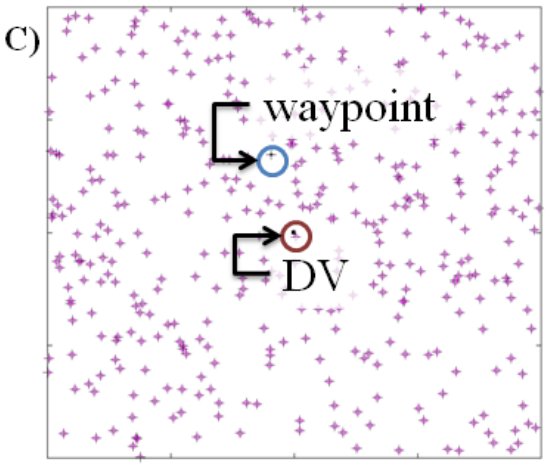
are randomly initialized within the environment such that the UAV is not initially in either of the avoidance sets $\mathscr{A}_o^i$ defined in (8.11) or the target set $\mathscr{T}$ defined in (8.10). Scenarios for the 100, 200 and 400 obstacle cluttered environments is shown in Fig. 8.5A, B and C. Also is shown on the bottom right of Fig. 8.5 is a table that shows the empirical collision probability where the UAV actively avoids collision with the obstacles and the theoretical collision probability without control.

8.8 Conclusions

In this paper, we presented the controller for a Dubins UAV which has the goal to reach a waypoint while simultaneously avoiding collision with multiple randomly moving circular obstacles. Since the UAV does not know each obstacle's trajectory,







	With Control	No Control (p_0)
A)	1.3 ± 0.02 %	4.4 %
B)	5.8 ± 0.06 %	8.7 %
C)	15.9 ± 0.05 %	17.4 %

Figure 8.5: Parts A, B and C are one instance of random initial placement of obstacles and waypoint in a 20×20 (m^2) area where the UAV is located in the center. Results for 1000 simulations for each scenario in A, B and C are summarized in a table at bottom right.

this uncertainty has to be anticipated during the navigation. While the minimum time control problem to reach the waypoint while avoiding all the obstacles can be posed, solving this optimal control problem is hindered by the number of state variables that should be considered. Therefore, we focus on reducing the complexity of the problem by solving two sub-problems. The first is the minimum time optimal control problem to reach a waypoint and the second is the minimum time optimal control problem to reach a safe configuration from the obstacle. Then combining the value functions associated with the solution of each sub-problem is done by solving a multi-objective optimization problem at each control update step. The resulting scalable navigation algorithm is able to reduce the expected probability of collision while navigating towards the waypoint and avoiding collision with Brownian moving obstacles.

Chapter 9

Min-Max Time Efficient Inspection of Ground Vehicles by a UAV Team

This chapter is a reprint of the paper

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. "Min–max time efficient inspection of ground vehicles by a UAV team." Robotics and Autonomous Systems 125 (2020): 103370.

9.1 Introduction

This paper is the result of an exploration to formulate the control of stochastic multi-agent systems by an integration of stochastic optimal control strategies that are designed for a pair of agents. Each pair for our inspection problem is defined as one UAV and one

ground vehicle, and to study such an integration of these control strategies, we consider a scenario with a team of N unmanned air vehicles (UAVs) that are tasked to inspect efficiently with respect to time a group of M ground vehicles. Though the location and orientation angles of the ground vehicles is known to the UAVs, the navigation strategy of ground vehicles is unknown. To anticipate this uncertainty, the headings of ground vehicles are described as stochastic processes, and as a result, the UAV navigation has to be a solution of a feedback stochastic control problem.

A solution of the feedback stochastic optimal control problem for the N UAVs and M ground vehicles has to depend on the relative state-space positions between any pair of UAVs and ground vehicles. This number of state-space variables quickly increases due to the number of combinations in which N UAVs can inspect M ground vehicles, and as such the number should also account for solutions of allowing a single UAV to inspect multiple ground vehicles. In principle, we can formulate the minimum time stochastic optimal control using the Hamilton-Jacobi-Bellman (HJB) equation, but it is a well known result of the so-called *curse-of-dimensionality* [136] that the computational complexity needed to solve the optimal solution goes quickly beyond the computational power of modern computers because of the combinatorial increase of the number of state-space variables. Consequently, we propose an approach in which we first solve for the stochastic optimal control of one UAV inspecting one ground vehicle (*one-on-one*) and use this to formulate a solution for the problem of N UAVs inspecting

M ground vehicles.

The minimum time stochastic optimal control of a single UAV entering the tail sector of another vehicle while safely navigating around it is presented in [125]. The computational method for this type of *one-on-one* agent problem has been improved and used with a scalable value approximation [73] in a complex scenario of a single UAV safely intercepting a group of vehicles [124]. Here we consider a scenario of having multiple UAVs inspecting multiple ground vehicles and we assume that the UAVs are able to avoid each other, or more realistically, they are able to fly at different altitudes. Consequently, collision avoidance is not considered and the main problem lies in how to assign [34, 166] each UAV to ground vehicles, so that the inspection time for all ground vehicles is time efficient. Other authors have studied a related assignment problem [24, 26], but in terms of path-planning [100].

Non path-planning, i.e., feedback-based navigations are frequently found in a line of work called sense and avoid [13], and game theoretic approaches to safe navigation [122]. They also appear in the context of pursuit-evasion games [81], including two car-like vehicles investigated in [64]. Scalable algorithms for multiple-pursuers/single-evader games have been considered in [54, 182]. Game theoretic problems for multi-pursuit and multi-evasion strategies using a hierarchical or decomposition algorithmic approach [62, 102] are more similar to the multi-agent problem in this paper.

In this work, we approach the problem of N UAVs inspecting M ground vehicles

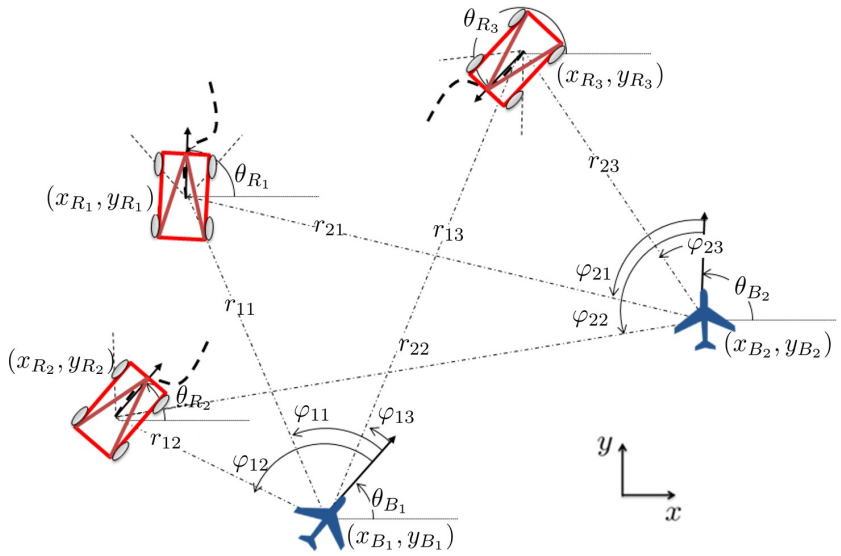


Figure 9.1: Geometry of the multi-agent/multi-target problem: θ_{B_i} and θ_{R_j} are the heading angles of the ith blue and jth red, respectfully. $\alpha_{ji} = \theta_{R_i} - \theta_{B_j}$ is the difference between the headings, and ϕ_{ji} is the bearing angle of the jth blue to ith red. r_{ji} is the distance between jth blue to ith red.

as a Markov inequality-based switching problem which is inspired by the result [120] for one UAV inspecting multiple ground vehicles. Dealing here with N UAVs, the switching is among possible inspection assignments of UAVs to the ground vehicles to be inspected. No knowledge of the ground vehicles navigation strategy or intent is known by UAVs, therefore, each of the ground vehicles heading angles is anticipated to be a Brownian random walk. The contribution of this paper is that it solves the stochastic multi-agent problem using *one-on-one* stochastic optimal control solutions and their composition with the promise of time-efficient navigation. The presented approach scales well with the number of agents and allows for real-time computations of control actions.

The paper is organized as follows. Section 9.2 discusses the problem, which stating succinctly is controlling a team of UAVs to inspect all ground vehicles in minimum time. The stochastic optimal control of a single UAV to inspect a single ground vehicle in minimum time is presented in Section 9.3. The time optimal assignment and dynamic re-assignment are discussed in Section 9.4 and Section 9.5, respectively. Section 9.6 shows results of our numerical simulations, and Section 9.7 provides conclusions.

9.2 Problem Formulation

Let us consider a scenario with five agents depicted in Figure 9.1. Three of the agents, labeled with R_i, $i = 1,2,3$, are ground vehicles with equal speeds $v_{R_i} = v_R$, and we refer to them as red agents. The other two agents, B_j, $j = 1,2$, are fixed-wing aerial vehicles flying at different altitudes at equal speeds $v_{B_j} = v_B$, and we refer to them as blue agents. The constant speed assumption approximates that the UAVs cannot stop and that without energy constraints, the UAVs will fly at maximum speeds to cover larger areas of interest. We will also assume that the speed of blue (aerial) agents is larger than the speed of red (ground) agents, $v_B > v_R$. The kinematics of the jth blue

agent is described using the deterministic kinematics of a Dubins vehicle given as

$$dx_{B_j} = v_B \cos(\theta_{B_j})dt \tag{9.1}$$

$$dy_{B_j} = v_B \sin(\theta_{B_j})dt \tag{9.2}$$

$$d\theta_{B_j} = u_{B_j}dt \tag{9.3}$$

where the couple (x_{B_j}, y_{B_j}) and θ_{B_j} describe the 2D blue agent's position and heading angle, respectively. The control input for each blue is a bounded turning rate $u_{B_j} \in [-1, +1]$.

We will assume that each blue agent knows the location of each red agent's position and orientation, which corresponds to knowing the full relative states between each jth blue agent and ith red agent described by

$$r_{ji} = \sqrt{(x_{R_i} - x_{B_j})^2 + (y_{R_i} - y_{B_j})^2}, \tag{9.4}$$

$$\phi_{ji} = \arctan\left(\frac{y_{R_i} - y_{B_j}}{x_{R_i} - x_{B_j}}\right) - \theta_{B_j}, \phi \in [-\pi, \pi) \tag{9.5}$$

$$\alpha_{ji} = \theta_{R_i} - \theta_{B_j}, \alpha \in [-\pi, \pi) \tag{9.6}$$

where r_{ji} is the distance between B_j and R_i, ϕ_{ij} is the bearing angle from B_j to R_i, and α_{ji} is the difference between the R_i and B_j heading angles. The definitions of these coordinates are illustrated in Figure 9.1. Based on these relative coordinates, we define

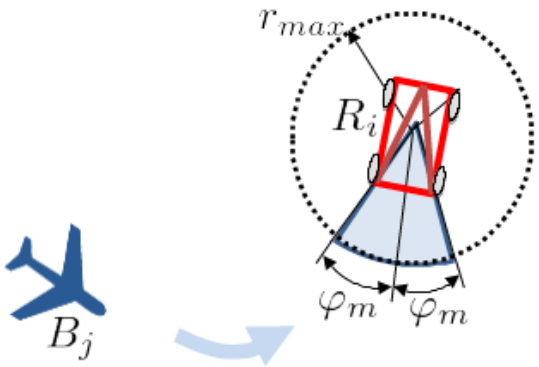


Figure 9.2: Tail sector of red agent R_i described by (9.7) where the blue agent B_j has to enter to inspect the red agent.

the tail sector $\mathscr{T}_i(t)$ for each ith red agent as

$$\mathscr{T}_i(t) = \left\{ (r,\phi,\alpha): \begin{array}{c} r \le r_{max} \\ |\phi| \le \phi_m \\ |\alpha| \le \alpha_m \end{array} \right\} \tag{9.7}$$

where α_m and ϕ_m are the angles defining the tail sector width and alignments of the blue agents' heading angles, and r_{max} defines the length of the tail sector. A successful inspection is defined using (9.7) as $(r_{ji}, \varphi_{ji}, \alpha_{ji}) \in \mathscr{T}_i$, i.e., it occurs when B_j is in this sector of R_i, and its heading is aligned with that of R_i. An illustration of the tail sector is given in Fig. 9.2.

The control problem in this paper is to find the control for the blue agent team providing that the time of inspection of all reds is minimal, i.e., that the blue team visit all tail sectors of red agents in minimum time. However, blue agents *have no knowledge of the red ground vehicles' navigation strategy, i.e., the motion of the corresponding tail*

sectors. To anticipate that uncertainty, the kinematics of each red R_i, $i = 1, ...M$ agent is modeled by the stochastic kinematics

$$dx_{R_i} = v_R \cos(\theta_{R_i})dt \tag{9.8}$$

$$dy_{R_i} = v_R \sin(\theta_{R_i})dt \tag{9.9}$$

$$d\theta_{R_i} = \sigma_R dw_i \tag{9.10}$$

where the position of R_i is given by x_{R_i}, y_{R_i} and the heading angle is $\theta_{R_i} = \int_0^t \sigma_R dw_i$. The latter describes that the heading angles of red agents are random walks since dw_i denotes the Wiener process increments. The scaling parameter σ_R is identical for all red agents.

A solution of the minimum time feedback optimal control problem for N blue and M red agents depends on the number of variables that increases quickly due to the number of combinations in which N blue agents can visit M tail sectors of red agents. This number should also account for solutions allowing that a single blue agent may need to visit multiple red agent tail sectors. While in principle we can formulate the stochastic optimal control using the Hamilton-Jacobi-Bellman (HJB) equation and all necessary coordinates describing relative positions of blues and reds, the computational complexity of such a solution goes quickly beyond the computational power of modern computers because of the number of relative coordinates, as well as due to the so-called *curse-of-dimensionality* [136].

We approach the problem by dividing it into (1) the minimum time stochastic op-

timal control problem of one blue inspecting one red agent (*one-on-one*), and (2) the problem of computing optimal inspection assignments for the N blue agents to time efficiently inspect M reds. This approach allows us to formulate the solution for the navigation strategy of N blues that scales well with the number of agents and guarantees that all M red agents are inspected efficiently with respect to time. This guaranteed property and time efficiency are achieved by the optimality of the *one-on-one* problem and assignment updates when the assignment leads to a probabilistic chance for the decrease of inspection time for all red agents by the blues.

9.3 Minimum Time Stochastic Optimal Control

In this section, we deal with the scenario of a single blue B_j that enters the tail sector $\mathscr{T}_i$ of a single red R_i in the minimum expected time, thus $i = j = 1$. To simplify the notation, in this section we will drop the subscripts i and j from B_j, R_i and $\mathscr{T}_i$.

Using Itô calculus, the kinematic model of B (9.1)-(9.3), the kinematic model of R (9.8)-(9.10) and the definitions of relative coordinates between B and R (9.4)-(9.6), we can derive the following stochastic differential equations describing the evolutions of

relative positions between B and R

$$dr = (v_R \cos(\phi - \alpha) - v_B \cos(\phi))dt = b_r dt \tag{9.11}$$

$$d\phi = \left(-u_B + \frac{-v_R \sin(\phi - \alpha) + v_B \sin(\phi)}{r} \right) dt = b_\phi dt \tag{9.12}$$

$$d\alpha = -u_B dt + \sigma_R dw = b_\alpha dt + \sigma_R dw \tag{9.13}$$

The minimum expected time control u_B for B to reach the target set $\mathscr{T}(t)$ of R is the control that minimizes the cost

$$\mathscr{J}(u_B) = E\left\{ g(\tau) + \int_0^\tau 1 dt \right\} \tag{9.14}$$

where $g(\tau) = g(r(\tau), \phi(\tau), \alpha(\tau))$ is the terminal cost defined as

$$g(\tau) = \left\{ \begin{array}{ll} 0 & \text{if } (r(\tau), \phi(\tau), \alpha(\tau)) \in \mathscr{T} \\ M & \text{if } (r(\tau), \phi(\tau), \alpha(\tau)) \in \mathscr{P} \end{array} \right\} \tag{9.15}$$

and the set $\mathscr{P}$ is defined as

$$\mathscr{P}(t) = \left\{ (r, \phi, \alpha): \begin{array}{r} r \leq r_{max} \\ |\phi| > \phi_m \\ |\alpha| > \alpha_m \end{array} \right\} \tag{9.16}$$

The cost function $\mathscr{J}$ is constructed to yield the optimal control u_B that minimizes the time for B to reach the target set $\mathscr{T}$ and avoids configurations in which B is in the proximity of R, but not in the target set $\mathscr{T}$. This is expressed by the definition of terminal cost $g(\tau)$ in which there is a large positive penalty $M \gg 0$ for the set $\mathscr{P}$ and no penalty for reaching the set $\mathscr{T}$.

The cost function $\mathscr{J}(u_B)$ gives rise to the stochastic HJB equation defining the evolution of the cost-to-go function $U = U(r, \phi, \alpha)$ for the optimal control

$$0 = \min_{u_B} \left\{ b_r \frac{\partial U}{\partial r} + b_\phi \frac{\partial U}{\partial \phi} + b_\alpha \frac{\partial U}{\partial \alpha} + \sigma^2 \frac{\partial^2 U}{\partial \alpha^2} + 1 \right\} \tag{9.17}$$

with the two boundary conditions $U = 0$ for all $(r, \phi, \alpha) \in \mathscr{T}$ and $U = M$ for all $(r, \phi, \alpha) \in \mathscr{P}$. The solution of the HJB equation yields the cost-to-go function U and the corresponding optimal state feedback control $u_B = u_B(r, \phi, \alpha)$. Once the optimal control is computed, it can be used to compute the expected time $V = V(r, \phi, \alpha)$ to reach the target under the optimal control by computing the steady-state solution of the backward Kolmogorov equation

$$0 = b_r \frac{\partial V}{\partial r} + b_\phi \frac{\partial V}{\partial \phi} + b_\alpha \frac{\partial V}{\partial \alpha} + \sigma^2 \frac{\partial^2 V}{\partial \alpha^2} + 1 \tag{9.18}$$

with the boundary condition $V = 0$ for all $(r, \phi, \alpha) \in \mathscr{T}$ and reflective boundary condition elsewhere on the boundary of the solution domain.

The solution of the optimal control problem used in this paper is based on the so-called locally consistent Markov chain discretization of the HJB equation [56]. The discretization yields a Markov chain with control u_B-dependent transition probabilities while the problem of solving the HJB equation is converted into a discrete state space dynamic programming problem that can be solved using value iterations [94]. Further details about the numerical method can be found in [124] and [125]. In both of these papers, the controllers have been implemented and tested with ground robots as a part

of different control problems. Similar controllers have been used for the navigation of a small UAV in the presence of stochastic winds [8]. A general explanation of the method for the control of nonholonomic vehicles is given in [38]. The method has also been used for target tracking problems [12] and flight tested with real UAVs [119].

Once the control is computed, we can use it to compute the solution of the Kolmogorov equation (9.18) to obtain expected time V, i.e. its discrete space numerical representation. Due to the similarity of the Kolmogorov equation (9.18) with (9.17), the computations are based on the same discretization scheme and value iterations [94], except that instead of *min* operator, the value iterations are based on already computed optimal control.

The units in (9.4)-(9.6), which we used to compute the numerical optimal control in the example of this paper, are normalized so that all the angles are in radians and the velocities are $v_B = 0.1$ and $v_R = 0.05$. The noise scaling parameter $\sigma_R = 10\pi/180$ and the maximum turning rate of each blue is $u_{max} = 0.5$. The tail sector (9.7) to be reached by the blue is defined by $r_{max} = r_{min} = 0.05$, $\phi_m = 10\pi/180$, and $\alpha_m = 20\pi/180$, where r_{max} is length of the tail sector and r_{min} is the smallest distance allowed in our computational domain given as

$$\mathscr{K} = \{[r_{min}, R_{max}] \times [-\pi, \pi - \Delta\phi] \times [-\pi, \pi - \Delta\alpha]\} \tag{9.19}$$

with $R_{max} = 2.04$, i.e. the largest distance allowed in our computational domain. The domain (9.19) is discretized with the steps $\Delta r = (R_{max} - r_{min})/99 \approx 0.0201$ and $\Delta\phi =$

$\Delta\alpha = 5\pi/180$ in the direction of r, ϕ, α state space variables. Since the angles ϕ and α in our problem formulations have full ranges, in the discretized computational domain, the pairs of points $(r^h, -\pi, \alpha^h)$ and $(r^h, \pi - \Delta\phi, \alpha^h)$, as well as $(r^h, \phi^h, -\pi)$ and $(r^h, \phi^h, \pi - \Delta\alpha)$ are next to each other, i.e., we use periodic boundary conditions along the ϕ and α state space variables. Other boundary conditions of the discrete approximation based on the locally consistent Markov chain approximation method take into account (9.15) which contains the boundary conditions for the tail sector (9.7) and (9.16).

9.4 Time Optimal Inspection Assignment

A general formulation of the assignment problem in which $B_j,\ j = 1, ...N$ agent should inspect R_i, $i = 1, ...M$ should take into account that a single B_j may be assigned to a sequence of R_i agents and that all R_i agents are assigned to at least one B_j agent. This type of problem can be formulated as an optimization problem on a graph. In the most general case, the graph nodes would be blue and red agents, and the edges between them would have an associated time to travel from any blue agent to any red agent. The solution would be a sequence of red agents that should be visited by each blue. However, in this paper we assume that the red agents are moving stochastically, which creates a major challenge for an optimal assignment solution. On the other hand, we can rely on the results obtained from the stochastic optimal control one-on-one

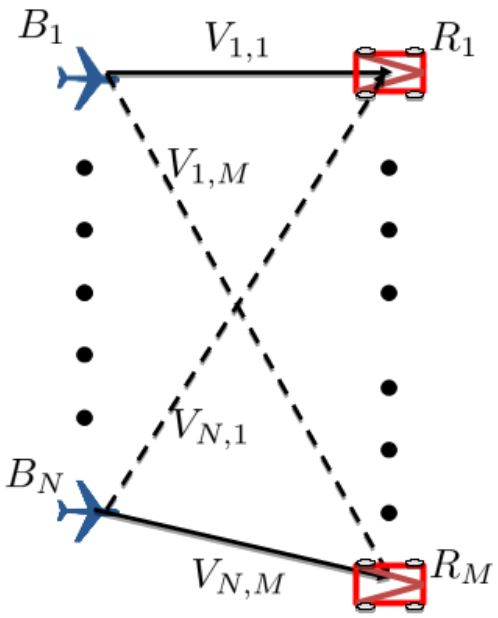


Figure 9.3: Assignment graph problem, where $V_{i,j}$ denotes the expected time of B_j inspecting R_i. The dashed lines denote a possible assignment and the solid lines denote an assigned blue-red pair.

solution presented in the previous section for easily obtaining the edge costs associated with the graph assignment problem.

To address the assignment problem, we consider the optimization on a graph in which there are no edges among the R_i agents. The graph is depicted in Fig. 9.3 and its edges are associated with the expected times $V_{j,i}(t) = V(r_{ji}(t), \phi_{ji}(t), \alpha_{ji}(t))$ of B_j reaching $\mathscr{T}_i$, i.e., jth blue inspecting the ith red agent. Given these expected times $V_{j,i}$, we propose an assignment that at $t = 0$ minimizes the expected time of inspection to the

last red, which corresponds to the longest expected time $V_{j,i}$. This can be formulated as

$$C^o = \min_{\mathscr{A}} \left\{ \max_{j \in \mathscr{N}_0} \{V_{j,i}(0) z_{j,i}(0)\} \right\}, \; z_{j,i} \in \{0,1\} \tag{9.20}$$

$$j \in \mathscr{N}_0 \subseteq \{1,2,...N\}, i \in \mathscr{M}_0 \subseteq \{1,2,...M\} \tag{9.21}$$

$$\text{subject to} \begin{cases} \sum_{i \in \mathscr{M}_0} z_{j,i}(0) = 1, \;\; \text{for all } j \\ \sum_{j \in \mathscr{N}_0} z_{j,i}(0) \geq 1, \;\; \text{for all } i, \text{ if } |\mathscr{N}_0| \geq |\mathscr{M}_0| \\ \sum_{j \in \mathscr{N}_0} z_{j,i}(0) \leq 1, \;\; \text{for all } i, \text{ if } |\mathscr{N}_0| < |\mathscr{M}_0| \end{cases} \tag{9.22}$$

where C^o is the initial optimal cost corresponding to the initial optimal assignment A^o. The minimization is over a finite set of all possible assignments $\mathscr{A} = \{A_1, A_2, ...\}$, and each assignment A_k is uniquely defined by the assignment variables $z_{j,i}$ that can be depicted by the graph in Fig. 9.3. The assignment variable $z_{j,i} = 1$ if B_j is assigned to inspect R_i, otherwise, $z_{j,i} = 0$. The sets $\mathscr{N}_0$ and $\mathscr{M}_0$ contain indexes of all blue and red agents, respectively, that are included in the assignment, and $|\mathscr{N}_0|$, $|\mathscr{M}_0|$ denote cardinal numbers of these sets.

The first constraint in (9.22) states that the assignment has to have each B_j assigned to one R_i. The second one states that if the number of blues is greater than or equal to the number of reds, then each R_i must be assigned to at least one blue. Finally, the third constraint covers the case when the number of reds is greater than the number of blues, which states that then each red must be assigned to at most one blue. Hence there will be reds that are left unassigned for the third constraint case.

By introducing a new decision variable μ, we can rewrite the optimization problem

(9.21)-(9.22) as a mixed integer linear programming (MILP) problem

$$C^o = \min \mu \tag{9.23}$$

$$z_{j,i} \in \{0,1\},\ j \in \mathscr{N}_0 \subseteq \{1,2,...N\}, i \in \mathscr{M}_0 \subseteq \{1,2,...M\} \tag{9.24}$$

$$\text{subject to} \begin{cases} V_{j,i}(0) z_{j,i} \leq \mu, \text{for all } i,j \\ \mu \geq 0, \\ \sum_{i \in \mathscr{M}_0} z_{j,i}(0) = 1, \ \text{ for all } j \\ \sum_{j \in \mathscr{N}_0} z_{j,i}(0) \geq 1, \ \text{ for all } i, \text{ if } |\mathscr{N}_0| \geq |\mathscr{M}_0| \\ \sum_{j \in \mathscr{N}_0} z_{j,i}(0) \leq 1, \ \text{ for all } i, \text{ if } |\mathscr{N}_0| < |\mathscr{M}_0| \end{cases} \tag{9.25}$$

The MILP minimizes the variable μ, and $\max\{\cdot\}$ in the objective function (9.20) is replaced by the inequality constraints $V_{j,i}(0) z_{j,i} \leq \mu$ for all $j \in N_0$ and $i \in M_0$. The second constraint $\mu \geq 0$ is not necessary, but it clarifies that μ has to be positive since it is a time variable.

9.4.1 More or equal number of blues than reds

When the number of available blues is equal or greater than the number of reds to be inspected, i.e. $|N_0| \geq |M_0|$, the initial optimal assignment (9.20) will always return the *absolute* maximum time required to inspect the last red.

The proof is straightforward as since $|N_0| \geq |M_0|$ the initial optimization problem (9.20) second constraint gets activated. The second constraint guarantees that each red is assigned a blue, hence the $\max_j \{V_{j,i}(0) z_{j,i}(0)\}$ is over all pairs of blues to reds

(which is not true if there are less blues than reds). Since each red is assigned, it indicates that every expected inspection time is being minimized. Since every inspection time is minimized then the *absolute* maximum time required to inspect the last red is minimized. Therefore the initial optimal assignment returned the *absolute* maximum time required to inspect the last red.

9.4.2 Less number of blues than reds

When the number of available blues is less than the number of reds to be inspected, i.e. $|N_0| < |M_0|$, the initial optimal assignment (9.20) may not necessarily return the *absolute* maximum time required to inspect the last red.

Consider the scenario where there are two reds and one blue positioned as shown in Fig. 9.4, which falls under the case of less blues than reds. The positions and heading angles of each red is such that they are initially separated by a distance $d > 0$ and will move in opposite directions thus further increasing the distance between them. The blue is positioned such that its inspection time to R_1 and R_2 is initially the same, i.e. $V_{11} = V_{12}$. As a result, the maximum is ambiguous since the initial optimal assignment returned from (9.20) can either be V_{11} or V_{12} corresponding to first inspecting R_1 or R_2. Let us assume that V_{11} is chosen as the maximum time (the same proceeding logic would apply to V_{12}). Therefore blue goes to inspect R_1. If we assume that both reds continue to move forward in a straight line, then once the blue inspected R_1 at expected

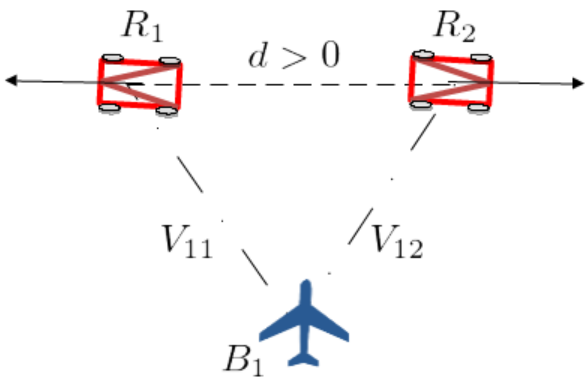


Figure 9.4: Counterexample with two reds and one blue showing when the initial optimal assignment (9.20) will fail to return the *absolute* maximum time to minimize.

time V_{11}, R_2 is at a nonzero distance away from R_1 and the blue. Hence the blue must travel an additional distance which requires additional nonzero time to inspect the other red. This additional time needed indicates that V_{11}, the returned maximum time required to inspect the last red, is not the *absolute* maximum time.

9.5 Dynamic Switching for Re-Assignment

In this section, we will summarize the Markov inequality switching rule [120] pertaining to assignment switchings. In order to determine how we can switch between assignments, we define the cost $C_k(t)$ of the assignment A_k evaluated at time t as

$$C_k(t) = \max_{j \in \mathcal{N}_0} \{V_{j,i}(t) z_{j,i}(t)\}, \; i \in \mathcal{M}_0 \tag{9.26}$$

where $C_k(t)$ is current cost of assignment A_k. Notice that we had C^o in the previous section which is $C^o = \min_k \{C_k(0)\}$ in expression (9.26).

Since red agents are moving stochastically, it is obvious that the initial configuration with cost C^o defined by the solution of (9.20) may be inferior to any other configuration A_k for which $C_k(t) < C^o$. To pursue the idea of optimality of assignment at every time instant t, one may think about solving the optimization (9.20) at every time instant (greedy approach). However, note that the navigation strategy is based on the minimization of expected times to the target sets $\mathscr{T}_i$, therefore, there is a non-zero probability for the increase of times to target sets. Because of that, although there may exist multiple assignments that result in the inspection of all red agents, once we start switching among them using the greedy approach, it can result in an infinite sequence of assignment switchings without ever inspecting any red agent. This is a well-known characteristic of the so-called *hybrid systems*.

To resolve the problem of an infinite sequence of switchings, here we propose to use the switching rule that was presented and analyzed in [120]. The rule is that if at time t

$$\frac{\min\{C_k(t), C_k(\tau^s)\} - C_m(t)}{\min\{C_k(t), C_k(\tau^s)\}} \geq p, \quad p \in (0,1) \tag{9.27}$$

then the switching from the assignment A_k to A_m takes place (notice that initially $C_k(0) = C^o$, the initial optimal assigment). In this rule, k is the index of the current assignment and m is the index of the next assignment, which is not necessarily $k+1$, $\tau^s < t$ is the time at which the assignment becomes A_k and s is the number of total switchings from $t = 0$. Hence after the switching at time $\tau^{s+1} = t$, s is incremented by

one.

If τ_F defines the time to inspect all red agents, then the above rule guarantees

$$Pr\{\tau_F < \min\{C_k(t), C_k(\tau^s)\}\} \geq p \tag{9.28}$$

which means that the switching happens only if it provides that the time τ_F is with a probability p shorter than the smallest of the current expected time to inspect all red agents and the expected time to inspect all red immediately after τ^s, i.e., the last switching of the assignment. Here we should acknowledge that switching probabilistically shortening the time to inspect the last red, but it does not explicitly minimize the time to capture any specific red. For example, a certain blue can come close to inspect a particular red but can change its assignment as long as the re-assignment reduces the expected time of inspection to the last red.

The above rule is introduced and analyzed in [120] for the case of a single blue and multiple red agents until the first red agent is inspected. The switching rule also relied on the existence of a threshold $\underline{C} > 0$, so that the switching stops once $C(\tau^s) \leq \underline{C}$. As in our current problem, the threshold $\underline{C}$ is the smallest nonzero expected time in the discrete state-space of the numerical stochastic optimal control solution, i.e. the neighboring state-space cells surrounding the discretized tail sector. We also have to deal with an additional level of complexity in our problem such as the time instants when a blue reaches the tail sector of a red, as other blues might be still trying to reach their respective reds. At this point, there are multiple ways to resolve this issue.

For example, the blue that has currently inspected a red can be excluded from further considerations, hence $|N_0|$ will decrease in size, and that blue will rendezvous back to base to await further instructions, or it can be assigned to inspect another red. In the latter case, it may happen that multiple blues can be assigned to a single red. All of these possibilities are covered by expressions (9.21) and (9.22). Therefore, at the time the first B_j reaches the target set $\mathscr{T}_i$ that corresponds to R_i, we propose to reset the time to $t = 0$, update $\mathscr{N}_0$ (if a blue was excluded) and $\mathscr{M}_0$, and proceed with one of two options. The first is if $|\mathscr{N}_0| \geq |\mathscr{M}_0|$ then we proceed with the use of the switching rule until all reds are captured. The second is if $|\mathscr{N}_0| < |\mathscr{M}_0|$ then we solve again (9.20) until $|\mathscr{N}_0| \geq |\mathscr{M}_0|$ at which point we proceed with the first option.

Perhaps, the most beneficial aspect of the switching rule is that not only does it guarantees that after every switch, the maximum expected time to inspect the last red is decreased, but that the target set of the next red will be reached in a finite expected time. Hence, by summing all the times of reaching $\mathscr{T}_i$ for each respective red, we can conclude that all reds are inspected in a finite expected time.

In summary, after every inspection of a red by a blue we recompute the optimal assignment for the current positions of all agents (greedy approach), and then proceed with switching among different assignments with the switching rule that guarantees that the expected inspection time of the next red agent is finite.

Initial sets $\mathcal{N}_0$ and $\mathcal{M}_0$
FirstAssignment $= True$
While $|\mathcal{M}_0| \neq 0$
  If $(|\mathcal{N}_0| < |\mathcal{M}_0|)$ or (FirstAssignment $== True$) ;then
    $C_0 = \min \mu,\ z_{j,i} \in \{0,1\}$
    $j \in \mathcal{N}_0 \subseteq \{1,2,...N\}, i \in \mathcal{M}_0 \subseteq \{1,2,...M\}$

$$\text{subject to} \begin{cases} V_{j,i}(0) z_{j,i}(0) \leq \mu, \text{for all } i,j \\ \mu \geq 0 \\ \sum_{i \in \mathcal{M}_0} z_{j,i}(0) = 1, \quad \text{for all } j \\ \sum_{j \in \mathcal{N}_0} z_{j,i}(0) \geq 1, \quad \text{for all } i, \text{ if } |\mathcal{N}_0| \geq |\mathcal{M}_0| \\ \sum_{j \in \mathcal{N}_0} z_{j,i}(0) \leq 1, \quad \text{for all } i, \text{ if } |\mathcal{N}_0| < |\mathcal{M}_0| \end{cases}$$

    FirstAssignment $= False$
  end
  While none of $R_i \in \mathcal{M}_0$ are inspected
    If $\frac{\min\{C_k(t), C_k(\tau^s)\} - C_m(t)}{\min\{C_k(t), C_k(\tau^s)\}} \geq p$, switch A_k to A_m
  end
  Update $\mathcal{M}_0$
end

Figure 9.5: Dynamic re-assignment algorithm: Sets $\mathscr{N}_0$ and $\mathscr{M}_0$ of available blue and red agents, respectively. The number of agents in each set is $|\mathscr{N}_0|$ and $|\mathscr{M}_0|$; C_k is the cost of the assignment A_k and is defined by (9.26).

9.6 Results

In this section, we first present an illustrative example with 2 blue and 3 red agents. The example deals with a smaller number of agents and allows us to describe the time progress of the presented scenario. Then we show results of a larger number of numerical simulations with 3 blue and 3 red agents, and another with 3 blue and 6 red agents. The two larger number of numerical simulations are used to illustrate the two cases in Section 9.4: an equal or less number of reds compared to number of blues, and a larger number of reds compared to number of blues.

9.6.1 Illustrative Example

To illustrate our results, we use an example with two UAVs (blue agents) and three ground vehicles (red agents). The velocity of UAVs is $v_B = 0.1$ and the velocity of ground vehicles is $v_R = 0.05$. These and all other problem parameters for the example are provided in Section 9.3, before and after (9.19), defining the computational domain of the *one-on-one* stochastic optimal control solution. Once the numerical solution and the expected time have been computed, we search over the discrete space of the solution to find the minimal non-zero expected time for a UAV to inspect a ground vehicle. This value defines the threshold $\underline{C} = 1.649$sec. Once the threshold is reached, we stop the switching assignments using the rule (9.27). The switching assignment rule parameter in the example is $p = 0.05$ and the rule defines switching among possible assignments.

In this example, we can identify six possible assignments that are labeled as A_i $i = 1,2...6$ and depicted in Fig 9.6. Beyond these assignments, the only other possible assignments are those in which a single ground vehicle is assigned to both UAVs, and we label all of them with the single label A_0. Given the small number of possible assignments, the optimization (9.20) can be performed by the evaluation of each assignment and the selection of the one with the smallest cost. However, the identification of all possible assignments and their enumeration would be a very complex process in a situation with more blue and red agents, and to avoid it we truly benefit from the MILP formulation of the optimal assignment.

For example, in the case of 2 blue and 3 red agents, we define a vector $\mathbf{z}$ as

$$\mathbf{z} = [z_{1,1}\ z_{1,2}\ z_{1,3}\ z_{2,1}\ z_{2,2}\ z_{2,3}\ \mu]^T,\ \mathbf{z} \in \{0,1\}^6 \times \mathbf{R} \tag{9.29}$$

which is composed of 6 binary and one real number components, and introduce the vector $\mathbf{f} = [0\ 0\ 0\ 0\ 0\ 0\ 1]^T$ to formulate the MILP from the previous section as

$$C^o = \min \mathbf{f}^T \mathbf{z} \tag{9.30}$$

$$\mathbf{Az} \leq \mathbf{b} \tag{9.31}$$

$$\mathbf{Bz} = \mathbf{1}_{2\times 1} \tag{9.32}$$

where

$$\mathbf{A} = \begin{bmatrix} V_{1,1} & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & V_{1,2} & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & V_{1,3} & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & V_{2,1} & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & V_{2,2} & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & V_{2,3} & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{9.33}$$

While the presented formulation is specific to the case of 2 blues and 3 reds, the matrices **A**, **B** and vectors on the right side of inequality and equality constraints can be generated and updated automatically in the numerical simulations to address any possible number of blue and red agents. Solving the MILP can be done using any optimization package and we use *intlinprog* from MATLAB 2016b in our simulations, which for the 2 blue and 3 reds problem takes less than a second to compute on an Intel Core i5 processor with 8GB of RAM.

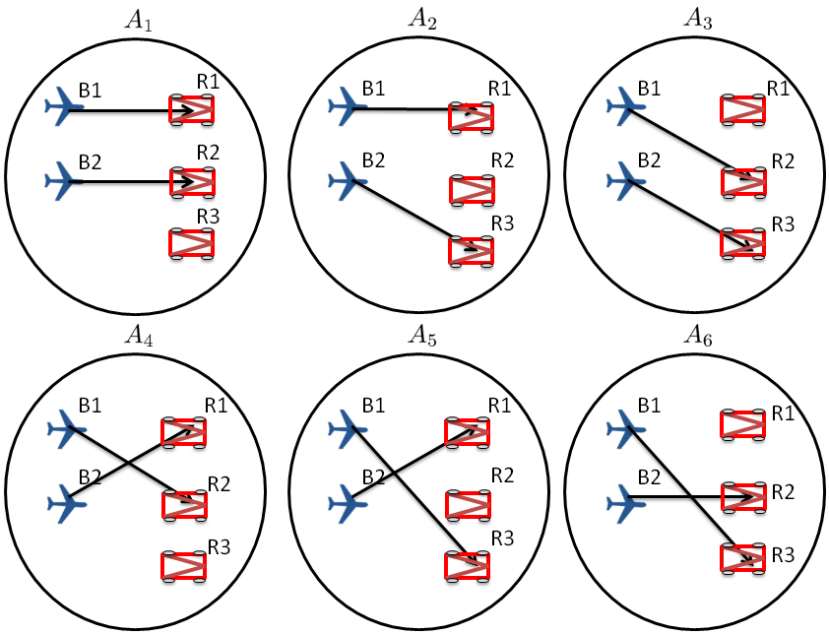


Figure 9.6: Possible assignments: each assignment is depicted by the lines connecting the UAVs ($B1$, $B2$) with the ground vehicles ($R1$, $R2$, $R3$). Beyond these assignments, those in which a single ground vehicle is assigned to both UAVs are labeled by the single label A_0.

Proceeding is an example numerical simulation, which nicely illustrates the approach presented in this paper. Later, we present the statistics of running multiple numerical simulations for 3 blue on 3 red, and 3 blue on 6 red scenarios.

The simulation in Fig.9.7A starts with the two blue UAVs behind the three red ground vehicles, where B_1 is assigned to inspect R_1 and B_2 to inspect R_2, i.e., the current assignment is A_1 as shown in Fig. 9.6. In Fig. 9.7B, B_1 switches its assignment to R_3 and after this, B_2 switches to R_1 (see Fig. 9.7C). This assignment lasts until R_3 has been inspected by B_1 at $t = 14.6$sec (see Fig. 9.7D) and is no longer of interest to B_1 or B_2.

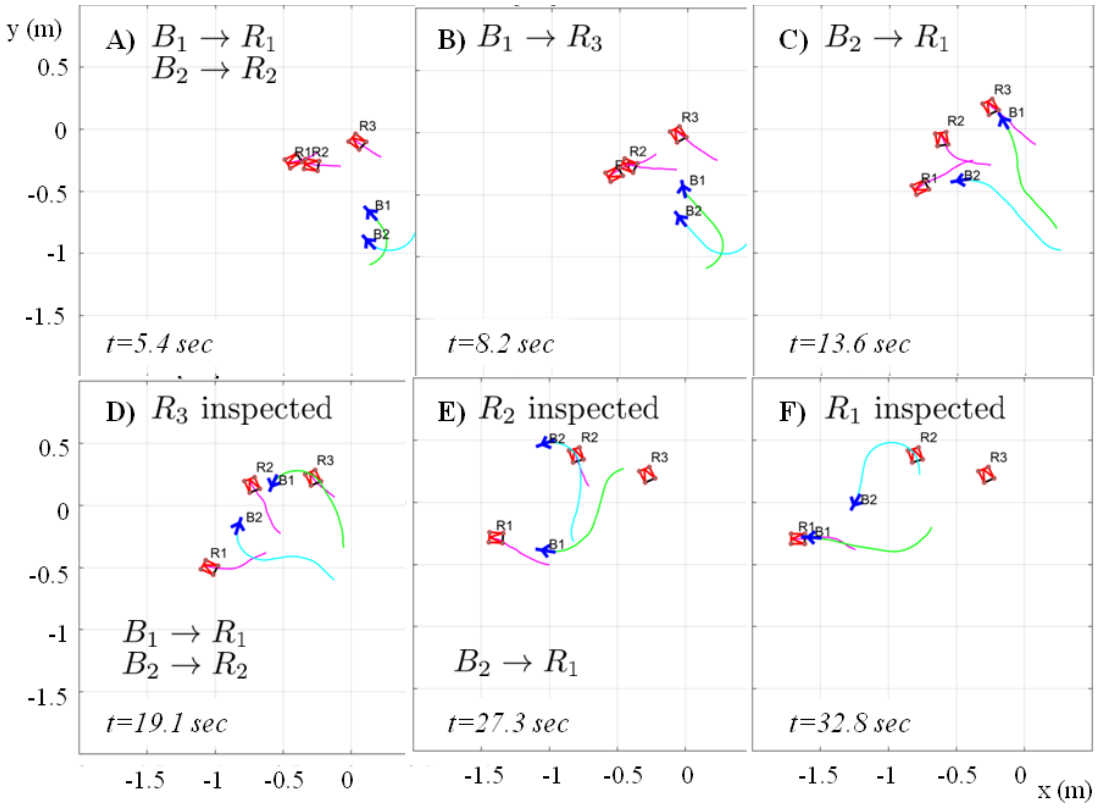


Figure 9.7: Simulation in which the two UAVs switch inspect the three ground vehicles. The progress of time is from A to F. The ground vehicles R_3, R_2, R_1 have been inspected at $t = 14.6$, $t = 23.7$, $t = 32.8$, respectively.

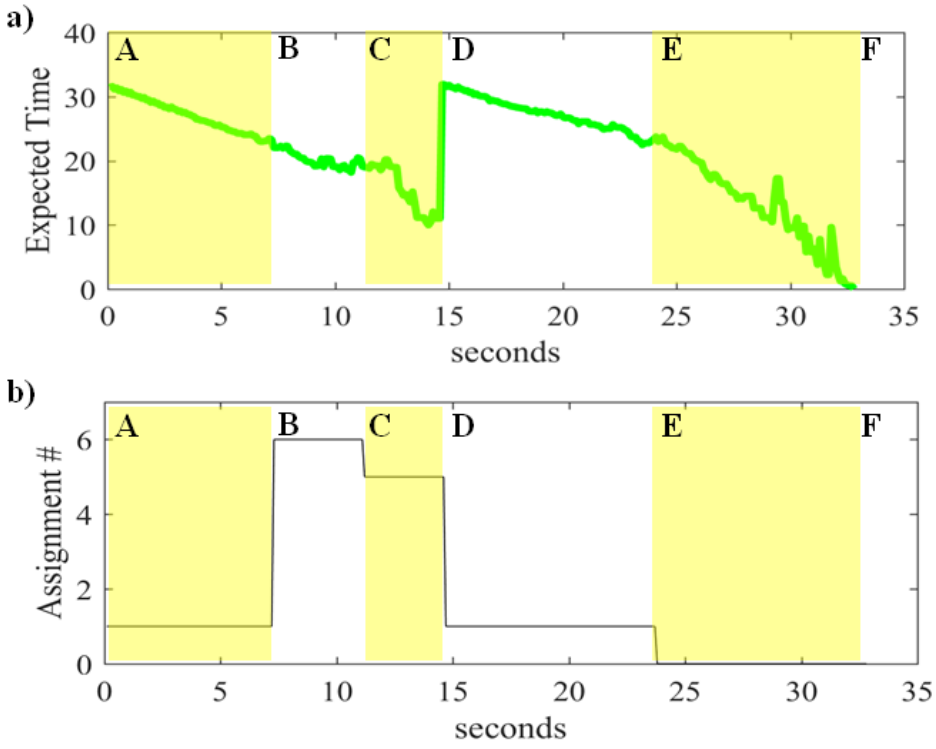


Figure 9.8: a) The cost of the current assignment A_i (see Fig. 4), which is the longest expected time of the assignment; b) the current assignment.

At the beginning of time interval D, a new optimal assignment is computed for B_1 and B_2. Because of that, we can observe a positive jump in the assignment cost, which is the longest expected time of the assignment. The new assignment is that B_1 inspects R_1 with the expected time to inspection of 31.7sec, and B_2 inspects R_2 with the expected time to inspection of 28.1sec; therefore, the cost of the assignment is 31.7sec. This is the optimal assignment since the alternative assignment has the cost of 37.3sec, which results from the expected times of 37.3sec for B_1 inspecting R_2 and 11sec for B_2 inspecting R_1. Following the optimal assignment, R_2 has been inspected by B_2 at $t = 23.7$ (see Fig. 9.7E) without any switchings. Following this, the only assignment is A_0 in which both B_1 and B_2 inspect the remaining ground vehicle R_1. Figure 9.8*a* shows that R_1 has been inspected by B_1 at $t = 32.8$sec at the end of the time interval F.

9.6.2 Numerical Simulation with Three Blue and Three Red Agents

Here we consider an example in which three blue and three red agents are initially positioned as depicted in Fig. 9.10*a)*. We generated 1000 possible stochastic trajectories for each red agent. We used those trajectories to run numerical simulations with three blue agents for 1000 times and two types of re-assignments. The first type of the re-assignments is the one in which the assignment is computed only at the beginning of the simulation, and after the inspection of each red. We call that type of the re-assignment *sequential* since we do not allow for dynamic re-assignments using the

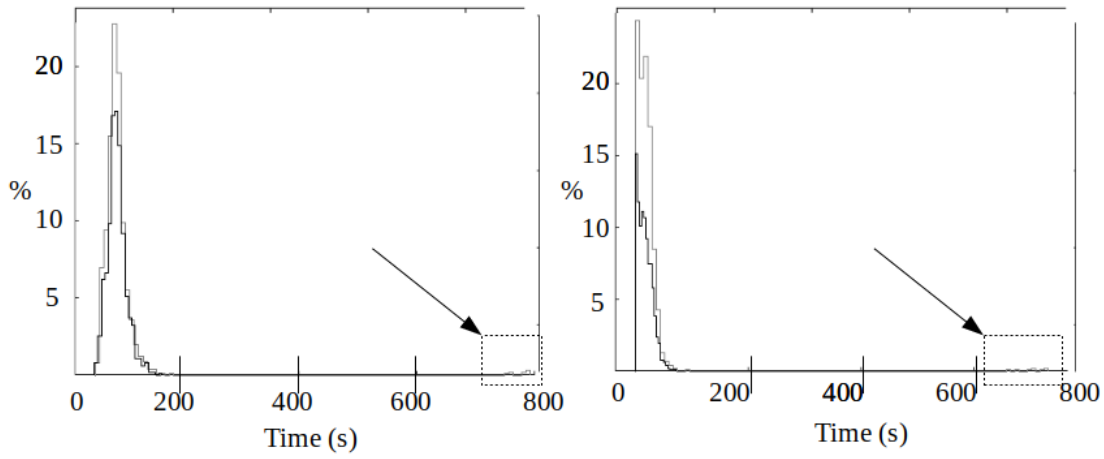


Figure 9.9: Results of 1000 simulations: (left panel) the histograms of the times to the inspection of the last red; (right panel) the histograms of the expected times to the inspection of the last red immediately following the inspection of the 2nd red. The thick line corresponds to the *dynamic* re-assignment algorithm and the thin line to the *sequential* one. The dashed line around 800s is portion of *sequential* runs that took longer than 700s.

rule (9.27) between the inspections of two reds. The second type of the re-assignments is fully *dynamic* re-assignment as described in Fig. 9.5.

From 1000 simulation runs for the *sequential* and *dynamic* re-assignments, we recorded times for the team of blues to inspect all six red agents. The data are summarized in Fig. 9.9 which shows that the distribution of times is very similar for both assignment, except for eight runs for the *sequential* as shown in Fig. 9.9a2. If we ignore those eight runs of the *sequential*, we obtain Fig. 9.9b, which shows the similarly of distribution for both algorithms with average time of 92.5556s (std=16.3641s) for *sequential* and average time of 92.8141s (std=15.9089s) for *dynamic*. However if we don't ignore those eight runs we obtain average time of 98.0303s (std=63.1493s) for

the *sequential*, therefore we can see that the same *dynamic* runs for those *sequential* runs drastically reduced the maximum time to inspect the reds.

9.6.3 Numerical Simulation with Three Blue and Six Red Agents

Here we consider an example in which three blue and six red agents are initially positioned as depicted in Fig. 9.10*b)*. We generated 1000 possible stochastic trajectories for each red agent. As in the three blue and three red agent scenario, we run two re-assignments; the *sequential* and *dynamic*. For an illustration of the simulations, we provide two examples for the *dynamic* re- assignments. (Please follow the links:
https://drive.google.com/open?id=1dKZb5MbYztexNmB-FhVNZgf-Z7VcrcuM
https://drive.google.com/open?id=1R38dYum-lsKJ0dLP3NGAZ7IL7LKQAnte - see Appendix D for snapshots of movies)

From 1000 simulation runs for the *sequential* and *dynamic* re-assignments, we recorded times for the team of blues to inspect all six red agents. The data are summarized in Fig. 9.11(left panel) which shows that the distribution of times is very similar for both assignments. We find that the average time in the case of the *sequential* re-assignment is 102.66s (std=41.76s) and is slightly shorter than the one resulting from the implementation of the *dynamic* re-assignment which is 104.55s (std=37.34s). The similarity could be due to the outcome that in 24% of the runs, the *dynamic* re-assignment algorithm does not have any re-assignment between inspections of two

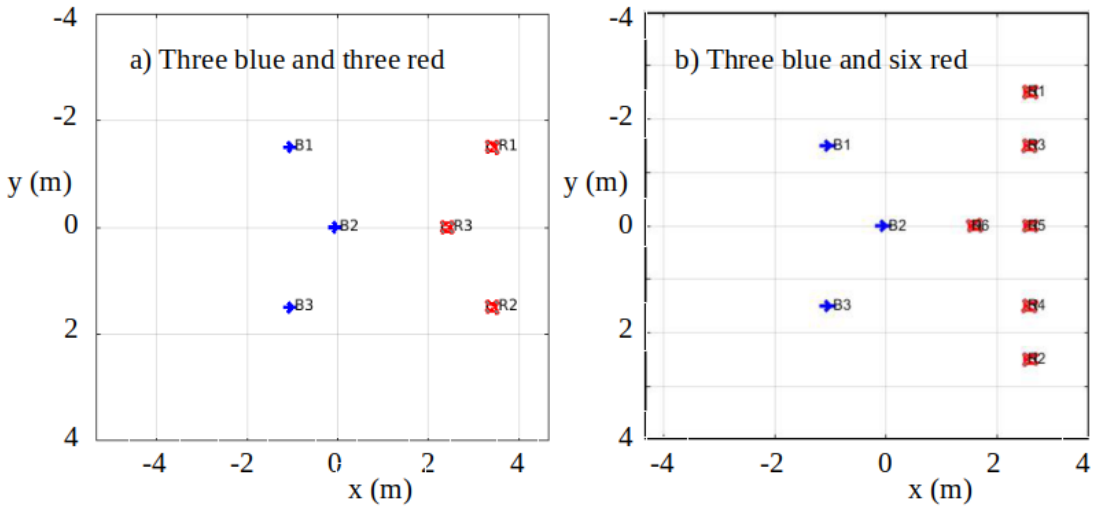


Figure 9.10: The initial configurations of blue and red agents: the blue agents are presented with arrow-like symbols depicting aircraft and the red agents with rectangular symbols depicting ground vehicles, i.e., cars. a) 3 blues and 3 reds. b) 3 blues and 6 reds.

reds, i.e., it behaves exactly as the *sequential* re-assignment algorithm. When we exclude from our analysis those runs, we find that the histograms obtained by the two assignments are still similar and the average time for the *dynamic* re-assignment is 105.26s (std=31.34s). Based on the histograms and the difference of average values being about 10% of the standard deviation of the distributions, we conclude that there is no significant improvement of the *dynamic* re-assignment over the simpler *sequential* re-assignment.

A further analysis of collected data shows that similarly as the *sequential* re-assignment has a slightly shorter time of inspection, the *dynamic* re-assignment has a slightly better positioning of blues after the inspection of each red. To illustrate this, we present

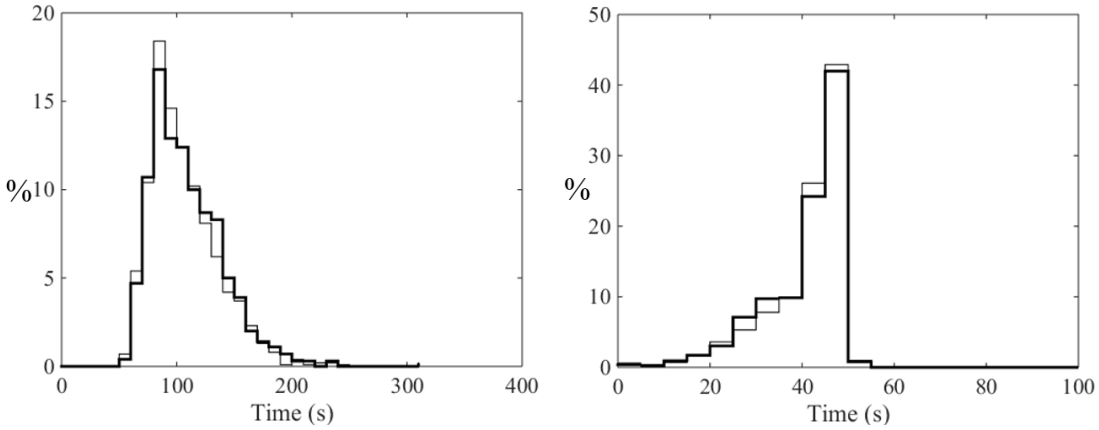


Figure 9.11: Results of 1000 simulations: (left panel) the histograms of the times to the inspection of the last red; (right panel) the histograms of the expected times to the inspection of the last red immediately following the inspection of the 5th red. The thick line corresponds to the *dynamic* re-assignment algorithm and the thin line to the *sequential* one.

in Fig. 9.11(right panel) histograms of the maximum expected time to inspect the last (6th) red after the 5th inspection. The histograms are only based on those 76% simulation runs in which the re-assignment between inspections happened. The average value associated with the histogram for the *sequential* re-assignment is 38.22s (std=8.59s) and the average value associated with the one for the *dynamic* re-assignment is 37.87s (std=8.64s).

9.7 Conclusions

In this paper we presented the control design for N UAVs tasked to perform the time efficient inspection of M ground moving vehicles. The navigation and intent of each ground vehicle are unknown, therefore, the uncertainty of its navigation has to

be anticipated in the navigation of each UAV. The controller for each UAV to inspect each ground vehicle is based on the minimum time stochastic optimal control. This *one-on-one* vehicle optimal control solution is used to compute the expected time of the inspection. We further use that expected time to formulate the assignment problem of deciding what ground vehicle each UAV should inspect. We formulate it as the optimization problem of minimizing the expected time to inspect all ground vehicles. Since the ground vehicles have uncertain trajectories, the optimal assignment may need to be recomputed. However, the recomputing may result in an indefinite sequence of assignment updates without the UAVs ever inspecting all ground vehicles. To address that, we update assignments with the Markov inequality rule. While the rule prevents the possibility of indefinite changes of assignments, it also updates an assignment if it leads to a statistically significant improvement of the expected time of the inspection. The proposed approach was illustrated by the numerical example with two UAVs and three ground vehicles. Furthermore, we ran a large number of numerical simulations for three UAVs and three ground vehicles, and another for three UAVs and six ground vehicles. The results show that if the number of UAVs (N) is greater or equal to the number of ground vehicles (M), then the dynamic algorithm does better than the sequential. This is due to the Markov inequality rule updates that correctly reduce the *absolute* global maximum time of inspection to the last red when $N \geq M$.

Chapter 10

Scalable Navigation for Tracking a Cooperative Unpredictably Moving Target in an Urban Environment

This chapter is a preprint to the paper

- Munishkin, Alexey A., Dejan Milutinović, and David W. Casbeer. "Scalable Navigation for Tracking a Cooperative Unpredictably Moving Target in an Urban Environment." In 2022 control conference on control technology and applications (CCTA)

10.1 Introduction

One common application of small fixed-wing unmanned aerial vehicles (UAVs) [25] is tracking and surveillance. Some specific examples include agriculture, forest fire and plume tracking, traffic monitoring, search and rescue, and convoy protection applications [130, 160, 179].

In this paper, we consider a scenario in which a search and rescue, or convoy protection mission involves a cooperative target which sends out its position to a UAV flying at a constant altitude, see Fig. 10.1. The UAV is tasked to orbit around the target at a distance and keep the line of sight (LOS) towards the target, so that the target and its surrounding area can be observed by a UAV mounted camera. The cooperative target is in an urban environment with tall buildings and we assume that it can move freely, i.e., it *does not have a pre-planned trajectory* which is communicated to the UAV. This uncertainty is an important part of the scenario. It implies that the cooperative target, either a human or a vehicle, is free to move around, worrying only about its own mission. The UAV must then be able to handle trajectory anomalies that occur with the cooperative target.

Dealing with the unpredictable motion of the target naturally leads to a feedback control-based navigation for target tracking. Examples of various approaches that have been used for the design of a feedback control-based navigation strategy are model-predictive control [86], non-linear Lyapunov function based control [32], Lyapunov

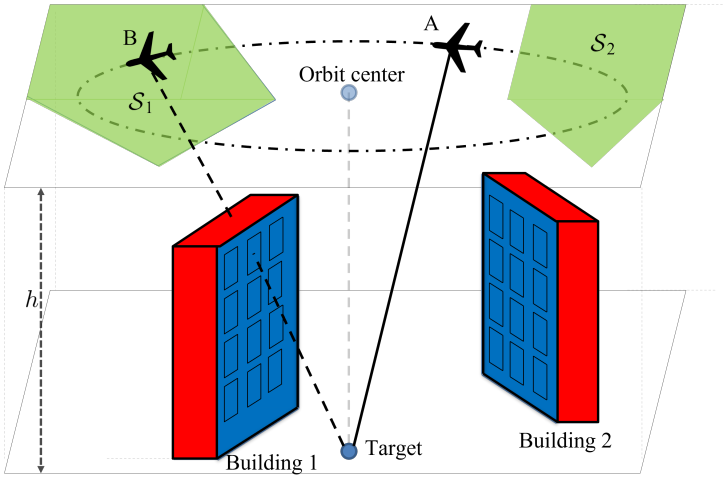


Figure 10.1: The UAV orbits around the target and above the buildings at a fixed altitude h. The parts of the operating regions (green), i.e., the shadows in which the target is occluded by Building 1 and Building 2, are denoted by $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively. From position A, the UAV has the line of sight towards the target, which is depicted by the solid line. From position B, the UAV does not have the line of sight, which is depicted by the dashed line.

vector fields [101], stochastic control [121] and game theory [142]. The navigation problem is significantly more complex when the UAV has to avoid certain parts of the operating space to maintain the GPS signal [187], or prefers certain relative positions to the target [198]. To address that complexity in [27], the authors used a deep reinforcement learning method for target tracking and avoiding static areas of the operating space. To account for no-fly zones and operating regions in which the target is occluded from the UAV, the authors in [173] proposed to use a control based on predictions of future target positions. The use of predictions has also been a topic for target tracking in an urban environment in [196], which includes an excellent survey on various approaches to target tracking.

In this work, we are particularly interested in having the UAV's target tracking take into account the realistic shapes that arise in the operating region as the target is occluded from the UAV by buildings. We refer to these operating region parts as *shadows* (see Fig. 10.1). A similar type of shadows has been considered for search [65] and path planning in [158], [128], as well as for tracking [159], in which the authors used a known target trajectory and a genetic algorithm. It should be noted that as the target moves, it also changes its relative position to surrounding buildings. Consequently, the shadows created by the buildings (see Fig. 10.1) are not static, but change with the motion of the target. Because of that, in this paper we propose a feedback control policy for orbiting above the freely moving target and accounting for the moving shadows.

Since the building shadows move, can overlap and collectively result in complex shapes, we describe here shadow moving boundaries by arrays of obstacles. In the light of this, the UAV must orbit the freely moving target while avoiding a large number (array) of obstacles. As the target moves, the obstacles, as well as their number can vary, therefore, we propose a scalable navigation strategy which is based on stochastic control. Our approach is to solve stochastic optimal control for keeping a randomly moving target at a constant distance and avoiding a single randomly moving obstacle. We solve this single-target single-obstacle control problem using numerical stochastic optimal control [94] from which we obtain the optimal control, as well as the corre-

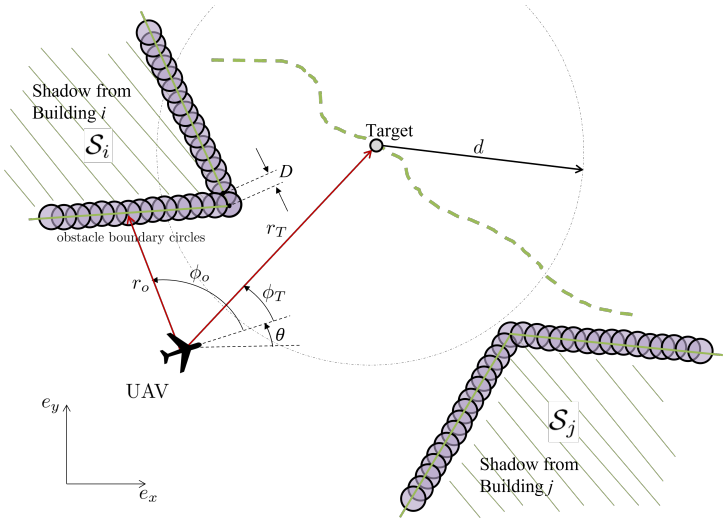


Figure 10.2: A top down view of the UAV navigation while avoiding occlusions: the shadow boundaries are defined using an array of circular obstacles of the radius D (gray circles). These are depicted for two shadows, $\mathcal{S}_i$ and $\mathcal{S}_j$. The desired circumnavigation distance is d, the distance and the bearing angle to the target are r_T and ϕ_T, respectively, and the distance and the bearing angle to an obstacle are r_o and ϕ_o, respectively.

sponding value function. Then, we use the computed value function and evaluate it with respect to each obstacle to obtain the scalable navigation strategy. This approach can be thought of as a form of approximate dynamic programming [137], and a variant of it has been successfully applied in a UAV intercept problem [124].

Contribution: To the best of the authors' knowledge, this is the first paper that develops a scalable fixed-wing UAV feedback control strategy for orbiting a freely moving target at a distance while avoiding moving shadows, created by buildings, from which the target is not visible.

The paper is organized as follows. In section 10.2, we formulate an optimal control problem associated with the scenario and explain the reason for proposing the scalable

approach to the solution. This is followed by Section 10.3 on the computation of the shadow describing obstacles and Section 10.4 on single-target single-obstacle feedback control, which navigates the UAV to orbit around a single target in the presence of a single obstacle in the operating space. Section 10.5 gives the proposed scalable strategy based on the feedback control and value functions computed for the single-target single-obstacle feedback control. Section 10.6 includes illustrative numerical simulation results for the scalable strategy and shows its performance, and Section 10.7 presents concluding remarks.

10.2 Problem Formulation

We consider a UAV that flies at a constant altitude $h > 0$ and at a fixed speed $v > 0$. We model it as a Dubins vehicle kinematic model

$$dx = v\cos\theta dt \tag{10.1}$$

$$dy = v\sin\theta dt \tag{10.2}$$

$$d\theta = udt \tag{10.3}$$

where x and y are the coordinates of the UAV position and θ is the heading angle. The UAV control variable is the bounded turning rate $u \in [-u_{max}, u_{max}]$, $u_{max} > 0$. Similarly as in [11], the UAV is tasked with maintaining a desired distance $d > 0$ from the target while the target motion is unpredictable, therefore, we model it as a 2D Brownian

motion, i.e.,

$$dx_T = \sigma_T dw_{x_T} \tag{10.4}$$

$$dy_T = \sigma_T dw_{y_T} \tag{10.5}$$

where (x_T, y_T) is the target position, and the variables dw_{x_T} and dw_{y_T} denote the unit intensity Wiener process increments along the x and y coordinate directions. The Wiener process scaling parameter is $\sigma_T > 0$.

In our problem formulation, the UAV has to avoid building shadows. They move as the result of the target movement and they can also overlap with each other. In summary, the shadow regions move and change their shape, which can be geometrically complex. To address this dynamic nature of the shadows, we propose that the shadow region boundaries are described by arrays of circular moving obstacles, see Fig. 10.2. The obstacles of radius D are positioned at (x_i, y_i), $i = 1, 2, ...$ along the boundary lines of the shadows. Since the UAV has no knowledge of the shadow boundary motions, the kinematics of each obstacle is modeled as a 2D Brownian motion

$$dx_i = \sigma_o dw_{x_i} \tag{10.6}$$

$$dy_i = \sigma_o dw_{y_i} \tag{10.7}$$

where (x_i, y_i) is the i-th obstacle position, dw_{x_i} and dw_{y_i} denote the Wiener process increments along the x- and y coordinate directions, and $\sigma_o > 0$ is a constant scaling parameter.

The relative position between the UAV and the target can be described by the distance r_T and the bearing angle ϕ_T, see Fig. 10.2, which are

$$r_T = \sqrt{(x_T - x)^2 + (y_T - y)^2} \tag{10.8}$$

$$\phi_T = \arctan\left(\frac{y_T - y}{x_T - x}\right) - \theta. \tag{10.9}$$

Similarly, the relative position between the UAV and the i-th obstacle can be described by the distance r_i and the bearing angle ϕ_i, which are

$$r_i = \sqrt{(x_i - x)^2 + (y_i - y)^2} \tag{10.10}$$

$$\phi_i = \arctan\left(\frac{y_i - y}{x_i - x}\right) - \theta. \tag{10.11}$$

Therefore, we formulate our tracking problem as a stochastic optimal control problem of finding the feedback controlled turning rate u that minimizes the cost

$$J(\tilde{x}, u) = E\left\{g_N(\tilde{x}) + \int_0^{\tau} e^{-\beta t}\,(r_T - d)^2\,dt\right\}, \tag{10.12}$$

where d is the desired distance to the target, τ is a terminal time at which the UAV enters a shadow described by the array of obstacles at relative positions (r_i, ϕ_i), i.e., there is i such that $r_i < D$, and $g_N(\tilde{x})$ is the terminal cost for when that happens, which is

$$g_N(\tilde{x}) = M \tag{10.13}$$

In this expression, $M \gg 0$ is a high penalty for the UAV entering a shadow and $\tilde{x} = (r_T, \phi_T, r_1, \phi_1, ..., r_N, \phi_N)$ is a $2N+2$ vector which stores all relative positions for one target and N obstacles.

The tracking problem formulated by (10.12) is a complex problem. The first part of that complexity stems from a potentially large number of obstacles N, i.e., the large state space to deal with. This can be a very difficult problem to solve, but in principle, we can write the Hamilton-Jacobi-Bellman (HJB) equation defining the solution. The second and even bigger part of the problem complexity is that as shadows dynamically change their shape, they overlap each other, or separate from each other, which means that the solution has to account for a variable number of obstacles N. In other words, we cannot count on the fact that the vector of relative positions $\tilde{x}$, i.e., the state vector for our problem is of a fixed constant length. Therefore, we propose a scalable solution to the problem.

The proposed scalable solution to the problem is based on the stochastic optimal control solution for orbiting the target while avoiding a single randomly moving obstacle. Under the assumption that the UAV is outside of any shadow, we use the single-target single-obstacle solution to synthesize the scalable solution to avoid all shadow defining obstacles. Finally, to account for a possibility that the UAV starts its trajectory within a shadow, or that it happens to enter a shadow, we define a rule for the UAV to escape the shadow.

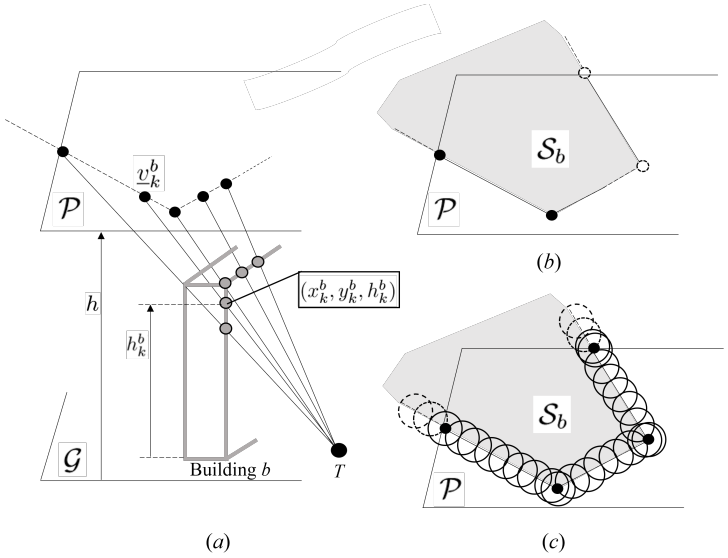


Figure 10.3: The process of computing shadow defining obstacles: (a) computing points $\underline{v}_k^b$ in the plane $\mathscr{P}$ at the height h which are on the lines going through the T target and building b edge points; (b) computing the convex hull $\mathscr{S}_b$ of all points $\underline{v}_k^b$ in the plane $\mathscr{P}$; (c) the shadow defining obstacles are depicted with circles centered at the vertices of the convex hull and equidistant points along the convex hull boundaries.

10.3 Shadow Describing Obstacles

Below we describe computations of shadow describing obstacles. The whole process of computing the shadows follows from the fact that buildings are convex objects and that the perspective projection preserves convexity [31]. In practice, the same algorithm can be used if a 3D map of an urban area, or its estimation, is provided.

Figure 10.3 illustrates the process of computing shadow describing obstacles for a single building b. The target T is located on the ground $\mathscr{G}$ at the position (x_T, y_T). The plane $\mathscr{P}$ for which we compute the shadow of the building b is at the height h above $\mathscr{G}$. The building is described by a sequence of 3D coordinates $[x_k^b, y_k^b, h_k^b]^T$ sampled along the building edges and at its corners, where "^{T}" is the vector transpose and $k = 1, 2, ...$ is a unique index identifying the point, see Fig. 10.3*a*. The vector connecting the target $[x_T, y_T, 0]^T$ and $[x_k^b, y_k^b, h_k^b]^T$ points is

$$\underline{s}_k = [x_k^b, y_k^b, h_k^b]^T - [x_T, y_T, 0]^T, \tag{10.14}$$

therefore, the line of vector $\underline{s}_k$ intersects the plane $\mathscr{P}$ at the point

$$\underline{v}_k = \frac{h}{h_k^b}\underline{s}_k + [x_T, y_T, 0]^T. \tag{10.15}$$

Since any building point that is close to the ground would have a corresponding $\underline{v}_k$ that is far away, we use the building points that are above a certain height. Therefore, $h_k^b > h_{min} > 0$ for all k. Once we compute $\underline{v}_k$ for all k, we compute the convex hull of the points $\mathscr{S}_k$, i.e., the shadows. These computations return the vertices of $\mathscr{S}_b$ as

depicted in Fig. 10.3*b*. Finally, we generate obstacles at the vertices of $\mathscr{S}_b$, as well as at the equidistant points (separated by the diameter of the obstacles) along the edges of $\mathscr{S}_b$ up to a certain large distance from the operating region as depicted in Fig. 10.3*c*.

Finally, after we compute all shadows $\mathscr{S}_b$, $b = 1, 2, ...$ for multiple buildings and corresponding obstacles defining each of them, we exclude any obstacle if the position of the obstacle happens to be in the interior of any shadow, or too far from the operating region. With this, we can deal with the fact that shadows overlap each other or separate from each other. Consequently, as expected the number of shadow describing obstacles changes in time. This is illustrated in Fig. 10.4 in which the position of the target T and shadows in Fig. 10.4*b* corresponds to the position in Fig. 10.4*a*. As the target continuously changes its position, the shadows continuously change as well. Figure 10.4*c* shows the shadows for the target position which is closer to the center of the environment.

10.4 Feedback Control for Tracking a Single Target while Avoiding a Single Obstacle

In this section, we consider the stochastic target that moves as described in (10.4)-(10.5) and a single stochastic obstacle that moves as described in (10.6)- (10.7). In contrast to the problem described in Section II, the problem here is tractable because it

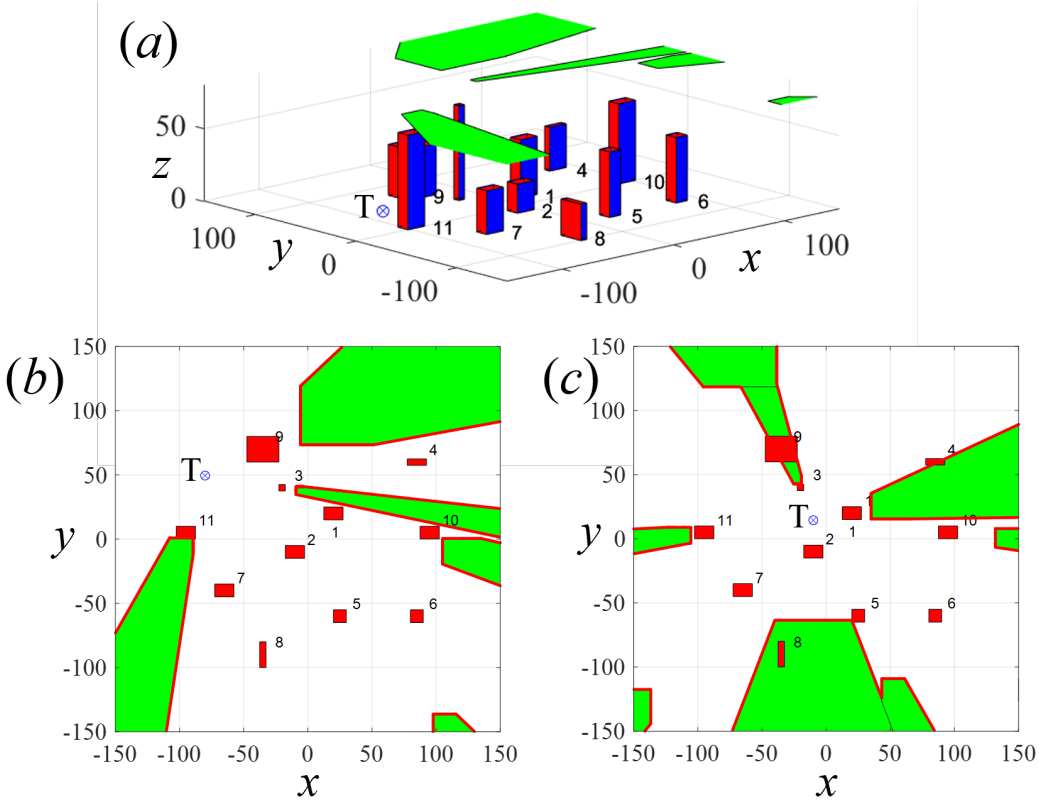


Figure 10.4: An example of 11 buildings with their shadows and a target that moves: (a) a 3D figure showing the target T (crossed circle) at the initial position and 11 buildings of various sizes. The shadows are computed for the operating space at height 80 m. (b) a 2D projection of the target T, buildings (red rectangles) and shadows (green) all associated with the initial position from the panel (a). (c) a 2D projection of the target T, buildings and shadows after the target moves to another position.

only deals with one obstacle. In Section V, we show how to develop a scalable solution to the problem using the single target controller developed here. Since we deal with a single obstacle, we will use $i = 0$, i.e, the position of the obstacle is given by (x_0, y_0), and the distance and the bearing angle from the UAV to the obstacle are r_0 and ϕ_0, respectively.

For the feedback control u that tracks the target at a constant distance d, we solve the stochastic optimal control problem

$$J_0(\tilde{x}_0, u) = E\left\{ g(\tilde{x}(\tau)) + \int_0^{\tau} e^{-\beta t} (r_T - d)^2 \, dt \right\}, \tag{10.16}$$

where $\tilde{x} = (r_T, \phi_T, r_0, \phi_0)$ and $\tilde{x}_0$ is initial state, τ is the terminal time at which $r_0 < D$ and the terminal cost g is

$$g(\tilde{x}(\tau)) = M \tag{10.17}$$

where $M \gg 0$ is a high penalty cost of the UAV getting in the proximity of obstacles.

To solve the problem, we use (10.3-(10.5), (10.7) with $i = 0$, (10.8)-(10.11), and Itô

calculus to derive the stochastic differential equations for r_T, ϕ_T, r_0, ϕ_0, which are

$$\begin{aligned} dr_T &= \left(-v\cos(\phi_T) + \tfrac{\sigma_T^2}{2r_T}\right)dt + \sigma_T dw_{r_T} \qquad (10.18)\\ &= b_{r_T}dt + n_{r_T}dw_{r_T}, \\ d\phi_T &= \left(\tfrac{v}{r_T}\sin(\phi_T) - u\right)dt + \tfrac{\sigma_T}{r_T}dw_{\phi_T} \qquad (10.19)\\ &= b_{\phi_T}dt + n_{\phi_T}dw_{\phi_T}, \\ dr_0 &= \left(-v\cos(\phi_0) + \tfrac{\sigma_o^2}{2r_0}\right)dt + \sigma_o dw_{r_0} \qquad (10.20)\\ &= b_{r_o}dt + n_{r_0}dw_{r_o}, \\ d\phi_0 &= \left(\tfrac{v}{r_0}\sin(\phi_o) - u\right)dt + \tfrac{\sigma_o}{r_0}dw_{\phi_0} \qquad (10.21)\\ &= b_{\phi_0}dt + n_{\phi_0}dw_{\phi_o}. \end{aligned}$$

Then, we formulate the solution using the HJB equation

$$0 = \min_u \{\mathscr{L}_u V(\tilde{x}) - \beta V(\tilde{x}) + (r-d)^2\}, \qquad (10.22)$$

where $\mathscr{L}_u$ is the differential operator

$$\begin{aligned} \mathscr{L}_u = {} & b_{r_o}\frac{\partial}{\partial r_0} + \frac{n_{r_o}^2}{2}\frac{\partial^2}{\partial r_0^2} + b_{\phi_o}\frac{\partial}{\partial \phi_0} + \frac{n_{\phi_o}^2}{2}\frac{\partial^2}{\partial \phi_0^2} \\ & + b_{r_T}\frac{\partial}{\partial r_T} + \frac{n_{r_T}^2}{2}\frac{\partial^2}{\partial r_T^2} + b_{\phi_T}\frac{\partial}{\partial \phi_T} + \frac{n_{\phi_T}^2}{2}\frac{\partial^2}{\partial \phi_0^2} \end{aligned} \qquad (10.23)$$

with the terms b_{r_o}, n_{r_o}, b_{ϕ_o}, n_{ϕ_o} and b_{r_T}, n_{r_T}, b_{ϕ_T}, n_{ϕ_T} defined as in (10.18)-(10.21). The HJB satisfies the boundary condition $V(\tilde{x}_0) = M$ for all $\tilde{x}_0$ with the component $r_0 \leq D$, which is due to the terminal cost $g(\tilde{x}(\tau))$.

The solution of (10.22) can be computed using a locally consistent Markov chain discretization of the HJB equation [94]. The discretization allows us to compute an approximate solution of the HJB equation using

$$V^h(\tilde{x}_0^h) = \min_u \left\{ \Delta t_{\tilde{x}_0^h,u} e^{-\beta \Delta t_{\tilde{x}_0^h,u}} \left(r_T^h - d \right)^2 + \right.$$
$$\left. \sum_{\text{for all } \tilde{x}_0 + \Delta\tilde{x}_0 \in \mathscr{N}_{\tilde{x}_0^h}} p^{\pm}_{\Delta\tilde{x}_0,u} \cdot V^h(\tilde{x}_0 \pm \Delta\tilde{x}_0^h) \right\} \tag{10.24}$$

where V^h denotes the discretized version of V, $\tilde{x}_0^h$ coordinates of the discretized state space, $\Delta\tilde{x}_0^h$ are discrete increments along one of the four coordinate directions, i.e., $\Delta\tilde{x}_0^h \in \{(\Delta r_T,0,0,0),(0,\Delta\phi_T,0,0),\ (0,0,\Delta r_0,0),\ (0,0,0,\Delta\phi_0)\}$, and $\mathscr{N}_{\tilde{x}_0^h}$ is the set of the discretized coordinates in the neighborhood of $\tilde{x}_0^h$. We compute the solution of (10.24) using the value iterations and the discretization [94] defined transition probabilities $p^{\pm}_{\Delta\tilde{x}_0,u}$ and interpolation times $\Delta t_{\tilde{x}_0^h,u}$.

Due to the limited space available, we skip the details on computing these values and point the reader to [94], as well as [39] for illustrative examples. The same discretization has been previously used to compute the controllers in [12, 124] and for the controller in [121], which was flight tested with a small UAV.

10.5 Tracking the Brownian Moving Target and Avoiding Shadows

Now we tackle the case in which there are multiple shadows $\mathscr{S}_j$, $j = 1,..N_B$ and corresponding N shadow defining obstacles indexed by i, $i = 1,...N$. The relative position of the UAV with respect to each obstacle and the target is $\tilde{x}_i = (r_T, \phi_T, r_i, \phi_i)$. In the previous section, we explained the process of obtaining the optimal control for the navigation of the UAV in the presence of a single obstacle and here we use that solution for the navigation in the case of multiple obstacles.

Let us define an array $\underline{V}^h$ composed of the computed value function V^h (10.24) evaluated for $\tilde{x}_i$, $i = 1,2,..N$, i.e.,

$$\underline{V}^h = \{V^h(\tilde{x}_1), ...V^h(\tilde{x}_N)\}. \tag{10.25}$$

Under the assumption that the UAV is not in a shadow and that it has to avoid all shadow defining obstacles, we use the following UAV control

$$u = u^h(\tilde{x}_i),\ i = \underset{i}{\text{argmax}}\,\underline{V}^h = \underset{i}{\text{argmax}}\{V^h(\tilde{x}_i)\}. \tag{10.26}$$

In words, we use the optimal control u for the obstacle i, which corresponds to the obstacle with the largest value $V^h(\tilde{x}_i)$. Since the optimal control is based on the minimization of the value function, this control implements $\min_u \max_i \underline{V}^h$, where the operator "max" is over the index i and "min" is inherited from the optimal control computations for u^h,

and accounts for all possible control actions. The advantage of the control in (10.26) is that it can be applied for a varying number of obstacles which can also be large. Therefore, the proposed control (10.26) is scalable.

In the computations of the optimal control u^h, we use the penalty M to obtain the optimal control providing that $r_i > D$, i.e., that the UAV avoids shadow defining obstacles. We can also check if UAV loses the line of sight to the target by checking if the UAV is within a convex hull of a shadow. Therefore, to make our control strategy complete, we use the following control

$$u = \begin{cases} u^h(\tilde{x}_i), & \max_i\{V^h(\tilde{x}_i)\} < M \\ & i = \text{argmax}_i\{V^h(\tilde{x}_i)\} \\ u_{esc}, & \max_i\{V^h(\tilde{x}_i)\} = M \text{ or} \\ & (x,y) \in \cup_j \mathscr{S}_j \end{cases} \tag{10.27}$$

where $\mathscr{S}_j$ is the set of points that belongs to a shadow resulting from a building j, $j = 1,2,...N_B$, and the control u_{esc} is the control to escape the shadow

$$u_{esc} = \begin{cases} u_{max}, & \phi_T \in (0,\pi) \\ -u_{max}, & \phi_T \in [-\pi,0) \\ 0, & \phi_T = 0 \end{cases} \tag{10.28}$$

The control u_{esc} is based on the fact that the shadow cannot appear at the position of the target, i.e., when the UAV is directly above the target. For this reason, u_{esc} is the so-called pure-pursuit controller which navigates the UAV with the heading pointing

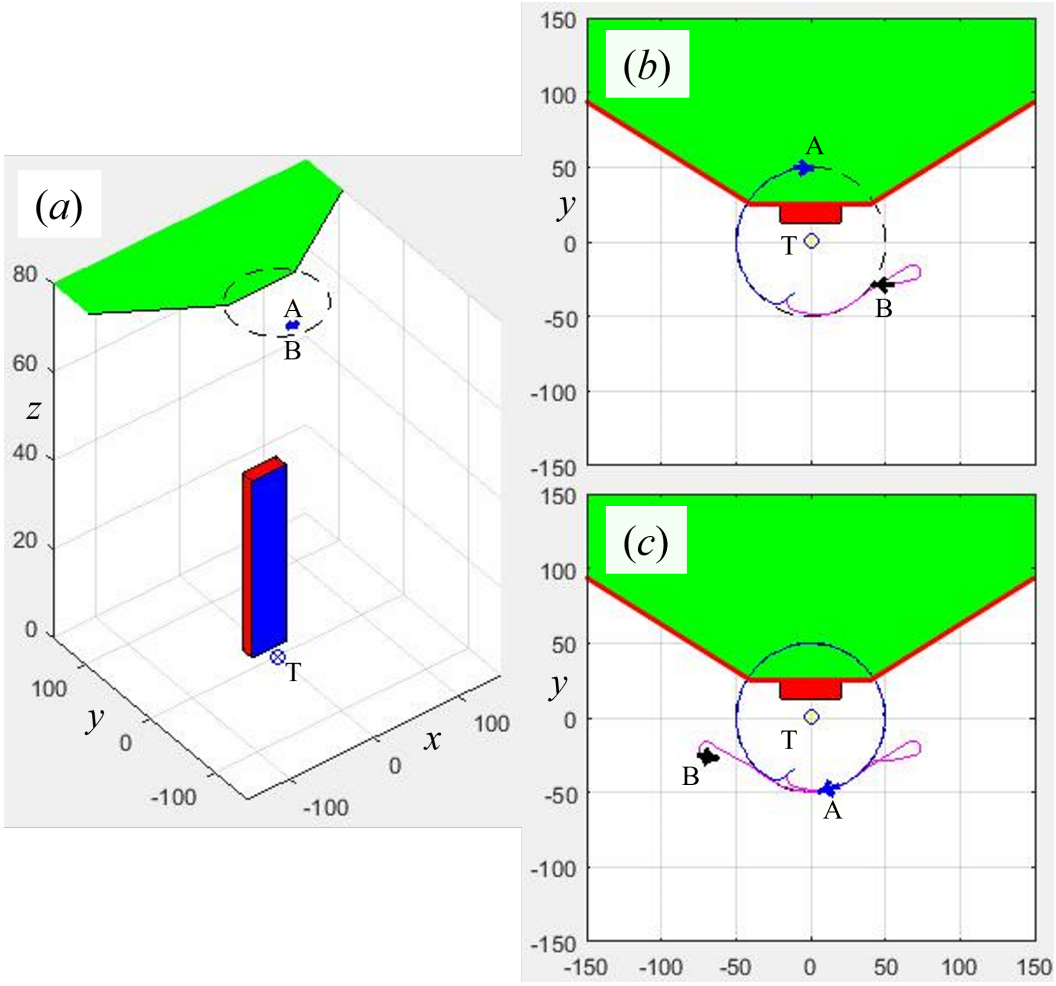


Figure 10.5: Navigation in the case of a stationary target T, single building, UAV A which orbits around the target and does not avoid the shadow and UAV B which does: (a) a 3D view of the environment with both UAVs at the same initial position; (b) from the initial position, UAV A goes clockwise and enters the shadow, while UAV B goes counter-clockwise and then turns to avoid the shadow; (c) UAV B makes the second turn to avoid the shadow while UAV A continues on its orbit.

towards the target. Once the UAV is out of the shadows and distant from any of the shadow defining obstacles, i.e., $\max_i\{V^h(\tilde{x}_i)\} < M$, the control resumes to be $u^h(\tilde{x}_i)$, $i = \text{argmax}_i\{V^h(\tilde{x}_i)\}$.

10.6 Results

The numerical simulation results of this section use the kinematic model (10.1)-(10.3) with a UAV velocity of $v = 10$ m/s. The UAV has a maximum turning rate of $u_{max} = 0.3$ rad/s. The sample time used in the simulations is $\Delta t = 0.1$ seconds.

The computational domain for computing the discrete approximation of the 4-dimensional HJB solution is bounded with $r_T^{min} = 10$, $r_T^{max} = 90$, and $r_o^{min} = 0.05$, $r_0^{max} = 2.04$, with respect to r_T and r_o variables, respectively. The discretization steps for r_T and r_o variables are $\Delta r_T = (r_T^{max} - r_T^{min})/199$ and $\Delta r_o = (r_o^{max} - r_o^{min})/99$, respectively. The other two variables ϕ_T and ϕ_o are both within the range $(-\pi, \pi]$ and are discretized with $\Delta\phi_T = \pi/180$ and $\Delta\phi^o = 0.05$ steps.

In the computation of the HJB solution, we use $\sigma_T = 0.05$ and $\sigma_o = 0.039$ to describe unpredictable motions of the target and a single shadow defining obstacle, respectively, where the diameter of the obstacle is set to $D = 0.1m$. For the boundary condition that accounts for $r_o \leq D$, we use the penalty $M = 10^4$.

Figure 10.5 shows an example with a single building and stationary target. The figure depicts trajectories of two UAVs, A and B. They both start from the same initial position as depicted in Fig. 10.5a. UAV A implements the circumnavigation controller from [6], which does not avoid shadows, while UAV B implements the circumnavigation controller (10.27), which avoids them. With this example, we show the basic functionality of the proposed navigation (10.27). Figure 10.6 shows the result of the

controller (10.27) for a moving target along the trajectory depicted by the yellow line. The environment has 8 buildings and it the same one presented in Fig. 10.4 from which we removed 3 buildings, which slightly reduces the number of shadows. The figure shows the current shadows and the UAV position at the times indicted in the panels (a)-(d), as well as the trajectory traversed by the UAV up to that time. While the figure illustrates the complexity of the UAV motion, it does not completely illustrate the efficiency of the navigation controller (10.27) to avoid shadows.

In order to get a grasp of how well the proposed controller (10.27) works, we compare it again with the circumnavigation controller from [6]. The performance is measured as a percentage of time that the UAV spends in the shadows in the environment from Fig. 10.4. For a comparison with N_B buildings $1 \leq N_B \leq 8$, we performed 50 simulation runs in which we randomly removed $11 - N_B$ buildings, randomly set the initial position of the UAV in the operating space and used the fixed target trajectory depicted in Fig. 10.6. While we could change all parameters of the problem, we reasoned that fixing the target trajectory made sense since the target motion in an urban environment is usually constrained to a street layout. The percentages of time recorded from our simulation runs are plotted in Fig. 10.7. The diagrams show that for the UAV that circumnavigates around the target without avoiding shadows, the percentage of time in the shadows increases with the number of buildings. When the UAV uses the controller (10.27), in the case of 1 and 2 buildings, there is a slight performance improvement

relative to the other controller. However, as the number of building increases, the performance difference increases as well in favor of the controller from (10.27). Overall, across all runs, one standard deviation band for the performance of the proposed con-

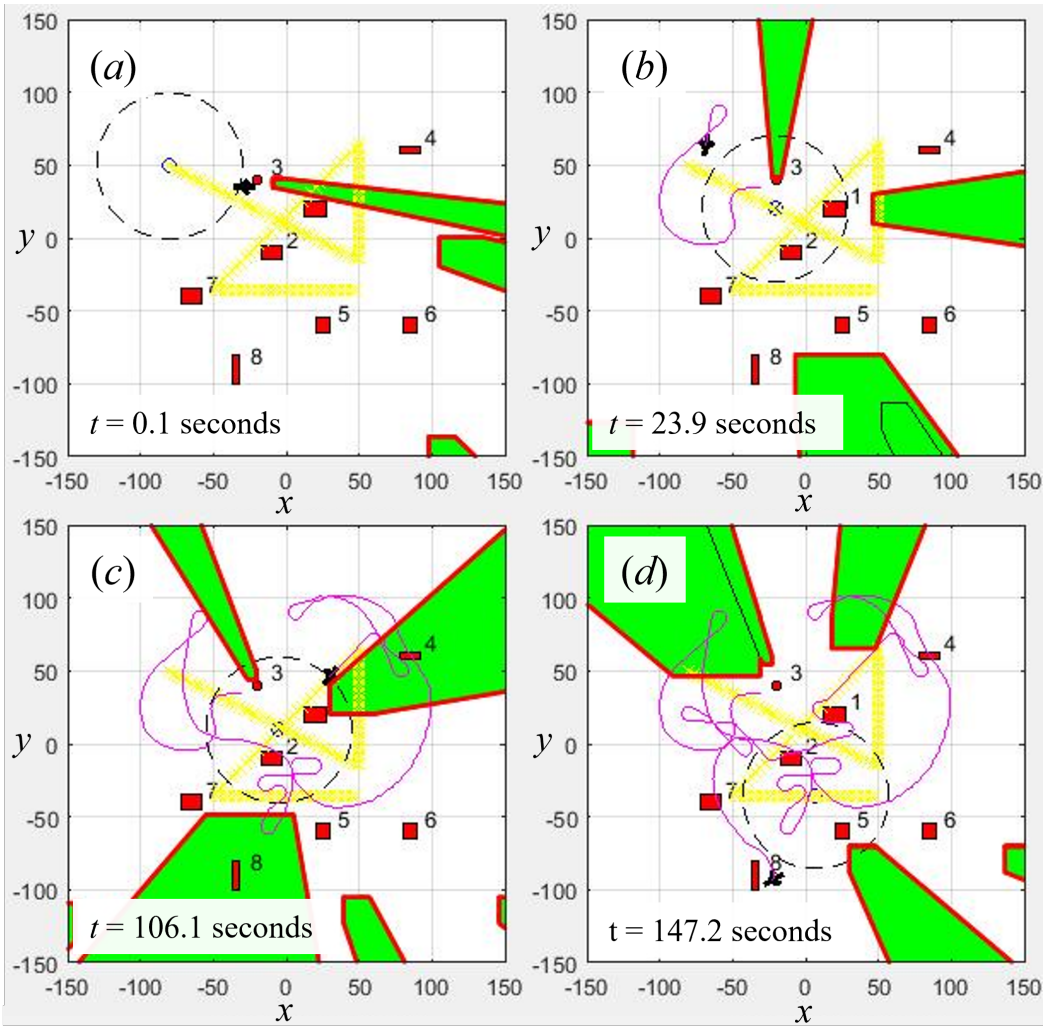


Figure 10.6: Environment with 8 buildings (red rectangles): the target is denoted by a circle and moves along the path (yellow line). The UAV orbits around the target and avoids moving shadows (green). The trajectory traversed by the UAV is depicted in red. Panels (a)-(d) are snapshots of the numerical simulation at $t = 0.1sec, 23.9sec, 106.1sec, 147.2sec$, respectively.

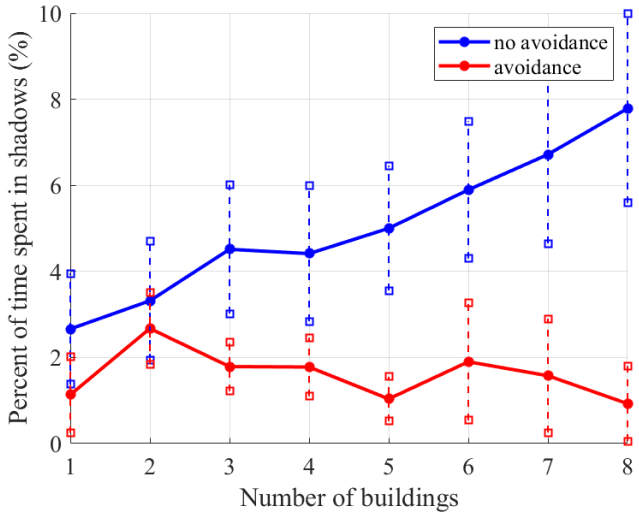


Figure 10.7: Percent of time that the UAV spent in shadows from 50 simulation runs: in each run, the initial position and selection of buildings from the Fig. 10.4 environment are random. The plot for the controller that does not avoid shadows is in blue and the one for the controller from (10.27) that avoids shadows is in red. The standard deviations are depicted as vertical dashed lines.

troller never exceeds 4%.

10.7 Conclusions

In this paper, we presented the controller for a UAV to orbit above a cooperative target and avoid positions at which it loses the line of sight towards the target. The target is cooperative because it constantly relays its location to the UAV. However, since the target moves in an urban environment with buildings of various sizes, the line of sight can be easily lost since building-induced shadows move as well. Both the motions of the target and shadows are considered unpredictable and, furthermore, the shadows

have complex shapes.

For the UAV to be able to orbit above the target and avoid shadows, we proposed the stochastic control approach. We described the shadows as an array of randomly moving obstacles and proposed the scalable strategy for their avoidance while orbiting above the target. The strategy is based on the solution of the stochastic optimal control problem of orbiting around the target while avoiding a single random obstacle. As illustrated in our numerical simulations, the proposed control strategy is shown to be successful in orbiting above the target and avoiding shadows.

For future work, it would be interesting to consider alternative scalable control designs, as well as their use for the same type of target tracking with multiple UAVs.

Chapter 11

A Safe Stochastic Optimal Feedback Control Approach to Autonomous Navigation with a Large Number of Obstacles

This chapter is a preprint to the paper

- Dejan Milutinović, Munishkin, Alexey A., and David W. Casbeer. "A Safe Stochastic Optimal Feedback Control Approach to Autonomous Navigation with a Large Number of Obstacles." Journal of Intelligent & Robotic Systems (submitted in 2022)

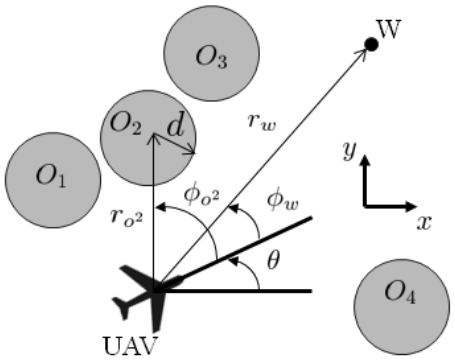


Figure 11.1: UAV navigation toward a waypoint (W) in the presence of obstacles (O_1, O_2, O_3, O_4). The heading angle θ of the UAV is measured counter-clockwise from the x-axis, and ϕ_{o^2} and ϕ_w are bearing angles of the UAV toward the 2nd obstacle and waypoint, respectively. The relative distances r_{o^2} and r_w from the UAV to the 2nd obstacle and waypoint are shown as well. The bearing angles and relative distances to the other obstacles are defined in a similar manner.

11.1 Introduction

This work is inspired by the autonomous navigation of small fixed-wing unmanned aerial vehicles (UAVs) for search and rescue, disaster response, environmental monitoring, and package delivery. These example scenarios require that the UAV autonomous navigation is *safe* and *scalable* [43, 71] to avoid multiple moving obstacles while navigating toward a waypoint as shown in Fig. 11.1. In the context of this paper, safety is related to a collision-free navigation, and scalability is related to the navigation's real-time ability to deal with a large number of obstacles. Due to the UAV navigation, we assume that the vehicle cannot stop to avoid obstacles, which makes the navigation problem even more interesting and difficult to solve.

Autonomous vehicle navigation intrinsically includes two time scales. The *long-*

time scale relates to the navigation between two geographical locations and is mostly developed around path-planning algorithms [100, 176]. The *short-time* scale addresses changes of the environment in the proximity of the vehicle that can be resolved using feedback control [85, 114]. Since both of the scales are relevant for safe navigation, an often used approach is to exploit planning algorithms (*long-time scale*) and re-compute the plans frequently at the rates (*short-time scale*) that allow for accounting changes in the proximity of the vehicle. Here, we propose an approach to the navigation in which the sequence in which we compose the navigation is reversed. We use stochastic optimal feedback control (*short-time scale*) for a vehicle to reach a target while avoiding a single randomly moving obstacle. Then, we use the feedback controller and its corresponding value function to search over a sequence of control actions (*long-time scale*) while navigating in the presence of multiple obstacles.

Related Works: Long- and short-time scale navigation strategies have been used in various robotic systems such as robotic arm movement [92, 106], autonomous car navigation [161, 66], robot-human interactions and motion planning [20], and UAV navigation [13, 71, 44]. Usually, the short-time scale navigation is in use to incrementally modify the long-time scale's collision-free paths rather than to re-plan whole paths toward the goal [44]. This modification removes unnecessary re-computations associated with long-time, i.e., global path planning methods, therefore, it can save computational time [66].

The short-time scale navigation in a rapidly changing uncertain environment has various proposed solution methods [154, 13, 90, 36, 51] and hierarchical methods seem to be the most promising [36], in particular, those methods that use feedback control. Feedback-based approaches for the short-time scale navigation use Model Predictive Control (MPC) [3, 98], rapidly-exploring random trees (RRTs) [91, 145], game theory [190, 52], barrier functions [30], reachability sets [1, 122], potential functions [178, 110], harmonic functions [112], artificial intelligence [95] and reinforcement learning [28] methods. Previous works on a scalable feedback controller that constructs a plan for future navigation in a multi-step, or repeating iteration scheme, are [195, 174], which most closely align with our work. Others include scalable frameworks for robot-human teams [20], using Voronoi cells [201], and using velocity obstacles [55] with reachable sets [188] for robot navigation.

Contributions: The first contribution of this paper is (1) the proof that a stochastic optimal feedback controller for the UAV to reach a waypoint while avoiding a single obstacle provides an upper bound for collision probability. The second contribution is (2) the proof that for the feedback controller, we can compute the probability exactly. The third contribution is (3) the proposed approach to navigation, in which we use the stochastic optimal feedback controller and its associated value function to navigate the UAV in the presence of multiple randomly moving obstacles.

Outline: The rest of this paper is organized as follows. In Section 2, we formulate

the control problem for the full state-space of reaching the waypoint while avoiding N obstacles. Then, in Section 3, we discuss the UAV-waypoint stochastic feedback optimal control problem with a single obstacle. There we introduce the feedback controller and its value function, and prove the controller properties. The use of the controller and the value function for the navigation in the presence of multiple obstacles is described in Section 4. There we introduce three search-based variants of the navigation. The three variants are compared in numerical simulations in Section 5. Concluding remarks are provided in Section 6.

11.2 Problem Formulation

Figure 11.1 depicts a small unmanned fixed-wing aerial vehicle (UAV) which navigates toward a waypoint W and avoids collisions with an unknown number of potentially moving obstacles O_i in its vicinity, $i = 1, 2, ...$ We assume that the UAV flies at a constant altitude and speed, and that it cannot stop. We describe the UAV kinematics using a deterministic Dubins vehicle [37], which is

$$dx = v\cos\theta dt \tag{11.1}$$

$$dy = v\sin\theta dt \tag{11.2}$$

$$d\theta = udt, \tag{11.3}$$

where $(x,y) \in \mathbb{R}^2$ is the UAV position, $\theta \in [-\pi,\pi)$ is the heading angle and $v > 0$ is a known positive constant speed. The control variable is the bounded turning rate $u \in [-u_{max}, u_{max}]$, where u_{max} is the maximal turning rate.

The UAV is tasked to fly towards the fixed waypoint $(x_w, y_w) \in \mathbb{R}^2$ and there is a number of circular obstacles of the radius $d > 0$ that can move in a free manner. Therefore, we describe their motion using the random Brownian walk [46] as

$$dx_{o^i} = \sigma_o dw_{x_{o^i}} \tag{11.4}$$

$$dy_{o^i} = \sigma_o dw_{y_{o^i}}, \tag{11.5}$$

where $(x_{o^i}, y_{o^i}) \in \mathbb{R}^2$, $i = 1,2,...$ are obstacle positions, and $dw_{x_{o^i}}$ and $dw_{y_{o^i}}$ denote the Wiener process increments [94]. The constant parameter $\sigma_o > 0$ is assumed to be identical for all obstacles.

Let us define the distance between the UAV and waypoint W, and the distance between the UAV and the ith obstacle O_i, r_w and r_o^i, respectively, as

$$r_w = \sqrt{(x-x_w)^2 + (y-y_w)^2} \tag{11.6}$$

$$r_{o^i} = \sqrt{(x-x_{o^i})^2 + (y-y_{o^i})^2}. \tag{11.7}$$

Furthermore, let us introduce a target set $\mathscr{X}_t$ around the waypoint W and an unsafe set $\mathscr{X}^i_{unsafe}$ around the ith obstacle O_i as

$$\mathscr{X}_t = \{(x,y) \mid r_w \leq R_{min}\} \tag{11.8}$$

$$\mathscr{X}^i_{unsafe} = \{(x,y) \mid r_{o^i} \leq d\}, i = 1,2,..., \tag{11.9}$$

where $R_{min} > 0$ and $d > 0$ are predefined distances from the waypoint and obstacles, respectively.

Problem (P): The problem we study here is a navigation strategy u such that the UAV safely navigates to reach the target set $\mathscr{X}_t$ and that the vehicle avoids all obstacles, i.e, that for $t > 0$ and $\mathscr{X}_{unsafe} = \cup_{i=1}^{N} \mathscr{X}^i_{unsafe}$

$$(x(0), y(0)) \notin \mathscr{X}_{unsafe} \Rightarrow \begin{cases} \exists \tau > 0, (x(\tau), y(\tau)) \in \mathscr{X}_t, \text{ and} \\ (x(t), y(t)) \notin \mathscr{X}_{unsafe},\ 0 < t \leq \tau. \end{cases} \tag{11.10}$$

In this paper, we address the problem using the relative position between the UAV and the waypoint as (r_w, ϕ_w) and between the UAV and each ith obstacle as (r_{o^i}, ϕ_{o^i}). The relative position coordinates are distances r_w and r_{o^i} given by (11.6) and (11.7), respectively, and the bearing angles ϕ_w and ϕ_{o^i} from Fig. 11.1 given by

$$\phi_w = \arctan\left(\frac{y - y_w}{x - x_w}\right) - \theta \tag{11.11}$$

$$\phi_{o^i} = \arctan\left(\frac{y - y_{o^i}}{x - x_{o^i}}\right) - \theta. \tag{11.12}$$

Using the relative position coordinates and Îto calculus [94], we can rewrite the stochastic kinematics (11.1)-(11.5) as

$$dr_w = \left(-v\cos(\phi_w) + \frac{\sigma_w^2}{2r_w}\right)dt = b_{r_w}dt \tag{11.13}$$

$$d\phi_w = \left(\frac{v}{r_w}\sin(\phi_w) - u\right)dt = b_{\phi_w}dt \tag{11.14}$$

$$dr_{o^i} = \left(-v\cos(\phi_{o^i}) + \frac{\sigma_o^2}{2r_{o^i}}\right)dt + \sigma_o dw_{r_{o^i}} = b_{r_{o^i}}dt + n_{r_{o^i}}dw_{r_{o^i}} \tag{11.15}$$

$$d\phi_{o^i} = \left(\frac{v}{r_{o^i}}\sin(\phi_{o^i}) - u\right)dt + \frac{\sigma_o}{r_o^i}dw_{(\phi_{o^i}+\theta)} = b_{\phi_{o^i}}dt + n_{\phi_{o^i}}dw_{(\phi_{o^i}+\theta)}, \tag{11.16}$$

where $dw_{r_{o^i}}$ and $dw_{(\phi_{o^i}+\theta)}$ denote the Wiener process increments [94] along the r_{o^i} and ϕ_{o^i} state-space variables. However, with this, we have to re-state the problem formulation term of the state vector

$$\tilde{x} = [\tilde{x}_w,\ \tilde{x}_{o^1}, ...\tilde{x}_{o^N}]^T, \tag{11.17}$$

which keeps the information about all relative positions between the vehicle and waypoint with $\tilde{x}_w = [r_w, \phi_w]^T$, and the vehicle and obstacles with $\tilde{x}_{o^i} = [r_{o^i}, \phi_{o^i}]^T$, $i = 1..N$, where $N \geq 1$ corresponds to the number of obstacles. With this, we can easily re-state the target and unsafe sets, as well as condition (11.10) as

$$\mathscr{X}_t = \{\tilde{x} \mid r_w \leq R_{min}\} \tag{11.18}$$

$$\mathscr{X}^i_{unsafe} = \{\tilde{x} \mid r_{o^i} \leq d\} \tag{11.19}$$

$$\tilde{x}(0) \notin \mathscr{X}_{unsafe} \Rightarrow \begin{cases} \exists \tau > 0, \tilde{x}(\tau) \in \mathscr{X}_t, \text{ and} \\ \tilde{x}(t) \notin \mathscr{X}_{unsafe},\ 0 < t \leq \tau, \end{cases} \tag{11.20}$$

while we can keep the rest of problem formulation (P) the same.

11.3 Safe Navigation for a Single Obstacle

The starting point of our analysis is to consider the case in which the vehicle navigates toward the waypoint in the presence of a single obstacle i. In that case, the

stochastic kinematic equations (11.13)-(11.16) are

$$d\tilde{x}_i = \begin{bmatrix} b_w(\tilde{x}_w, u) \\ \\ b_{o^i}(\tilde{x}_{o^i}, u) \end{bmatrix} dt + \begin{bmatrix} 0 \\ \\ n_{o^i}(\tilde{x}_{o^i}) \end{bmatrix} dW = b(\tilde{x}_i, u)dt + n(\tilde{x}_i)dW_i, \tag{11.21}$$

where $b_w(\cdot) = [b_{r_w}, b_{\phi_w}]^T$, $n_o^i(\cdot) = [n_{r_o^i}, n_{\phi_o^i}]^T$ and b_{o^i} are defined by (11.13)-(11.16), the Wiener process increment $dW_i = [0, 0, dw_{r_{o^i}}, dw_{(\phi_{o^i}+\theta)}]^T$ has components defined by (11.15)-(11.16) and $\tilde{x}_i \in R^4$.

Let us assume that the state $\tilde{x}_i(t)$ in (11.21) is controlled by a feedback control strategy $u = u(\tilde{x}_i)$. Therefore, with $b_u(\tilde{x}_i) = b(\tilde{x}_i, u(\tilde{x}_i))$, we can conclude that

$$d\tilde{x}_i = b_u(\tilde{x}_i)dt + n(\tilde{x}_i)dW_i. \tag{11.22}$$

This nonlinear stochastic differential equation was analyzed in [141] with regard to the probability P of reaching the unsafe set from a state $\tilde{x}_i(s)$ at the time point s, i.e.,

$$P\left\{\tilde{x}_i(\tau) \in \mathscr{X}^i_{unsafe} \text{ for some } \tau \geq s \mid \tilde{x}_i(s)\right\} = P(\tilde{x}_i(s)) \leq \gamma, \tag{11.23}$$

where τ is the first time point at which a trajectory $\tilde{x}_i(t)$, $s \leq t \leq \tau$ enters the unsafe set $\mathscr{X}^i_{unsafe}$, and γ is the upper bound of that probability. Specifically, the result in [141](Theorem 15), when applied to (11.22), states that (11.23) holds if there exists the

so-called barrier function B which has continuous second derivatives and satisfies

$$B(\tilde{x}_i) \leq \gamma,\ \gamma \in [0,1],\ \forall \tilde{x}_i \in \mathscr{X}_0 \subseteq R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\} \tag{11.24}$$

$$B(\tilde{x}_i) \geq 1,\ \forall \tilde{x}_i \in \mathscr{X}^i_{unsafe} \tag{11.25}$$

$$B(\tilde{x}_i) \geq 0,\ \forall \tilde{x}_i \in \mathscr{X}_0 \subseteq R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\} \tag{11.26}$$

$$\mathscr{L}^u B(\tilde{x}_i) \leq 0 \quad ,\ \forall \tilde{x}_i \in \mathscr{X}_0 \subseteq R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}, \tag{11.27}$$

where $\mathscr{L}^u$ is the differential operator

$$\mathscr{L}^u = \frac{\partial(\cdot)}{\partial \tilde{x}} b_u(\tilde{x}_i) + \frac{1}{2} Tr\left(n^T(\tilde{x}_i) \frac{\partial^2(\cdot)}{\partial \tilde{x}^2} n(\tilde{x}_i)\right). \tag{11.28}$$

In the following, we show that conditions (11.24)-(11.27) hold for the solution of the Hamilton-Jacobi-Bellman (HJB) partial differential equation [94] associated with a stochastic optimal control problem.

Let us consider a control $u = u(\tilde{x}_i)$ which minimizes the following cost function

$$J(\tilde{x}_i(s), u) = E\left\{g(\tilde{x}_i(\tau_s)) + \int_t^{\tau_s} dt\right\},\ \tilde{x}_i(s) \in R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}, \tag{11.29}$$

where E is the expectation with respect to stochastic trajectories of $\tilde{x}_i(t)$ and $s \leq t \leq \tau_s < \infty$ is the first time of entry of $\tilde{x}_i(t)$ to the closed set terminal set $\{\mathscr{X}^i_{unsafe} \cup \mathscr{X}_t\}$ from an initial state $\tilde{x}_i(s) \in R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}$. The part of cost under the integral accounts for the time to reach the terminal set, while the terminal cost part $g(\tilde{x}_i(\tau_s))$ is

$$g(\tilde{x}_i(\tau)) = \begin{cases} M & \text{if } \tilde{x}_i(\tau_s) \in \mathscr{X}^i_{unsafe} \\ 0 & \text{if } \tilde{x}_i(\tau_s) \in \mathscr{X}_t, \end{cases} \tag{11.30}$$

where $M \gg 0$ is a large positive constant to denote a large penalty for hitting an obstacle, i.e., entering the set $\mathscr{X}^i_{unsafe}$. The optimal control solution u^* that achieves the minimum cost in (11.29) results in the value function

$$V(\tilde{x}_i) = \min_u E\{J(\tilde{x}_i, u)\}. \tag{11.31}$$

The value function satisfies the HJB partial differential equation [94]

$$0 = \min_u \{\mathscr{L}^u V(\tilde{x}_i) + 1\}, \;\; \tilde{x}_i \in R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}. \tag{11.32}$$

The HJB equation is defined by the same differential operator $\mathscr{L}^u$ as in (11.28) and boundary condition

$$V(\tilde{x}_i) = \begin{cases} M, \; \tilde{x}_i \in \mathscr{X}^i_{unsafe} \\ 0, \; \tilde{x}_i \in \mathscr{X}_t. \end{cases} \tag{11.33}$$

The solution of the HJB equation and the following theorem establish the relation between V and the barrier function B. In the following, we will assume that $V \in C^2(\tilde{x}_i)$, i.e., that it is a function with continuous second derivatives, which means that the HJB equation has a classical solution. This is a suitable assumption given that the same assumption is used in the definition of the barrier function B. In the next theorem, we use the following assumptions:

(A1) State $\tilde{x}_i$ obeys (11.22);
(A2) $V(\tilde{x}_i)$ is the solution of the HJB equation (11.32) for the optimal control problem that minimizes the cost (11.29) and has continuous second derivatives;
(A3) $V(\tilde{x}_i) < M$, $\tilde{x}_i \in R^4 \setminus \{\mathscr{X}^i_{unsafe} \cup \mathscr{X}_t\}$, where M is the penalty from the terminal costs (11.30), which is the same value used in the boundary condition (11.33).

Theorem 1: If all the assumptions (A1), (A2) and (A3) are satisfied, then the function $Y(\tilde{x}_i) = \frac{1}{M}V(\tilde{x}_i)$ is a barrier function.

Proof: Our proof is based on checking conditions (11.24)-(11.27). The costs, as well as the terminal costs are non-negative, therefore, $V(\tilde{x}_i) \geq 0$ for all $\tilde{x}_i \in R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}$, which implies that $Y(\tilde{x}_i) \geq 0$ and confirms that $Y(\tilde{x}_i)$ satisfies (11.26). Also, since $M = V(\tilde{x}_i)$ for all $\tilde{x}_i \in \mathscr{X}^i_{unsafe}$, we conclude that $Y(\tilde{x}_i) = 1$ and confirm $Y(\tilde{x}_i)$ satisfies (11.25).

Next, for the optimal control $u = u^*$, we have that

$$0 = \mathscr{L}^{u^*}V(\tilde{x}_i) + 1, \tag{11.34}$$

for $\tilde{x}_i \in R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}$, which after the substitution $V(\tilde{x}_i) = MY(\tilde{x}_i)$ can be rewritten as

$$\mathscr{L}^{u^*}Y(\tilde{x}_i) = -\frac{1}{M} < 0, \tag{11.35}$$

confirming that $Y(\tilde{x}_i)$ satisfies (11.27).

Finally, from the assumption that $V(\tilde{x}_i) < M$ for all $\tilde{x}_i \in R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}$ and $V(\tilde{x}_i) = M$ for $\tilde{x}_i \in \mathscr{X}^i_{unsafe}$, we can conclude that for any $\tilde{x}_i \in \mathscr{X}_0$, $\mathscr{X}_0 \subseteq R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}$, we also have $V(\tilde{x}_i) < M$, i.e., that $Y(\tilde{x}_i) < \gamma$, $\gamma \in [0,1)$, which implies that (11.24) is satisfied by $Y(\tilde{x}_i)$. Since we have verified that $Y(\tilde{x}_i)$ satisfies all conditions (11.24)-(11.27), we conclude that the function $Y(\tilde{x}_i)$ is a barrier function and finish the proof. □

There are two reasons for presenting *Theorem 1*. The first and main reason is to illustrate the relation between the recently proposed concept of barrier function and the use of stochastic optimal control, i.e., its corresponding HJB solution. The second one is because *Theorem 1* is applicable for the general stochastic differential equation. However, the continuity of second derivatives of the solution $V(\tilde{x}_i)$ is not guaranteed and there is a certain condition that has to be satisfied, e.g., that the HJB is *uniformly parabolic* (see [58], page 156). This may give an impression that the HJB approach for the specification of the barrier function is more restrictive than the one from [141], but it is quite the opposite. When the HJB solution $V(\tilde{x}_i)$ does not have continuous second derivatives, the HJB solution for the value function $V(\tilde{x}_i)$ can still exist in a weaker sense, which is a viscosity solution. Viscosity solutions represent a wider class of solutions for $V(\tilde{x}_i)$ and solutions with continuous second derivatives, i.e., classical solutions also belong to the class of viscosity solutions.

The definition of viscosity solution is provided in [58], and we use it to define a hazard function $H(\tilde{x}_i)$. To introduce the definition, we use the HJB equation (11.32), from which we obtain the optimal control u^*. Furthermore, based on the optimal control u^*, we define the steady-state solution of the backward Kolmogorov (BK) [46] equation

$$0 = \mathscr{L}^{u^*} T(\tilde{x}_i) + 1, \;\; \tilde{x}_i \in R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}, \tag{11.36}$$

with the boundary condition $T = 0$ for all $\tilde{x}_i \in \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}$. The BK equation solution is the expected time T to reach either the target set $\mathscr{X}_t$ or the unsafe set $\mathscr{X}^i_{unsafe}$.

Finally, instead of the assumption (A2), we use the following one:

(A2') $V(\tilde{x}_i)$ is the viscosity solution of the HJB equation (11.32) for the optimal control problem that minimizes the cost (11.29), u^* is the optimal control and $T(\tilde{x}_i)$ is the viscosity solution of the BK equation (11.36).

Definition 1: Under assumptions (A1), (A2') and (A3), the hazard function $H(\tilde{x}_i)$ is

$$H(\tilde{x}_i) = V(\tilde{x}_i) - T(\tilde{x}_i),\ \tilde{x}_i \in R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}. \tag{11.37}$$

Theorem 2: The hazard function $H(\tilde{x}_i)$ satisfies the partial differential equation

$$(a):\ \ 0 = \mathscr{L}^{u^*} H(\tilde{x}_i),\ \ \tilde{x}_i \in R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}, \tag{11.38}$$

with the boundary condition $H(\tilde{x}_i) = M$, for all $\tilde{x}_i \in \mathscr{X}^i_{unsafe}$ and M is from (A3). Also,

$$(b): \tilde{x}_i \in \mathscr{X}_t \Rightarrow \frac{1}{M} H(\tilde{x}_i) = 0 \text{ and } \tilde{x}_i \in \mathscr{X}^i_{unsafe} \Rightarrow \frac{1}{M} H(\tilde{x}_i) = 1, \tag{11.39}$$

and the probability $P(\tilde{x}_i(s))$ of reaching the unsafe set $\mathscr{X}^i_{unsafe}$ from (11.23) evaluated for $\tilde{x}_i(s) = \tilde{x}_i$ is

$$(c): P(\tilde{x}_i) = \frac{1}{M} H(\tilde{x}_i), \tag{11.40}$$

where $\tilde{x}_i \in R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\}$.

Proof: To prove (a), we start from (11.32) written for the optimal control $u = u^*$ and the hazard definition $H(\tilde{x}_i) = V(\tilde{x}_i) - T(\tilde{x}_i)$ to obtain

$$0 = \mathscr{L}^{u^*} V(\tilde{x}_i) + 1 = \mathscr{L}^{u^*} \left(H(\tilde{x}_i) + T(\tilde{x}_i)\right) + 1 = \mathscr{L}^{u^*} H(\tilde{x}_i) + \underbrace{\mathscr{L}^{u^*} T(\tilde{x}_i) + 1}_{0}, \tag{11.41}$$

where the last part of the expression is 0 since $T(\tilde{x}_i)$ is the solution of the BK equation (11.36). Therefore, we conclude that $0 = \mathscr{L}^{u^*} H(\tilde{x}_i)$. To show (b), we should note that

$$\tilde{x}_i \in \mathscr{X}_t \Rightarrow V(\tilde{x}_i) = T(\tilde{x}_i) = 0 \Rightarrow \frac{1}{M} H(\tilde{x}_i) = \frac{1}{M} \left(V(\tilde{x}_i) - T(\tilde{x}_i) \right) = 0 \tag{11.42}$$

and

$$\tilde{x}_i \in \mathscr{X}^i_{unsafe} \Rightarrow V(\tilde{x}_i) = M, T(\tilde{x}_i) = 0 \Rightarrow \frac{1}{M} H(\tilde{x}_i) = \frac{1}{M} \left(V(\tilde{x}_i) - T(\tilde{x}_i) \right) = 1, \tag{11.43}$$

and with this we have proved (b).

To prove part (c), let us write $P(\tilde{x}_i(s))$ using the conditional probability density function (pdf) $\pi(\tilde{x}_i(\tau)|\tilde{x}_i(s))$ of $\tilde{x}_i(\tau) \in \mathscr{X}^i_{unsafe}$, given $\tilde{x}_i(s)$, i.e.,

$$P(\tilde{x}_i(s)) = \int_{R^4} \frac{1}{M} g(\tilde{x}_i(\tau)) \pi(\tilde{x}_i(\tau)|\tilde{x}_i(s)) d\tilde{x}_i(\tau). \tag{11.44}$$

The term $\frac{1}{M} g(\tilde{x}_i(\tau))$ accounts for the probability mass in the unsafe set $\mathscr{X}^i_{unsafe}$ since according to (11.30), the term $\frac{1}{M} g(\tilde{x}_i(\tau)) = 1$ for $\tilde{x}_i(\tau) \in \mathscr{X}^i_{unsafe}$ and $\frac{1}{M} g(\tilde{x}_i(\tau)) = 0$, otherwise. Similarly, we can express the probability $P(\tilde{x}_i(s+ds))$, where $ds > 0$ is an infinitesimally small increment of time as

$$P(\tilde{x}_i(s+ds)) = \int_{R^4} \frac{1}{M} g(\tilde{x}_i(\tau)) \pi(\tilde{x}_i(\tau)|\tilde{x}_i(s+ds)) d\tilde{x}_i(\tau). \tag{11.45}$$

Since the joint conditional pdf of $(\tilde{x}_i(\tau), \tilde{x}_i(s+ds))$, given $\tilde{x}_i(s)$, is

$$\pi(\tilde{x}_i(\tau), \tilde{x}_i(s+ds)|\tilde{x}_i(s)) = \pi(\tilde{x}_i(\tau)|\tilde{x}_i(s+ds), \tilde{x}_i(s)) \pi(\tilde{x}_i(s+ds)|\tilde{x}_i(s)) \tag{11.46}$$

and due to the Markov property $\pi(\tilde{x}_i(\tau)|\tilde{x}_i(s+ds),\tilde{x}_i(s)) = \pi(\tilde{x}_i(\tau)|\tilde{x}_i(s+ds)$, we obtain

$$\pi(\tilde{x}_i(\tau)|\tilde{x}_i(s)) = \int_{R^4} \pi(\tilde{x}_i(\tau)|\tilde{x}_i(s+ds))\pi(\tilde{x}_i(s+ds)|\tilde{x}_i(s))d\tilde{x}_i(s+ds), \tag{11.47}$$

which after the substitution in (11.44) yields

$$P(\tilde{x}_i(s)) = \int_{R^4}\int_{R^4} \frac{1}{M} g(\tilde{x}_i(\tau)) \quad \pi(\tilde{x}_i(\tau)|\tilde{x}_i(s+ds)) \times \tag{11.48}$$

$$\pi(\tilde{x}_i(s+ds)|\tilde{x}_i(s))d\tilde{x}_i(s+ds)d\tilde{x}_i(\tau), \tag{11.49}$$

and after accounting for (11.45) results in

$$P(\tilde{x}_i(s)) = \int_{R^4} P(\tilde{x}_i(s+ds))\pi(\tilde{x}_i(s+ds)|\tilde{x}_i(s))d\tilde{x}_i(s+ds). \tag{11.50}$$

The last expression can be rewritten as

$$P(\tilde{x}_i(s)) = E\left\{P(\tilde{x}_i(s+ds))\right\}, \tag{11.51}$$

where the expectation operator is conditional accounting for the pdf of $\tilde{x}_i(s+ds)$, given $\tilde{x}_i(s)$. Finally, using the definition of infinitesimal generator for stochastic processes, we obtain

$$\mathscr{L}^{u^*}P(\tilde{x}_i) = \lim_{ds\to 0} \frac{E\left\{P(\tilde{x}_i(s+ds)\right\} - P(\tilde{x}_i(s))}{ds} \tag{11.52}$$

$$= \lim_{ds\to 0} \frac{P(\tilde{x}_i(s)) - P(\tilde{x}_i(s))}{ds} = 0. \tag{11.53}$$

To conclude, the probability $P(\tilde{x}_i)$ satisfies

$$0 = \mathscr{L}^{u^*}P(\tilde{x}_i),\ \tilde{x} \in R^4 \setminus \{\mathscr{X}_t \cup \mathscr{X}^i_{unsafe}\} \tag{11.54}$$

and due to its definition, $P(\tilde{x}_i) = 1$ for $\tilde{x}_i \in \mathscr{X}^i_{unsafe}$ and $P(\tilde{x}_i) = 0$ for $\tilde{x}_i \in \mathscr{X}_t$. These two boundary values match the boundary conditions for $\frac{1}{M}H(\tilde{x}_i)$ (see part (b) of the theorem), which also satisfies $\mathscr{L}^{u^*}\left(\frac{H(\tilde{x}_i)}{M}\right) = 0$ due to $\mathscr{L}^{u^*}H(\tilde{x}_i) = 0$ (see part (a) of the theorem) and the linearity of the operator $\mathscr{L}^{u^*}$. Therefore, under the assumption (A2'), i.e, the viscosity solution for $H(\tilde{x}_i)$, i.e., $\frac{1}{M}H(\tilde{x}_i)$, there is also a solution of (11.54) for $P(\tilde{x}_i)$ which is $P(\tilde{x}_i) = \frac{1}{M}H(\tilde{x}_i)$, which concludes our proof. □

11.4 Multi-step Optimization for a Safe Navigation around Multiple Obstacles

In principle, one can formulate an optimal control similar to (11.29) for the whole state vector $\tilde{x} = [\tilde{x}_w, \tilde{x}_{o^1}, ...\tilde{x}_{o^N}]^T$ which accounts for the UAV-target, all UAV-obstacle relative positions, and corresponding target $\mathscr{X}_t$ and unsafe sets $\mathscr{X}^i_{unsafe}$. This formulation would result in an HJB equation for $V(\tilde{x})$ that can be used to obtain the optimal control u^*. However, in that case, the HJB equation has to be solved in $2(N+1)$ dimensions, which is computationally challenging [135], even for a small number of obstacles, e.g., $N = 5$. Therefore, we take an approach in which we utilize the solution for a single obstacle from the previous section.

Instead of the single vector $\tilde{x}$, here we deal with multiple vectors $\tilde{x}_i$, $i = 1, 2, ...N$. Note that the general solution from the previous section for navigation to W avoiding

one obstacle i applies for any $i = 1,2,\ldots N$, in other words,

$$\tilde{x}_i = \tilde{x}_j \Rightarrow u(\tilde{x}_i) = u(\tilde{x}_j), V(\tilde{x}_i) = V(\tilde{x}_j), T(\tilde{x}_i) = T(\tilde{x}_j), H(\tilde{x}_i) = H(\tilde{x}_j). \tag{11.55}$$

This similarity gives us a reason to consider the single controller

$$u = u(x_{i_{max}}),\ i_{max} = \arg\max_i \{H(\tilde{x}_1), H(\tilde{x}_2), \ldots\}. \tag{11.56}$$

In this case, the control accounts only for the target and the obstacle with the highest hazard. This may be an interesting solution, but in general it is myopic and can fail in the case of a cluttered environment since the control reacts only to the single and usually the closest obstacle. An alternative to this approach is to adopt the one-step look-ahead optimal controller proposed in [124] which defines the control by minimizing an approximation of the value function. However, the single step minimization may not be sufficient in the case of a set of obstacles blocking the path toward the target from multiple directions. For this reason, here we propose a control defined from an algorithm that accounts not for a single, but for multiple steps ahead.

To formulate the approach, let us define V^{upper}, which is the largest value of the array of value functions $V(\tilde{x}_i)$, $i = 1,2...N$, i.e.,

$$V^{upper}(X(t)) = \max_i \{V(\tilde{x}_1(t)), V(\tilde{x}_2(t)), \ldots\}, \tag{11.57}$$

where on the right side we have introduced the vector $X(t) = [\tilde{x}_1(t), \tilde{x}_2(t) \ldots \tilde{x}_N(t)]$ and t denotes the time. Below we explain the computation for the control u_k.

Multi-step Optimization: Let us assume that we know $V^{upper}(X(t_k))$. In the first step, we predict $X(t_{k+1})$ from $X(t_k)$, assign $s_0 = t_k$ and perform the minimization

$$(Step\ 1): \quad V^{upper}(X(s_1)) = \min_{c_0} V^{upper}(\overline{X}(s_1)|X(s_0), c_0), \tag{11.58}$$

where $\overline{X}(s_1)|X(s_0), c_0$ denotes that the prediction of $X(s_1)$, $s_1 > s_0$ depends on $X(s_0)$ and control c_0. If we can confirm that

$$V^{upper}(\overline{X}(s_1)) < V^{upper}(X(s_0)), \tag{11.59}$$

then the control for the step t_k is $u_k = c_0$. Otherwise, we go to the second step, in which we search over c_0 and c_1 control values to perform the minimization

$$(Step\ 2): \quad V^{upper}(\overline{X}(s_2)) = \min_{c_1, c_0} V^{upper}(\overline{X}(s_2)|\overline{X}(s_1), c_1|X(s_0), c_0), \tag{11.60}$$

where the notation in the term on the right side explains that $\overline{X}(s_2)$ depends on the previous predictions and the corresponding controls. If we can confirm that

$$V^{upper}(\overline{X}(s_2)) < V^{upper}(X(s_0)), \tag{11.61}$$

then the control for the step t_k is $u_k = c_0$. Otherwise, we proceed with an attempt to compute $V^{upper}(\overline{X}(s_3))$. In general, if we cannot verify $V^{upper}(\overline{X}(s_j)) < V^{upper}(X(s_0))$ in any of the steps up to the nth step, $j = 1, ...n-1$, where *Step n* is

$$(Step\ n): \quad V^{upper}(\overline{X}(s_n)) = \min_{c_n, ...c_1, c_0} V^{upper}(\overline{X}(s_n)|...|\overline{X}(s_1), c_1|X(s_0), c_0), \tag{11.62}$$

in which we check if

$$V^{upper}(\overline{X}(s_n)) < V^{upper}(X(s_0)). \tag{11.63}$$

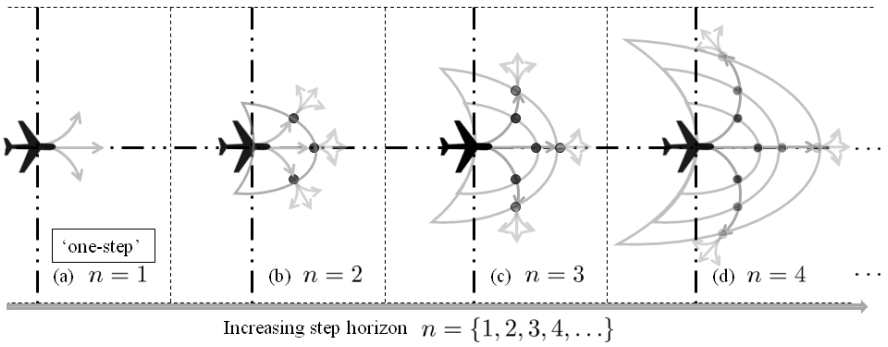


Figure 11.2: Illustration of the multi-step UAV navigation approach using BFS. In the first step ($n = 1$), we search over a single control action (see (11.58)). For simplicity, in the illustration we assume that there are only three available control actions, go straight, i.e., turning rate 0, turn-left, i.e., a positive turning rate, and turn-right, i.e., a negative turning rate. As we progress through the steps $n \in \{2, 3, 4, \ldots\}$, we search over longer sequences of the three control actions as illustrated in the panels (b)-(d).

If the answer is positive (true or yes), then $u_k = c_0$. Otherwise, we continue to the next step. We continue with this process of checking condition (11.63) and if positive, assigning $u_k = c_0$, or until we reach some maximal value for $n = n_{max}$. In the first case, we apply the control u_k and we repeat the same process in the next time step. In the second case, we state that we cannot find the solution for the control u_k. In a larger navigation system, this is the time point when the UAV can request help from a planner that has a more holistic view of the UAV mission and environment.

Navigation search algorithms (BFS, DFS, BFSC): The process of finding the V^{upper} minimizing sequence of controls c_i, $i \leq n-1$, for each step n in (11.58)-(11.63), can be seen as a tree search traversal problem [167, 35] if c_i takes discrete values from the set $[-u_{max}, u_{max}]$. The root of the tree corresponds to the UAV initial position, the depth of the tree corresponds to the step n, and each node of the tree graph at the depth n

corresponds to a single sequence $\{c_0, c_1, ...c_n\}$. During the search, the algorithm first goes over the nodes at the same depth, and if the search is not concluded, then it goes to the next depth $n+1$. This search process goes on until the solution is found or the depth $n = n_{max}$ is reached. This is the the so-called Breadth-first search (BFS) [167, 35]. However, the search can be organized by going in the depth. If at the certain node with the sequence $A : \{c_0, c_1, ...c_n\}$ at the depth n the condition (11.63) is not satisfied, the search can go immediately to the node $B : \{c_0, c_1, ...c_n, c_{n+1}\}$ obtained by adding c_{n+1} to the sequence A. This search process goes on until the solution is found or the depth $n = n_{max}$ is reached. In the latter case, the search returns to the node at a lower depth that has not been explored. This type of search is the so-called Depth-first search (DFS) [167, 35]. The algorithms describing BFS and DFS are provided in Appendix B.

In addition to the BFS and DFS search, in this work we also use the search process in which the search at the depth n goes only over nodes in which $c_i = c_{i-1}$, $i = 1, ...n-1$, therefore, over the sequences that keep the control constant, e.g., keep turning left or right, or keep going straight. Since we have implemented this search as the BFS, we call it BFSC, where the last 'C' stands for the word *constant*. The algorithm is also described in Appendix B and we introduce it in the paper only as a benchmark.

11.5 Results

In this section, we present three different simulation scenarios for a UAV reaching a waypoint while avoiding collision with obstacles. The first two are scenarios with the UAV navigation through a corridor of obstacles and around a circular wall of obstacles. The third is a scenario in which the UAV avoids a randomly-generated collection of moving random obstacles. We conclude this section with a comparison among BFS, DFS and BFSC search-based navigations.

In the simulations, the UAV model is expressed by the Dubins kinematics (11.1)-(11.3) and the model for each stochastic moving obstacle is expressed in (11.4)-(11.5). To compute the discrete approximation V^h of the value function V from Sections 3 and 4, we use the discrete step sizes (see Appendix B) of $\Delta r_w = \Delta r_o = (R_{max} - R_{min})/100$ (m), $\Delta\phi_w = \Delta\phi_o = 5\pi/180$ (rad), where $R_{max} = 2.04$ (m) and $R_{min} = 0.05$ (m) are the maximum and minimum distances to either the waypoint or obstacle in the discretized state-space $\tilde{x}^h$. For the BFS, DFS and BFSC search methods, we use the maximum depth (step) $n_{max} = 5$ and discrete time steps of 0.1 seconds. The maximum turning rate $u_{max} = 1$ rad/s and we use discrete control steps $\Delta u = 0.5$ rad/s, therefore, all control actions u_k are from the set $\{-1, -0.5, 0, 0.5, 1\}$. Since $R_{min} > 0$, we stop the simulation when the UAV has reached the R_{min} distance from the waypoint or when the UAV has collided with an obstacle of the radius $d = 0.1$ (m).

Out of the three search algorithms, i.e., BFS, DFS and BFSC, the last one is ex-

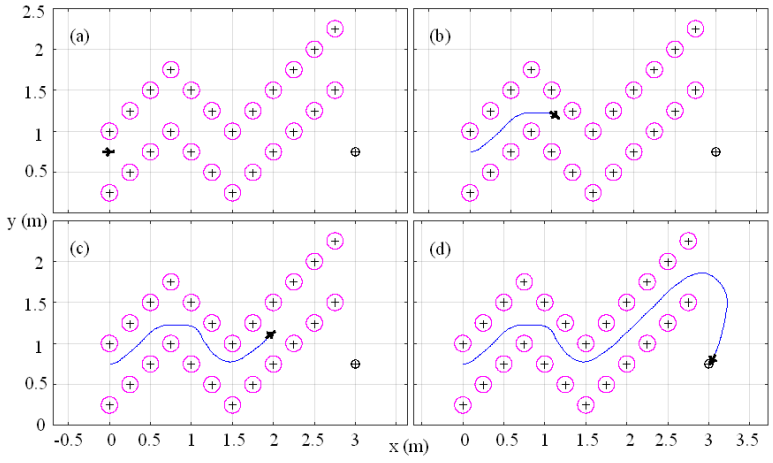


Figure 11.3: Corridor of stationary obstacles. The UAV starts (a) at (0,0.75) and navigates to reach the waypoint at (3,0.75) (black circle). During the navigation, the UAV avoids the obstacles (magenta circles) as illustrated in ther panels (a)-(d).

pected to be the worst one since it searches only over a subset of nodes, i.e., control sequences. Therefore, for the stationary set of obstacles in Fig. 11.3 and Fig. 11.4, we show results for the BFSC, which represent the worst case scenario. Fig. 11.3 shows the navigation obtained by the BFSC. Fig. 11.3(a) shows the initial UAV location with the heading angle $\theta = 0$ (rad), the waypoint and location of all twenty-four obstacles forming the corridor. Figs. 11.3 shows that the UAV navigated the obstacle corridor towards the waypoint which was reached as shown in Fig. 11.3(d).

Fig. 11.4 shows a scenario with twelve obstacles forming a semi-circular wall. The UAV is facing the wall with the heading angle $\theta = 0$ (rad) and navigates using the BFSC. Fig. 11.4(a) shows the initial UAV location and heading. Figs. 11.4(b) and (c) show the navigation of the UAV as it transitioned from the inner to the outer side of the wall and moved toward the waypoint. Fig. 11.4(d) shows that the UAV reached the

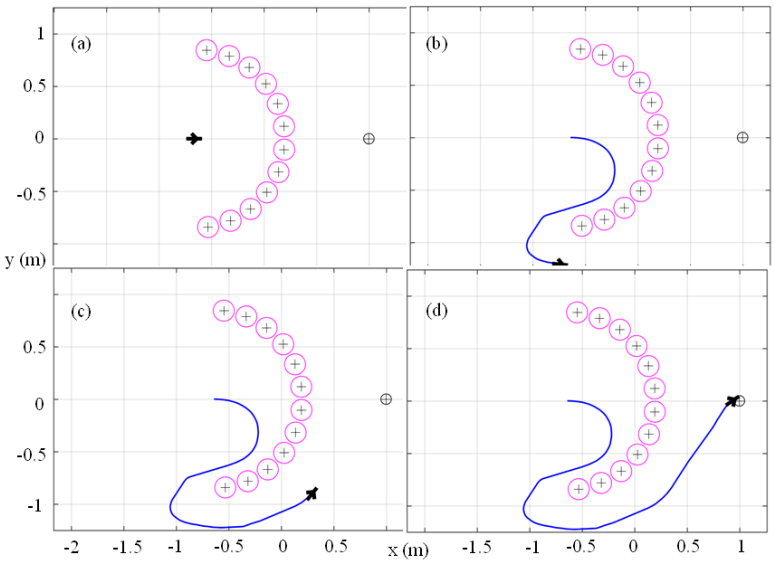


Figure 11.4: Semi-circular wall of stationary obstacles. The UAV starts (a) at (-0.75,0) and navigates to reach the waypoint at (1,0) (black circle). During the navigation, the UAV avoids the obstacles (magenta circles) as illustrated in the panels (a)-(d).

waypoint.

Finally, Fig. 11.5 illustrates a scenario with twenty-four obstacles, showing that the BFSC search-based navigation can handle unstructured and random scenarios where the obstacles move randomly. Initially, given the unfavorable starting condition in Fig. 11.5a where the UAV faces close-by obstacles, the UAV makes a hard right to avoid collision and then has to maneuver to the left to avoid collision as shown in Fig. 11.5(b) and (c). Once the UAV has reached a clear region, the UAV was in the position to reach the waypoint as depicted in Figure 11.5(d).

Comparison of BFS, DFS and BFSC search-based navigations: For the comparison, we use all the same parameters as in the other simulations from this section. However,

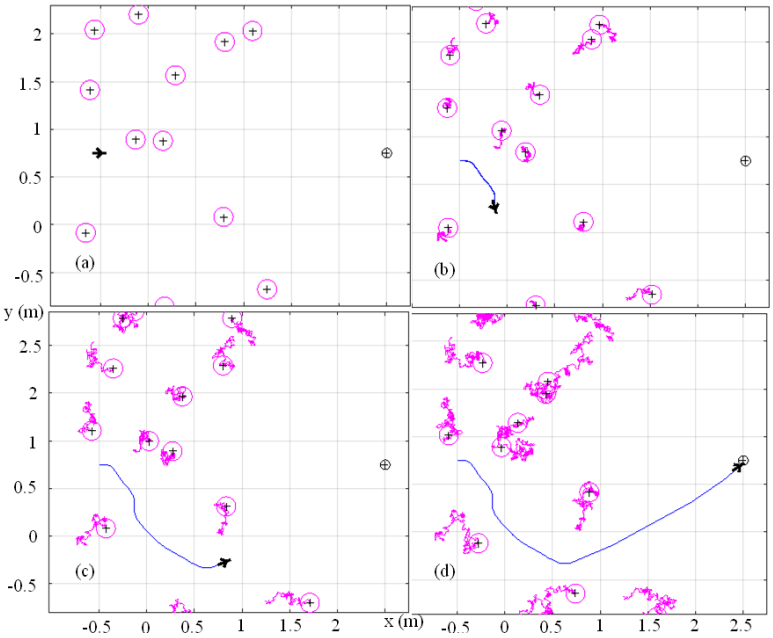


Figure 11.5: Randomly moving obstacles. The UAV starts (a) at (-0.5,0.75) and navigates to reach the waypoint at (2.5,0.75) (black circle). During the navigation, the UAV avoids the obstacles (magenta circles) as illustrated in the panels (a)-(d).

when it comes to $n_{max} = 5$, as well as 5 available values for control actions, the tree that we need to explore has 3905 nodes (see Appendix B). To reduce the search time, we use $n_{max} = 3$, only three actions from the set $\{-1, 0, 1\}$ and only 39 tree nodes, i.e., sequences that we need to explore. With this, we gain a speed-up which allows us to run 200 simulation runs for the UAV starting at $(-1.5, 0)$ with the heading $\theta = 0$, the waypoint at $(0, 0)$, and a varying number of randomly placed, randomly moving obstacles from 2 to 50. To compare the use of BFS, DFS, and BFSC search algorithms in the navigation, out of 200 simulations for each obstacle scenario, we count how many simulation runs end with the collision between the UAV and an obstacle.

Figure 11.6 illustrates the results from a simulation with the obstacle noise parameter $\sigma = 0.039$ which is the same as the σ_o value used to compute the stochastic control in Section 11.3, and corresponds to the obstacle mean speed of 0.07 m/s, which is 70% of the UAV speed of $v = 0.1$ m/s. The figure shows that with 50 obstacles, the DFS search-based navigation reaches the range of 70% of collisions. While the BFS-based navigation performs slightly better than the BFSC one, they are consistently better than the DFS-based navigation. Next, we set $\sigma_o = 0.0273$ corresponding to the obstacle mean speed, which is 49% of the UAV speed of $v = 0.1$ m/s, and repeat the whole simulation to compute the results in Fig. 11.7. The results show that the collision rates across all approaches decrease, but that their relative performances remain the same, i.e., that the BFS-based navigation is slightly better than the BFSC one and that both

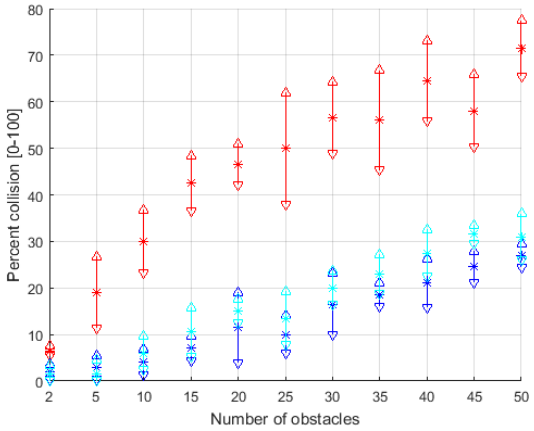


Figure 11.6: Percent of collisions vs. number of obstacles from 200 simulations with $\sigma_o = 0.039$. The data points are for DFS- (red), BFS- (blue) and BFSC- (cyan) based navigation.

are consistently better than the DFS-search based navigation. This is also confirmed with the results in Fig. 11.8, which are computed for $\sigma_o = 0.0039$. This σ_o value corresponds to the obstacle mean speed of 0.007 m/s, which is 7% of the UAV speed of $v = 0.1$ m/s.

Overall, we can conclude that the BFS-based navigation described in Algorithm 1 is the safest controller in reaching the waypoint based on Fig. 11.6-11.8 compared to its counterpart DFS Algorithm 2 and BFSC Algorithm 3. However, our results show that the BFS is only slightly better than the BFSC one and that both are significantly better than the DFS-based navigation. The weak performance of the DSF-based navigation can be explained by its searching for the farthest nodes and computing a longer sequence of actions, which in the presence of randomly moving obstacles have little chance to play out completely.

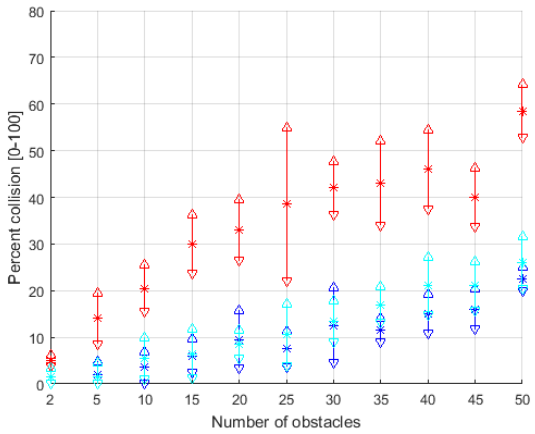


Figure 11.7: Percent of collisions vs. number of obstacles from 200 simulations with $\sigma_o = 0.0273$. The data points are for DFS- (red), BFS- (blue) and BFSC- (cyan) based navigation.

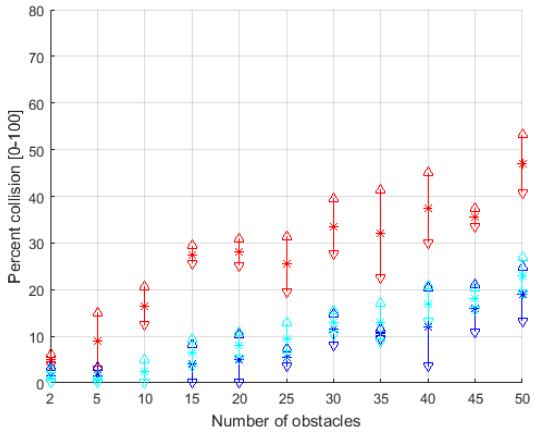


Figure 11.8: Percent of collisions vs. number of obstacles from 200 simulations with $\sigma_o = 0.0039$. The data points are for DFS- (red), BFS- (blue) and BFSC- (cyan) based navigation.

11.6 Conclusions

In this paper, we presented the feedback control approach to navigate a UAV, which is modeled as the turning rate-bounded Dubins vehicle, toward a waypoint while avoiding collision with multiple randomly moving obstacles. The approach is based on the stochastic optimal control for the vehicle, waypoint and a single obstacle with an associated value function, which is the solution of the Hamilton-Jacobi-Bellman (HJB) equation. We used the value function, evaluated it with respect to multiple obstacles and searched over available control actions to navigate the UAV.

For the feedback control that accounts for a single obstacle, we showed that it provides safety guarantees, i.e., the upper bound for the probability of collision. Moreover, we proved that we can compute that probability exactly. For the navigation around multiple obstacles, we proposed to search over sequences of control actions accounting for the largest value of the value function evaluated for the obstacles. We considered the following three types of searches over the action sequences: Depth-first-search (DFS), Breadth-first-search (BFS), and Breadth-first-search with constant control actions (BFSC). We compared the navigation based on the three types using stochastic simulations with multiple obstacles to conclude that the BFS and its variant BFSC provide a consistently better performance than DFS. While the BFS-based navigation performs slightly better than the BFSC one, the latter one can be preferred for the sake of using a simpler algorithm which can be computed more efficiently.

With this work, we presented the approach to synthesize the control for safe navigation among a large number of randomly moving obstacles. The approach uses the value function of the stochastic optimal controller for avoiding a single obstacle and searching over a sequence of actions to navigate around the obstacles. While our approach is useful in the area of UAV navigation, it may prove to be fruitful in other applications as well, including those in which a vehicle, or a robotic mechanism navigates toward a target set and has to avoid obstacles in its way.

Chapter 12

Concluding Remarks and Future Work

While each of the preceding papers contained conclusions on a specific problem or area of research, we provide here some brief overall remarks on the dissertation and directions for future work.

The work in this thesis started with one-on-one stochastic optimal control problems, including vehicle-to-vehicle and vehicle-to-goal (or waypoint) problems. Chapter 4 covered the topic of a single UAV entering the tail sector of another UAV whose navigation is unknown, and chapter 5 covered the topic of a single UAV avoiding to collide with an unpredictably moving obstacle while navigating towards a waypoint in minimum time. These two one-on-one problems serve as the building blocks and

inspiration for the work presented in the later chapters. The complexity of the underlining stochastic optimal control problem increased as we added more vehicles such as in chapters 6, 7 and 11, or added more obstacles, such as in chapters 8, 9 and 10. As the result of adding more vehicles or obstacles to the studied problem, we ended up with complex control problems due to the large number of states arising in the underlining stochastic optimal control problem that we have to deal with. Solving an optimal control problem with a large number of states might not be possible due to the so-called *curse of dimensionality* [137]. Moreover, we also had to account for the fact that the *number of states in the problems could vary with time* since the number of vehicles and obstacles could vary with time as well. In order to solve these complex stochastic optimal control problems, we decomposed the original stochastic optimal control problem into one-on-one vehicle-to-vehicle and vehicle-to-goal parts, and composed their solutions to solve the original complex problem. We used two types of compositions, a one-step or multi-step lookahead optimization and a Markov inequality switching approach. Though we did not provide guarantees for our heuristic methods to work in all possible scenarios, we provided guidelines and algorithms for checking if the method fails to find a solution. Many of these methods such as the Markov inequality switching have a solid mathematical analysis and proofs that provide guarantees for the method to work for the scenarios that are studied in this work.

At the end of the work in this thesis, we found that some of the studied questions

can be explored in more depth and that some of them opened new directions for future work. Some potential directions for future theoretical research include dealing with Markov Jumps and Levy Processes [42], as well as incorporating some decentralized control for multi-agent cooperative agent teams [19, 23]. On the implementation side, work towards developing specialized hardware and software for using the presented scalable control methods in UAV applications [163, 51, 36, 90, 156] would be likely fruitful research directions as well.

Appendix A

Two-Target Stochastic Kinematics Derivations

A.1 Derivation of dr

We start with the 2nd order Taylor expansion for r^2

$$\begin{aligned} 2rdr &= 2(x_R - x_B)d(x_R - x_B) + \frac{1}{2}2d(x_R - x_B)^2 \\ &\quad +2(y_R - y_B)d(y_R - y_B) + \frac{1}{2}2d(y_R - y_B)^2 \end{aligned}$$

and substitute dx_B, dx_R, dy_B and dy_R with the right-hand sides of (4.1)-(4.2) and (4.4)-(4.5) to obtain

$$\begin{aligned} 2rdr &= 2r\cos(\theta_B+\phi)[v_R\cos\theta_R - v_B\cos\theta_B]dt \\ &\quad +dx_R^2 - 2dx_Rdx_B + dx_B^2 \\ &\quad +2r\sin(\theta_B+\phi)[v_R sin\theta_R - v_B sin\theta_B]dt \\ &\quad +dy_R^2 - 2dy_Rdy_B + dy_B^2 \end{aligned}$$

Then, by the Îto calculus rule, we ignore any term with the order greater than dt, which results in

$$\begin{aligned} 2rdr &= 2r\cos(\theta_B+\phi)[v_R\cos\theta_R - v_B\cos\theta_B]dt \\ &\quad +2r\sin(\theta_B+\phi)[v_R\sin\theta_R - v_B\sin\theta_B]dt \end{aligned}$$

Dividing the last expression with r yields

$$\begin{aligned} dr &= \cos(\theta_B+\phi)[v_R\cos\theta_R - v_B\cos\theta_B]dt \\ &\quad +\sin(\theta_B+\phi)[v_R\sin\theta_R - v_B\sin\theta_B]dt \\ &= [v_e\cos(\theta_B+\phi-\theta_R) - v_B\cos(\theta_B+\phi-\theta_B)]dt \end{aligned}$$

and we finally obtain dr as

$$dr = [v_R\cos(\phi-\alpha) - v_B\cos(\phi)]dt \qquad \text{(A.1)}$$

A.2 Derivation of $d\alpha$

The difference of the heading angles is $\alpha = \theta_R - \theta_B$, therefore,

$$d\alpha = d\theta_R - d\theta_B$$

which after the substitution from the right-hand side of (4.3) and (4.6) results in

$$d\alpha = -u_B dt + \sigma_R dw \tag{A.2}$$

A.3 Derivation of $d\phi$

To derive $d\phi$, we start with

$$\begin{aligned} d(\theta_B + \phi) = & \frac{\partial}{\partial y_R}\left[\arctan\left(\frac{y_R - y_B}{x_R - x_B}\right)\right] dy_R \\ & + \frac{\partial}{\partial y_B}\left[\arctan\left(\frac{y_R - y_B}{x_R - x_B}\right)\right] dy_B \\ & + \frac{\partial}{\partial x_R}\left[\arctan\left(\frac{y_R - y_B}{x_R - x_B}\right)\right] dx_R \\ & + \frac{\partial}{\partial x_B}\left[\arctan\left(\frac{y_R - y_B}{x_R - x_B}\right)\right] dx_B \\ & + O(dt^2) \end{aligned}$$

Then, by the Îto calculus rule, we ignore any term with the order greater than dt, therefore,

$$\begin{aligned} d\theta_B + d\phi &= -\left(\frac{y_R - y_B}{(y_R - y_B)^2 + (x_R - x_B)^2}\right) dx_R \\ &+ \left(\frac{y_R - y_B}{(y_R - y_B)^2 + (x_R - x_B)^2}\right) dx_B \\ &+ \left(\frac{x_R - x_B}{(y_R - y_B)^2 + (x_R - x_B)^2}\right) dy_R \\ &- \left(\frac{x_R - x_B}{(y_R - y_B)^2 + (x_R - x_B)^2}\right) dy_B \end{aligned}$$

and the substitution of the right-hand side of expressions (4.1)-(4.2) and (4.4)-(4.5) results in

$$\begin{aligned} d\theta_B + d\phi &= \frac{1}{r}[-v_R \cos\theta_R \sin(\theta_B + \phi) \\ &\quad + v_B \cos\theta_B \sin(\theta_B + \phi) \\ &\quad + v_R \sin\theta_R \cos(\theta_B + \phi) \\ &\quad - v_B \sin\theta_B \cos(\theta_B + \phi)]dt \end{aligned}$$

Now we exploit trigonometric identities and $\alpha = \theta_R - \theta_B$ to obtain

$$d\theta_B + d\phi = \frac{1}{r}\left[v_R \sin(\alpha - \phi) + v_B \sin\phi\right] dt$$

and, since $d\theta_B = u_B dt$, we finally obtain

$$d\phi = \left[-u_B + \frac{1}{r}\left[-v_R \sin(\phi - \alpha) + v_B \sin\phi\right]\right] dt$$

Appendix B

Single Obstacle Stochastic Kinematics Derivations

B.1 Derivation of dr_o

Starting with (5.22) and applying the 2nd order Taylor series expansion for r_o^2

$$\begin{aligned} 2r_o dr_o = {} & 2(x_o - x)dx_o + \frac{1}{2}2(dx_o)^2 - 2(x_o - x)dx + \frac{1}{2}2(dx)^2 \\ & + 2(y_o - y)dy_o + \frac{1}{2}2(dy_o)^2 - 2(y_o - y)dy + \frac{1}{2}2(dy)^2 \\ & - \frac{1}{2}2dxdx_o - \frac{1}{2}2dydy_o \end{aligned}$$

Then we substitute dx, dy, dx_o and dy_o with the right-hand sides of (5.1), (5.2), (5.4), (5.5), and by the Itô calculus rule, we ignore terms with orders greater than dt to obtain

$$\begin{aligned} 2r_o dr_o = {} & 2r_o \cos(\phi_o + \theta)\sigma dw_x + \frac{1}{2}\sigma^2 \cos^2(\phi_o + \theta)dt \\ & 2r_o \cos(\phi_o + \theta)(-v\cos\theta)dt \\ & 2r_o \sin(\phi_o + \theta)\sigma dw_y + \frac{1}{2}\sigma^2 \sin^2(\phi_o + \theta)dt \\ & 2r_o \sin(\phi_o + \theta)(-v\sin\theta)dt \end{aligned}$$

Substituting the sum and difference formula for the cos and grouping terms, we obtain (5.6), i.e.,

$$dr_o = \left(-v\cos\phi_o + \frac{\sigma^2}{2r_o}\right)dt + \sigma dw_{r_o}$$

B.2 Derivation of $d\phi_o$

To derive ϕ_o, we start with

$$\begin{aligned} d(\phi_o + \theta) = {} & \frac{\partial}{\partial y_o}\arctan\left(\frac{y_o - y}{x_o - x}\right)dy_o \\ & + \frac{\partial}{\partial y}\arctan\left(\frac{y_o - y}{x_o - x}\right)dy \\ & + \frac{\partial}{\partial x_o}\arctan\left(\frac{y_o - y}{x_o - x}\right)dx_o \\ & + \frac{\partial}{\partial x}\arctan\left(\frac{y_o - y}{x_o - x}\right)dx \\ & + O(dt^2) \end{aligned}$$

Then, by the Itô calculus rule, we ignore any term with the order greater than dt, therefore,

$$\begin{aligned} d\phi_o + d\theta = & \frac{x_o - x}{(x_o - x)^2 + (y_o - y)^2} dy_o \\ & - \frac{x_o - x}{(x_o - x)^2 + (y_o - y)^2} dy \\ & - \frac{y_o - y}{(x_o - x)^2 + (y_o - y)^2} dx_o \\ & + \frac{y_o - y}{(x_o - x)^2 + (y_o - y)^2} dx \end{aligned}$$

The substitution of the right-hand side of expressions (5.1), (5.2), (5.4) and (5.5) results in

$$\begin{aligned} d\phi_o + d\theta = & \frac{r_o \cos(\phi_o + \theta)\, \sigma}{r_o^2} dw_y \\ & - \frac{r_o \cos(\phi_o + \theta)\, v \sin\theta}{r_o^2} dt \\ & - \frac{r_o \sin(\phi_o + \theta)\, \sigma}{r_o^2} dw_x \\ & + \frac{r_o \sin(\phi_o + \theta)\, v \cos\theta}{r_o^2} dt \end{aligned}$$

Now, we exploit trigonometric identities to obtain

$$d\phi_o + d\theta = \frac{v}{r_o} \sin\phi_o dt + \frac{\sigma}{r_o} dw_{\phi_o}$$

and since $d\theta = u dt$, we finally obtain (5.7), which is

$$d\phi_o = \left(\frac{v}{r_o} \sin\phi_o - u \right) dt + \left(\frac{\sigma}{r_o} \right) dw_{\phi_o}$$

Appendix C

Supplementary Video Snapshots for Chapter 7

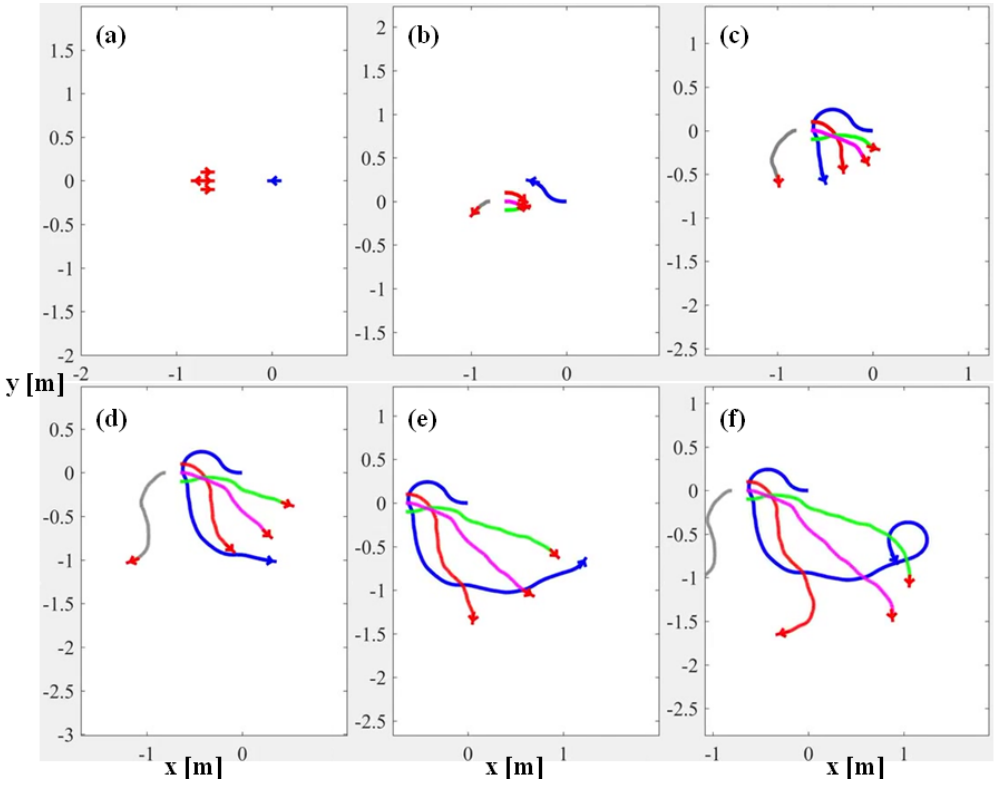


Figure C.1: Simulation result of Fig. 7.6. The B agent trajectory is colored blue and the R_1 to R_4 trajectories are colored red, green, magenta and gray, respectively.

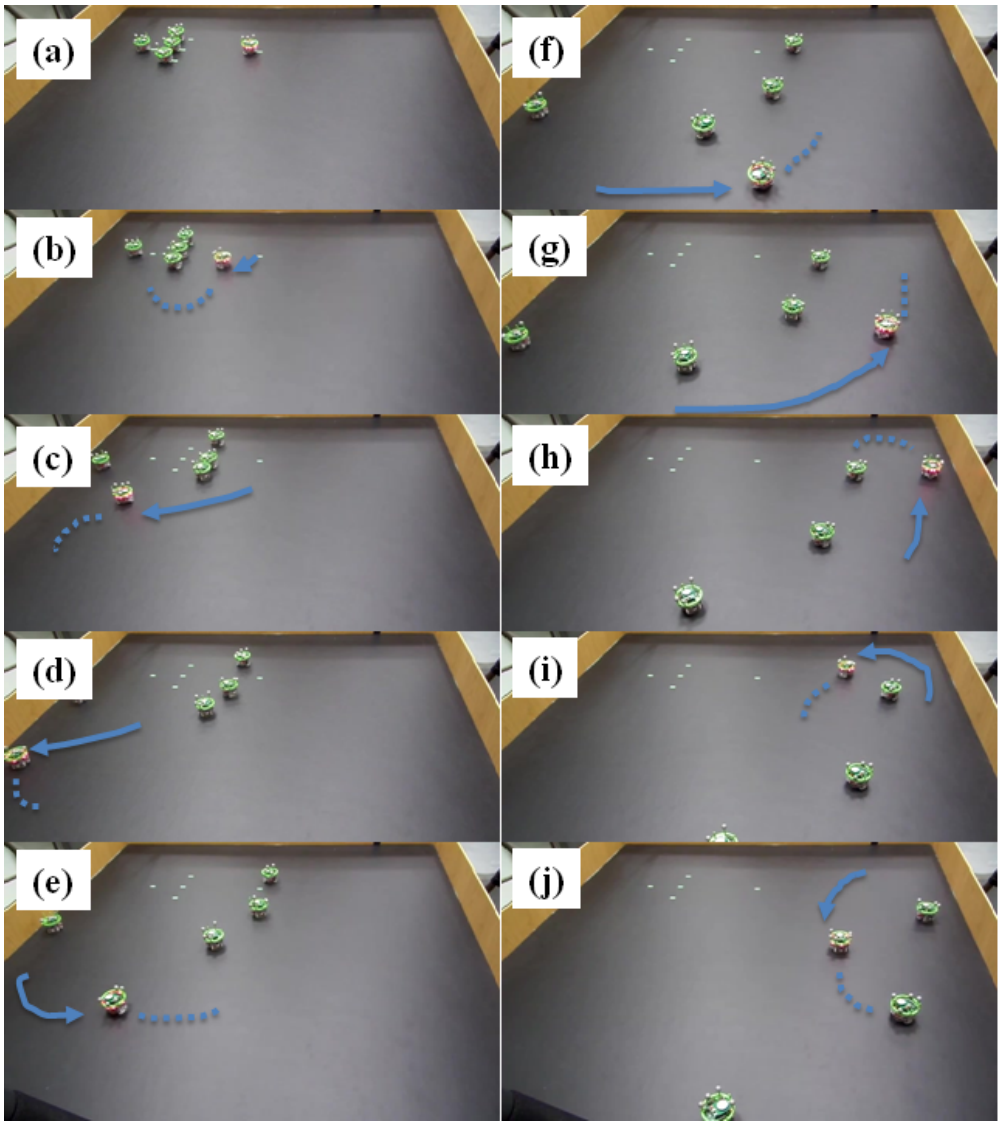


Figure C.2: Robot experiment result of Fig. 7.9 with five e-puck robots. Each robot has a unique configuration of infrared reflecting markers (green rings with silver spheres) tracked by a motion capturing system with four Bonita 10 Vicon cameras. The robots move from frames a-j.

Appendix D

Supplementary Video Snapshots for Chapter 9

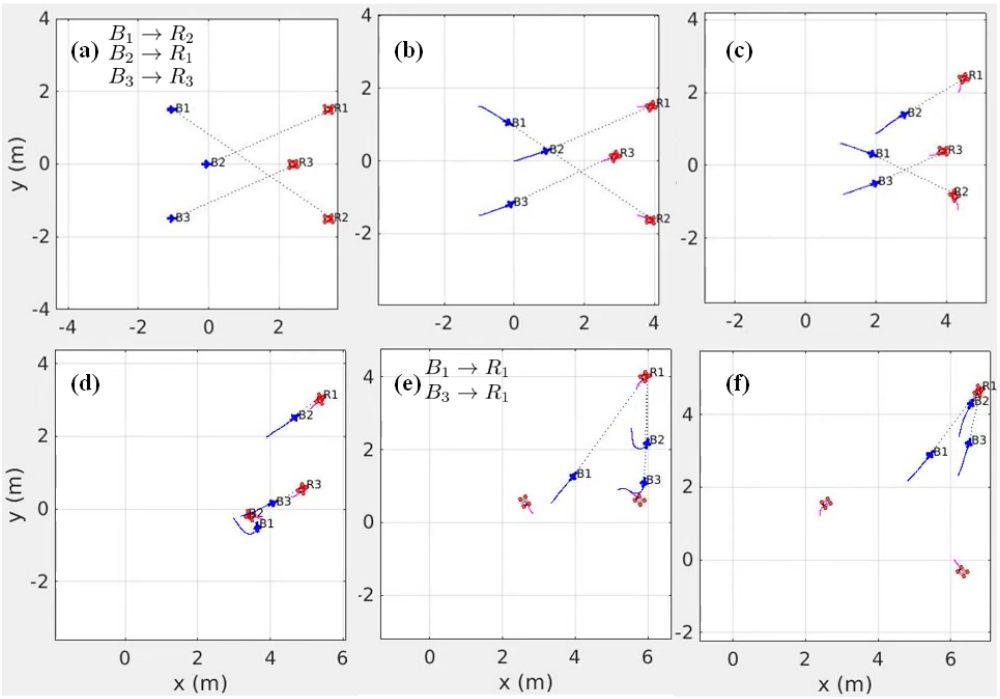


Figure D.1: The initial configurations of blue and red agents: the blue agents are presented with arrow-like symbols depicting aircraft and the red agents with rectangular symbols depicting ground vehicles, i.e., cars. For 3 blues and 3 reds

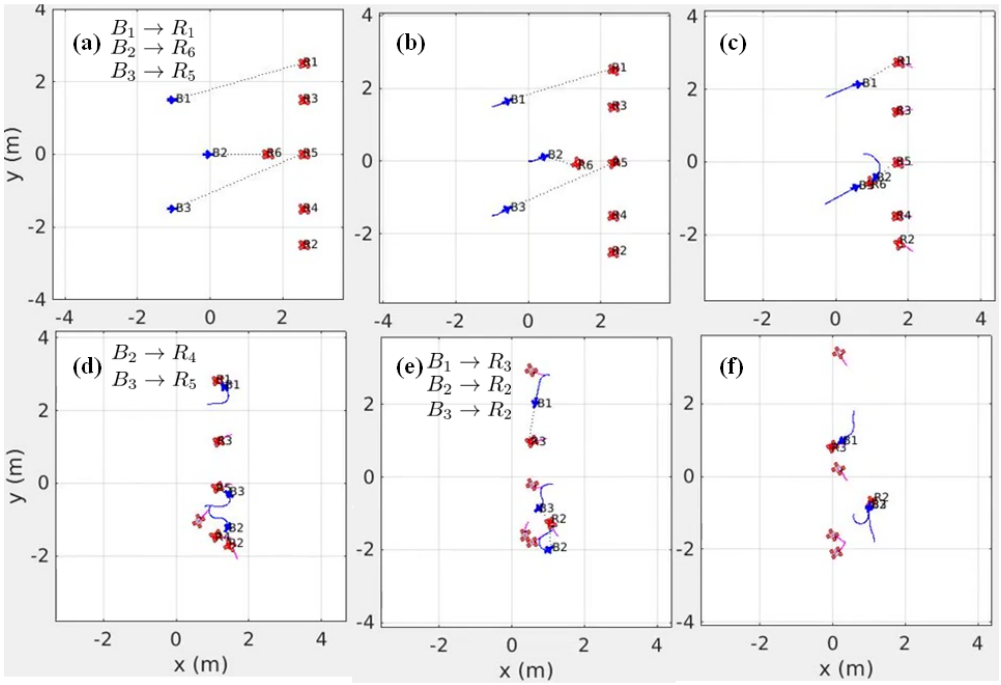


Figure D.2: The initial configurations of blue and red agents: the blue agents are presented with arrow-like symbols depicting aircraft and the red agents with rectangular symbols depicting ground vehicles, i.e., cars. For 3 blues and 6 reds

Appendix E

Supplementary Material for Chapter 11

Numerical approximation method for stochastic control

The solution of (11.32) can be computed using a locally consistent Markov-chain discretization of the HJB equation [94]. The discretization yields a Markov-chain with control dependent transition probabilities and converts (11.32) into a dynamic programming problem which can be solved over a discrete space using so-called value iterations [175]. The value iterations result in a discrete approximation of the value function V^h and optimal control u^{*h}, where both are in the form of a four-dimensional lookup table. The superscript h indicates that the value function and control are computed for the

discretized problem.

To discretize (11.32) in the state-space [94], we use discrete steps Δr_w, $\Delta\phi_w$, Δr_o, and $\Delta\phi_o$ for the discretization of r_w, ϕ_w, r_o, and ϕ_o, respectively. The upwind discrete approximations of derivatives in (11.32) are

$$\frac{\partial V}{\partial \tilde{x}_i} \approx \frac{b^+_{\tilde{x}^h_i}}{\Delta\tilde{x}_i}\left(V(\tilde{x}^h_i+\Delta\tilde{x}_i)-V(\tilde{x}^h_i)\right)-\frac{b^-_{\tilde{x}^h_i}}{\Delta\tilde{x}_i}\left(V(\tilde{x}^h_i)-V(\tilde{x}^h_i-\Delta\tilde{x}_i)\right) \tag{.1}$$

$$\frac{\partial^2 V}{\partial \tilde{x}_i^2} \approx \frac{n^2_{\tilde{x}_i}}{2\Delta\tilde{x}_i^2}\left(V(\tilde{x}^h_i+\Delta\tilde{x}_i)-V(\tilde{x}^h_i)\right)-\frac{n^2_{\tilde{x}_i}}{2\Delta\tilde{x}_i^2}\left(V(\tilde{x}^h_i)-V(\tilde{x}^h_i-\Delta\tilde{x}_i)\right), \tag{.2}$$

where $b^{\pm}_{\tilde{x}^h_i}=\max[0,\pm b_{\tilde{x}^h_i}]$ and $n_{\tilde{x}_i}$ can be one of expressions n_{r_o}, n_{ϕ_o} defined in (11.15)-(11.16). The discrete step $\Delta\tilde{x}_i$ can be of one of the discrete state-space steps $\Delta r_w, \Delta r_o, \Delta\phi_w, \Delta\phi_o$. The superscript h indicates terms that are evaluated at the points of the discretized state-space $r_w^{h+1}-r_w^h=\Delta r_w$, $\phi_w^{h+1}-\phi_w^h=\Delta\phi_w$, $r_o^{h+1}-r_o^h=\Delta r_o$, and $\phi_o^{h+1}-\phi_o^h=\Delta\phi_o$. After the substitution of (11.28), (.2)-(.2) in (11.32), we move all the terms that include $V(r_w^h,\phi_w^h,r_o^h,\phi_o^h)$ to the left side of expression (11.32) to obtain the dynamic programming expression

$$V^h(\tilde{x}_i)=\min_u\left\{\Delta t^h_{\tilde{x}_i,u}+\sum_{\Delta\tilde{x}_i} p^{\pm}_{\Delta\tilde{x}_i,u}V^h(\tilde{x}_i+\Delta\tilde{x}_i)\right\}, \tag{.3}$$

where

$$p^{\pm}_{\Delta\tilde{x}_i,u}=\Delta t^h_u\left(\frac{b^{\pm}_{\tilde{x}_i}}{\Delta\tilde{x}_i}+\frac{n_{\tilde{x}_i}}{(2\Delta\tilde{x}_i^2)}\right) \tag{.4}$$

are the discrete Markov-chain transition probabilities $p^{\pm}_{\Delta\tilde{x}_i,u}$ (.4) from the points $\tilde{x}_i\pm\Delta\tilde{x}_i=(r_w^h\pm\Delta r_w,\phi_w^h\pm\Delta\phi_w,r_o\pm\Delta r_o,\phi_o^h\pm\Delta\phi_o)$ of the discrete state-space to the point

$\tilde{x}_i = (r_w^h, \phi_w^h, r_o^h, \phi_o^h)$ and contain the implicit time interpolation interval [94] as

$$\Delta t_{\tilde{x}_i,u}^h = 1 \Bigg/ \left(\sum_{\Delta \tilde{x}_i} \frac{|b_{\tilde{x}}|}{\Delta \tilde{x}_i} + \frac{(n_{\tilde{x}_i})^2}{(\Delta \tilde{x}_i)^2} \right), \tag{.5}$$

where $|b_{\tilde{x}_i}^h| = b_{\tilde{x}_i}^+ + b_{\tilde{x}_i}^-$.

Expression (.3) is the discrete version of (11.32) and the discrete approximation V^h of the value function V can be solved numerically using value iterations [175] starting from an initial guess for the $V^h(\tilde{x}_i)$ values.

Our discrete solution V^h is solved on a discrete state-space domain $\mathscr{X}^h$ as

$$\mathscr{X}^h = \{[R_{min}, R_{max}]^2 \times [-\pi, \pi - \Delta\phi_o]\} \times [-\pi, \pi - \Delta\phi_w]\}, \tag{.6}$$

which is the set bounded by the minimal R_{min} and maximal R_{max} distances. The discrete state-space $\mathscr{X}^h$ is an approximation to the original state-space given in (11.17) for the single obstacle scenario, i.e., $N = 1$. Since in our problem formulation, the angles ϕ_w and ϕ_o have a full 2π range, the computational domain is periodic, i.e., the pairs of points $(r_w^h, -\pi, r_o^h, \phi_o^h)$, $(r_w^h, \pi - \Delta\phi_w, r_o^h, \phi_o^h)$ for ϕ_w^h are next to each other and similarly for the points $(r_w^h, \phi_w^h, r_o^h, -\pi)$, $(r_w^h, \phi_w^h, r_o^h, \pi - \Delta\phi_o)$ for ϕ_o^h.

Search Algorithms

Algorithm 1. *BFS_MultiStep(X_0, n_{max}):*

1: *initialize Q be an empty queue*

2: *initialize U_{seq} := an empty sequence*

3: *Q.enqueue*($[X_0, U_{seq}]$)

4: ▷ *Main loop*

5: ***for*** *Each step* $n \in [1 : n_{max}]$ ***do***

6: $[\overline{X}_n, U_{seq}]$:= *Q.dequeue()*

7: ***for*** *Each discrete control* $u^h \in [-u_{max} : \Delta u : u_{max}]$ ***do***

8: $\overline{X}_n := \sum_{\forall \tilde{y}^h \text{ neighbors of } \overline{X}_n} \{p^{\pm}_{\Delta \tilde{x}^h, u^h} \tilde{y}^h\}$

9: $U_{seq} := \{U_{seq}, u^h\}$ ▷ *prepend operation*

10: *Q.enqueue*($[\overline{X}_n, U_{seq}]$)

11: ▷ *Check if feasible solution*

12: ***if*** $V^{upper}(\overline{X}_n) < V^{upper}(X_0)$ ***then***

13: *return* U_{tmp} ▷ *Found solution*

14: ***end if***

15: ***end for***

16: ***end for***

17: *return empty sequence* ▷ *Fail condition*

Algorithm 2. *DFS_MultiStep*(X_0, n_{max})*:*

1: *initialize S be an empty stack*

2: *initialize* U_{seq} := *an empty sequence*

3: *S.push*($[X_0, U_{seq}]$)

4: ▷ *Main loop*

5: ***for*** *Each step* $n \in [1 : n_{max}]$ ***do***

6: $[\overline{X}_n, U_{seq}]$:= *S.pop()*

7: ***for*** *Each discrete control* $u^h \in [-u_{max} : \Delta u : u_{max}]$ ***do***

8: $\overline{X}_n := \sum_{\forall \tilde{y}^h \text{ neighbors of } \overline{X}_n} \{p^{\pm}_{\Delta \tilde{x}^h, u^h} \tilde{y}^h\}$

9: $U_{seq} := \{U_{seq}, u^h\}$ ▷ *prepend operation*

10: *S.push(*$[\overline{X}_n, U_{seq}]$*)*

11: ▷ *Check if feasible solution*

12: ***if*** $V^{upper}(\overline{X}_n) < V^{upper}(X_0)$ ***then***

13: *return* U_{tmp} ▷ *Found solution*

14: ***end if***

15: ***end for***

16: ***end for***

17: *return empty sequence* ▷ *Fail condition*

Algorithm 3. *BFSC_MultiStep(*X_0, n_{max}*):*

1: *initialize* U_{seq} := *an empty sequence*

2: ▷ *Main loop*

3: ***for*** *Each step* $n \in [1 : n_{max}]$ ***do***

4: *set flag := true*

5: ***for*** *Each discrete control* $u^h \in [-u_{max} : \Delta u : u_{max}]$ ***do***

6: $\overline{X}_n := \sum_{\forall \tilde{y}^h \text{ n step neighbors of } X_0} \{p^{\pm}_{\Delta \tilde{x}^h, u^h} \tilde{y}^h\}$

```
7:          if flag is true then
8:              set V_min := V^upper(X̄_n)
9:              set u_next := u^h
10:             set flag := false
11:         else
12:             if V_min > V^upper(X̄_n) then
13:                 set V_min := V^upper(X̄_n)
14:                 set u_next := u^h
15:             end if
16:         end if
17:     end for
18:                                          ▷ Check if feasible solution
19:     if V_min < V^upper(X_0) then
20:         for n times do
21:             U_seq := {U_seq, u_next}     ▷ prepend operation
22:         end for
23:         return U_seq                     ▷ Found solution
24:     end if
25: end for
26: return empty sequence                    ▷ Fail condition
```

Remark: The maximum number of nodes to search at the step level n for p possible control actions is p^n. For the search up to the nth step, it is

$$\sum_{j=1}^{n} p^j = p + p^2 + p^3 + \ldots + p^n = p\frac{(1-p^n)}{(1-p)} \tag{.7}$$

where the step level is $n \in \{1,2,3,\ldots\}$ and is bounded by the max step level, i.e., $n \leq n_{max} < \infty$.

Bibliography

[1] A. Abate, M. Prandini, J. Lygeros, and S. Sastry. Probabilistic reachability and safety for controlled discrete time stochastic hybrid system. *Automatica*, 44:2724–2734, March 2008.

[2] Martin Aigner and Michael Fromme. A game of cops and robbers. *Discrete Applied Mathematics*, 8(1):1–12, 1984.

[3] Mazen Alamir. *A Pragmatic Story of Model Predictive Control: Self Contained Algorithms and Case-studies*. CreateSpace Independent Publishing Platform, 2013.

[4] Javier Alonso-Mora, Andreas Breitenmoser, Martin Rufli, Paul Beardsley, and Roland Siegwart. *Optimal Reciprocal Collision Avoidance for Multiple Non-Holonomic Robots*, pages 203–216. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[5] Ignacio Alzugaray, Lucas Teixeira, and Margarita Chli. Short-term uav path-

planning with monocular-inertial slam in the loop. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2739–2746. IEEE, 2017.

[6] R. Anderson and D. Milutinović. A stochastic approach to dubins feedback control for target tracking. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3917–3922, September 2011.

[7] R. P. Anderson, E. Bakolas, D. Milutinović, and P. Tsiotras. The Markov-Dubins problem in the presence of a stochastic drift field. *Decision and Control (CDC)*, Dec. 2012.

[8] R. P. Anderson, E. Bakolas, D. Milutinović, and P. Tsiotras. Optimal feedback guidance of a small aerial vehicle in a stochastic wind. *Journal of Guidance, Control, and Dynamics*, 36(4):975–985, 2013.

[9] R. P. Anderson and D. Milutinović. Dubins vehicle tracking of a target with unpredictable trajectory. In *Proc. 4th ASME Dynamic Syst. Control Conf.*, pages 675–682, Arlington, VA, USA, 2011.

[10] R. P. Anderson and D. Milutinović. Anticipating stochastic observation loss during optimal target tracking by a small aerial vehicle. In *Proc. Int. Conf. Unmanned Aircraft Syst. (ICUAS)*, pages 278—-287, Atlanta, GA, USA, 2013.

[11] Ross Anderson and Dejan Milutinović. Anticipating stochastic observation loss during optimal target tracking by a small aerial vehicle. In *Proceedings 2013*

International Conference on Unmanned Aircraft Systems (ICUAS), pages 278–287, 2013.

[12] Ross P. Anderson and Dejan Milutinović. A stochastic approach to dubins vehicle tracking problems. *IEEE Transactions on Automatic Control*, 59(10):2801–2806, 2014.

[13] P. Angelov. *Sense and Avoid in UAS: Research and Applications*. John Wiley & Sons, 2012.

[14] Panos J Antsaklis, James A Stiver, and Michael Lemmon. Hybrid system modeling and autonomous control systems. In *Hybrid systems*, pages 366–392. Springer, 1992.

[15] M. D. Ardema, M. Heymann, and N. Rajan. Combat games. *Journal of Optimization Theory and Applications*, 46(4):391–398, 1985.

[16] R. C. Arkin. *Behavior-based Robotics*. MIT press, 1998.

[17] Rajesh Kumar Arora. *Optimization: algorithms and applications*. Chapman and Hall/CRC, 2019.

[18] T. W. Athan and P. Y. Papalambros. A Note on Weighted Criteria Methods for Compromise Solutions in Multi-Objective Optimization. *Engineering Optimization*, 27(2), Apr 1996.

[19] Md Ali Azam, Hans D Mittelmann, and Shankarachary Ragi. Uav formation shape control via decentralized markov decision processes. *Algorithms*, 14(3):91, 2021.

[20] Andrea Bajcsy, Sylvia L Herbert, David Fridovich-Keil, Jaime F Fisac, Sampada Deglurkar, Anca D Dragan, and Claire J Tomlin. A scalable framework for real-time multi-robot, multi-human collision avoidance. In *2019 international conference on robotics and automation (ICRA)*, pages 936–943. IEEE, 2019.

[21] E. Bakolas and P. Tsiotras. Optimal Synthesis of the Asymmetric Sinistral/Dextral Markov-Dubins Problem. *Journal of Optimization Theory and Applications*, 150(2):233–250, 2011.

[22] A. J. Barry, A. Majumdar, and R. Tedrake. Safety verification of reactive controllers for uav flight in cluttered environments using barrier certificates. *IEEE International Conference on Robotics and Automation*, May 2012.

[23] Salvatore Rosario Bassolillo, Egidio D'Amato, Immacolata Notaro, Luciano Blasi, and Massimiliano Mattei. Decentralized mesh-based model predictive control for swarms of uavs. *Sensors*, 20(15):4324, 2020.

[24] R. W. Beard, T. W. McLain, M. A. Goodrich, and E. P. Anderson. Coordinated Target Assignment and Intercept for Unmanned Air Vehicles. *IEEE Transactions on Robotics and Automation*, 18(6):911–922, 2002.

[25] Randal W. Beard and Timothy W. McLain. *Small Unmanned Aircraft: Theory and Practice*. Princeton University Press, 2012.

[26] J. S. Bellingham, M. Tillerson, M. Alighanbari, and J. P. How. Cooperative path planning for multiple UAVs in dynamic and uncertain environments. *Proceedings of the 41st IEEE Conference on Decision and Control*, 3:2816–2822, 2002.

[27] Sarthak Bhagat and P.B. Sujit. UAV target tracking in urban environments using deep reinforcement learning. In *Proceedings of 2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 694–701, 2020.

[28] Fang Bin, Feng XiaoFeng, and Xu Shuo. Research on cooperative collision avoidance problem of multiple uav based on reinforcement learning. In *2017 10th International Conference on Intelligent Computation Technology and Automation (ICICTA)*, pages 103–109. IEEE, 2017.

[29] Inc. Bluetooth SIG. Bluetooth technology website. `https://www.bluetooth.com/`.

[30] U. Borrmann, L. Wang, A. D. Ames, and M. Egerstedt. Control Barrier Certificates for Safe Swarm Behavior. *Analysis and Design of Hybrid Systems (ADHS)*, 48(27):68–73, Oct. 2015.

[31] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[32] A. Brezoescu, R. Lozano, and P. Castillo. Lyapunov-based trajectory tracking controller for a fixed-wing unmanned aerial vehicle in the presence of wind. *International Journal of Adaptive Control and Signal Processing*, 29(3):372–384, 2015.

[33] Roger W Brockett. Parametrically stochastic linear differential equations. In *Stochastic Systems: Modeling, Identification and Optimization, I*, pages 8–21. Springer, 1976.

[34] R. Burkard, M. Dell'Amico, and S. Martello. *Assignment Problems: Revised Reprint*. SIAM: Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2009. Chapter 4: Linear Assignment Problem.

[35] Andrew Butterfield, Gerard Ekembe Ngondi, and Anne Kerr. *A dictionary of computer science*. Oxford University Press, 2016.

[36] PR Chandler and Meir Pachter. Research issues in autonomous control of tactical uavs. In *Proceedings of the 1998 American Control Conference. ACC (IEEE Cat. No. 98CH36207)*, volume 1, pages 394–398. IEEE, 1998.

[37] Hamidreza Chitsaz and Steven M LaValle. Time-optimal paths for a dubins airplane. In *2007 46th IEEE conference on decision and control*, pages 2379–2384. IEEE, 2007.

[38] J. Choi and D. Milutinović. Tips on stochastic optimal feedback control and bayesian spatio-temporal models: Applications to robotics. *Journal of Dynamic Systems, Measurement and Control*, 137:pp. 030801, 2015.

[39] Jongeun Choi and Dejan Milutinović. Tips on Stochastic Optimal Feedback Control and Bayesian Spatiotemporal Models: Applications to Robotics. *Journal of Dynamic Systems, Measurement, and Control*, 137(3), 10 2014. 030801.

[40] H. Choset. Coverage for robotics – A survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31(1):113–126, October 2001.

[41] V. Cichella, T. Marinho, D. Stipanović, N. Hovakimyan, I. Kaminer, and A. Trujillo. Collision Avoidance Based on Line-of-Sight Angle. *Intelligent & Robotic Systems*, March 2018.

[42] Erhan Cinlar and E. oCınlar. *Probability and stochastics*, volume 261. Springer, 2011.

[43] Maria C Consiglio, James P Chamberlain, Cesar A Munoz, and Keith D Hoffler. Concepts of integration for uas operations in the nas. *International Congress of the Aeronautical Sciences (ICAS)*, 2012.

[44] Navid Dadkhah and Berenice Mettler. Survey of motion planning literature in the presence of uncertainty: Considerations for uav guidance. *Journal of Intelligent & Robotic Systems*, 65(1):233–246, 2012.

[45] Jose B. Cruz Jr. Dongxu Li and Corey J. Schumacher. Stochastic multi-player pursuit-evasion differential games. *International Journal of Robust and Nonlinear Control*, 18(6):218–247, May 2008.

[46] Joseph L. Doob. *Stochastic Processes*. New York: Wiley, 1953.

[47] L. E. Dubins. On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents. *American Journal of Mathematics*, 79(3):497–516, 1957.

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

[49] T Eisele. Nonexistence and nonuniqueness of open-loop equilibria in linear-quadratic differential games. *Journal of Optimization Theory and Applications*, 37(4):443–468, 1982.

[50] J. Eklund, J. Sprinkle, H. Kim, and S. Sastry. Implementing and testing a nonlinear model predictive tracking controller for aerial pursuit/evasion games on a fixed wing aircraft. In *2005 American Control Conference (ACC)*, volume 3, page 1509–1514, 2005.

[51] Stanislav Emel'yanov, Dmitry Makarov, Aleksandr I. Panov, and Konstantin Yakovlev. Multilayer cognitive architecture for uav control. *Cognitive Systems*

Research, 39:58–72, 2016. From human to artificial cognition (and back): new perspectives of cognitively inspired AI systems.

[52] I. Exarchos, P. Tsiotras, and M. Pachter. UAV Collision Avoidance based on the Solution of the Suicidal Pedestrian Differential Game. *AIAA Guidance, Navigation, and Control Conference*, 2016.

[53] Wael Farag. Robot arm navigation using deep deterministic policy gradient algorithms. *Journal of Experimental & Theoretical Artificial Intelligence*, pages 1–11, 2022.

[54] A. Festa and R. B. Vinter. Decomposition of Differential Games with Multiple Targets. *Journal of Optimization Theory and Applications*, pages 849–875, 2016.

[55] Paolo Fiorini and Zvi Shiller. Motion planning in dynamic environments using velocity obstacles. *The international journal of robotics research*, 17(7):760–772, 1998.

[56] W. H. Fleming and H. M. Soner. *Controlled Markov Processes and Viscosity Solutions*. Springer Science+Business Media, 2006.

[57] Wendell H. Fleming and Raymond W. Rishel. *Deterministic and Stochastic Optimal Control*. Springer Science & Business Media, December 2012.

[58] Wendell H Fleming and Halil Mete Soner. *Controlled Markov processes and viscosity solutions*, volume 25. Springer Science & Business Media, 2006.

[59] Gene F Franklin, J David Powell, Abbas Emami-Naeini, and J David Powell. *Feedback control of dynamic systems*, volume 4. Prentice hall Upper Saddle River, NJ, 2002.

[60] Zachariah E Fuchs, Alexander Von Moll, and David Casbeer. Engage or retreat differential game with n-targets and distributed defensive assets. In *2021 IEEE Conference on Control Technology and Applications (CCTA)*, pages 386–393. IEEE, 2021.

[61] C. Gardiner. *Stochastic Methods: A Handbook for the Natural and Social Sciences*. Springer, 2009.

[62] J. Ge, L. Tang, J. Reimann, and G. Vachtsevanos. Hierarchical Decomposition Approach for Pursuit-Evasion Differential Game with Multiple Players. *IEEE Aerospace Conference*, 2006.

[63] Wayne M. Getz and G. Leitmann. Qualitative Differential Games with Two Targets. *Journal of Mathematical Analysis and Applications*, 68:421–430, January 1979.

[64] Wayne M. Getz and Meir Pachter. Capturability in a Two-Target ‘’Game of Two Cars”. *Journal of Guidance and Control*, 4(1):15–22, 1981.

[65] Christopher Geyer. Active target search from UAVs in urban environments. In *Proceedings of 2008 IEEE International Conference on Robotics and Automation*, pages 2366–2371. IEEE, 2008.

[66] Jared Giesbrecht. Global path planning for unmanned ground vehicles. Technical report, DEFENCE RESEARCH AND DEVELOPMENT SUFFIELD (ALBERTA), 2004.

[67] D. T. Gillespie. Exact Stochastic Simulation of Coupled Chemical Reactions. *Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

[68] Jacob R Goodman and Leonardo J Colombo. Collision avoidance of multiagent systems on riemannian manifolds. *SIAM Journal on Control and Optimization*, 60(1):168–188, 2022.

[69] W. Grimm and K. H. Well. Modeling Air Combat as Differential Game Recent Approaches and Future Requirements. *In Hamalainen, R. P., H. Ehtamo*, pages 1–13, 1991.

[70] J. Guldner and V. I. Utkin. Sliding Mode Control for Gradient Tracking and Robot Navigation Using Artificial Potential Fields. *IEEE Transactions on Robotics and Automation*, 11(2):247–254, April 1995.

[71] Suraj G Gupta, Mangesh M Ghonge, Pradip M Jawandhiya, et al. Review of

unmanned aircraft system (uas). *International journal of advanced research in computer engineering & technology (IJARCET)*, 2(4):1646–1658, 2013.

[72] Shlomi Hacohen, Shraga Shoval, and Nir Shvalb. Navigation function for multi-agent multi-target interception missions. 2022.

[73] A. Hashemi, D. W. Casbeer, and D. Milutinović. Scalable value approximation for multiple target tail-chase with collision avoidance. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 2543–2548, Dec 2016.

[74] A. Hashemi, D. W. Casbeer, and D. Milutinović. Scalable Value Approximation for Multiple Target Tail-Chase with Collision Avoidance. *Decision and Control (CDC)*, December 2016.

[75] Udai Hassein. Deveopment of framework for in-vehicle passing collison warning system considering driver characteristics, 2017.

[76] Sikha Hota and Debasish Ghose. Optimal path planning for an aerial vehicle in 3d space. In *49th IEEE Conference on Decision and Control (CDC)*, pages 4902–4907. IEEE, 2010.

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

[78] Xinting Hu, Bizhao Pang, Fuqing Dai, and Kin Huat Low. Risk assessment model for uav cost-effective path planning in urban environments. *IEEE Access*, 8:150162–150173, 2020.

[79] H. Huang, J. Ding, W. Zhang, and C. J. Tomlin. Automation-Assisted Capture-the-Flag: A Differential Game Approach. *IEEE Transactions on Control Systems Technology*, 23(3):1014–1028, May 2015.

[80] Myung Hwangbo, James Kuffner, and Takeo Kanade. Efficient two-phase 3d motion planning for small fixed-wing uavs. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 1035–1041. IEEE, 2007.

[81] R. Isaacs. *Games of Pursuit*. Santa Monica, CA: RAND Corporation, 1951.

[82] R. Isaacs. *Differential Games*. Wiley, New York, NY, 1965.

[83] V. Isler, D. Sun, and S. Sastry. Roadmap Based Pursuit-Evasion and Collision Avoidance. *robotics science and systems*, 2005.

[84] B. W. Israelsen, N. Ahmed, K. Center, R. Green, and W. Bennett Jr. Adaptive simulation-based training of ai decision-makers using bayesian optimization. *https://arxiv.org/abs/1703.09310*, 2017.

[85] Yusong Jiao, Juan Du, Xinmin Wang, and Rong Xie. H∞ state feedback con-

trol for uav maneuver trajectory tracking. In *2010 International Conference on Intelligent Control and Information Processing*, pages 253–257, 2010.

[86] Yeonsik Kang and J. Karl Hedrick. Linear tracking for a fixed-wing uav using nonlinear model predictive control. *IEEE Transactions on Control Systems Technology*, 17(5):1202–1210, 2009.

[87] Hassan K Khalil. *Nonlinear Systems*. Pearson Higher Ed, 2014.

[88] J. Kim and P. K. Khosla. Real-Time Obstacle Avoidance Using Harmonic Potential Functions. *IEEE Transactions on Robotics and Automation*, June 1992.

[89] Mykel J Kochenderfer and Tim A Wheeler. *Algorithms for optimization*. Mit Press, 2019.

[90] TJ Koo, David Hyunchul Shim, O Shakernia, B Sinopoli, Y Ma, F Hoffmann, and S Sastry. Hierarchical hybrid system design on berkeley uav. *International Aerial Robotics Competition*, 1998.

[91] Mangal Kothari, Ian Postlethwaite, and Da-Wei Gu. Multi-uav path planning in obstacle rich environments using rapidly-exploring random trees. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 3069–3074. IEEE, 2009.

[92] Hanna Kurniawati, Yanzhu Du, David Hsu, and Wee Sun Lee. Motion planning

under uncertainty for robotic tasks with long time horizons. *The International Journal of Robotics Research*, 30(3):308–323, 2011.

[93] H. J. Kushner. *Stochastic Stability and Control*. New York: Academic, 1967.

[94] Harold J. Kushner and Paul Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time*, volume 24 of *Stochastic Modelling and Applied Probability*. Springer New York, New York, NY, 2001.

[95] Janusz Kusyk, M Umit Uyar, Kelvin Ma, Eltan Samoylov, Ricardo Valdez, Joseph Plishka, Sagor E Hoque, Giorgio Bertoli, and Jeffrey Boksiner. Artificial intelligence and game theory controlled autonomous uav swarms. *Evolutionary Intelligence*, pages 1–18, 2020.

[96] Huibert Kwakernaak and Raphael Sivan. *Linear Optimal Control Systems*. Wiley-Interscience, 1972.

[97] R. Lachner. Collision avoidance as a differential game: real-time approximation of optimal strategies using higher derivatives of the value function. *Systems, Man, and Cybernetics*, Oct. 1997.

[98] Sandip K Lahiri. *Multivariable predictive control: Applications in industry*. John Wiley & Sons, 2017.

[99] V. Lakshmikantham, V. M. Matrosov, and S. Sivasundaram. *Vector Lyapunov*

Functions and Stability Analysis of Nonlinear Systems. Springer Science & Business Media, 1991.

[100] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[101] Dale Lawrence, Eric Frew, and William Pisano. Lyapunov vector fields for autonomous uav flight control. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2012.

[102] Dongxu Li, Jose B. Cruz Jr., Genshe Chen, Chiman Kwan, and Mou-Hsiung Chang. A Hierarchical Approach To Multi-Player Pursuit-Evasion Differential Games. *Proceedings of the 44th IEEE Conference of Decision and Control (CDC)*, 44(5):5674–5679, 2005.

[103] Qinghua Li, Haiming Li, Jiahui Wang, and Chao Feng. Robot navigation in crowds environment base deep reinforcement learning with pomdp. In *International Conference on Multimedia Technology and Enhanced Learning*, pages 675–685. Springer, 2022.

[104] Xueting Li, Sha Yi, and Katia Sycara. Multi-agent deception in attack-defense stochastic game. In *International Symposium Distributed Autonomous Robotic Systems*, pages 242–255. Springer, 2021.

[105] P. L. Lions. On the hamilton-jacobi-bellman equations. *Acta Applicandae Mathematica*, 1(1):17–41, 1983.

[106] Shuai Liu and Pengcheng Liu. A review of motion planning algorithms for robotic arm systems. *RiTA 2020*, pages 56–66, 2021.

[107] F. S. Lobato and V. Steffen Jr. *Multi-Objective Optimization Problems: Concepts and Self-Adaptive Parameters with Mathematical and Engineering Applications*. Springer International Publishing AG, Gewerbestrasse 11, 6330 Cham, Switzerland, 2017. Chapter 2: Multi-objective Optimization Problem.

[108] Chunbo Luo, Sally I McClean, Gerard Parr, Luke Teacy, and Renzo De Nardi. Uav position estimation and collision avoidance using the extended kalman filter. *IEEE Transactions on Vehicular Technology*, 62(6):2749–2762, 2013.

[109] Jaakko Luttinen, Tapani Raiko, and Alexander Ilin. Linear state-space model with time-varying dynamics. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 338–353. Springer, 2014.

[110] N. Malone, H. Chiang, K. Lesser, M. Oishi, and L. Tapia. Hybrid Dynamic Moving Obstacle Avoidance Using a Stochastic Reachable Set-Based Potential Field. *IEEE Transactions on Robotics*, 33(5), Oct. 2017.

[111] Xuerong Mao. *Stochastic Differential Equations and Applications*. Horwood Publishing, Chichester, UK, 2008. Second Edition.

[112] A. A. Masoud. A Harmonic Potential Field Approach for Navigating a Rigid,

Nonholonomic Robot in a Cluttered Environment. *IEEE International Conference on Robotics and Automation*, pages 3993–3999, May 2009.

[113] MathWorks. Matlab - the language of technical computing. `www.mathworks.com/products/matlab/`.

[114] Akiko Matsuura, Shinji Suzuki, Mitsuru Kono, and Akitoshi Sakaguchi. Lateral guidance control of uav using feedback error learning. In *AIAA Infotech@ Aerospace 2007 Conference and Exhibit*, page 2727, 2007.

[115] J. S. McGrew, J. P. How, B. Williams, and N. Roy. Air-Combat Strategy using Approximate Dynamic Programming. *Journal of Guidance, Control, and Dynamics*, 33(5):1509–1514, September 2010.

[116] R. L. McNeely, R. V. Iyer, and P. R. Chandler. Tour Planning for an Unmanned Air Vehicle Under Wind Conditions. *Journal of Guidance, Control, and Dynamics*, 30(5):1299–1306, Sept. 2007.

[117] N. Meuleau, M. Hauskrecht, K. Kim, L. Peshkin, L. P. Kaelbling, T. Dean, and C. Boutilier. Solving Very Large Weakly Coupled Markov Decision Processes. *AAAI*, 1998.

[118] Marc A Michot. *Brockett's necessary conditions and the stabilization of nonlinear control systems*. San Jose State University, 2011.

[119] D. Milutinović, D. W. Casbeer, D. Kingston, and S. A. Rasmussen. Stochastic approach to small uav feedback control for target tracking and blind spot avoidance. In *Proceedings of the 1st IEEE Conference on Control Technology and Applications*, 2017.

[120] D. Milutinović, D. W. Casbeer, and M. Pachter. Markov Inequality Rule for Switching among Time Optimal Controllers in a Multiple Vehicle Intercept Problem. *Automatica*, 87:274–280, 2018.

[121] Dejan Milutinović, David W. Casbeer, Derek Kingston, and Steven Rasmussen. A stochastic approach to small UAV feedback control for target tracking and blind spot avoidance. In *Proceedins of 2017 IEEE Conference on Control Technology and Applications (CCTA)*, pages 1031–1037, 2017.

[122] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin. A Time-Dependent Hamilton-Jacobi Formulation of Reachable Sets for Continuous Dynamic Games. *IEEE Transactions on Automatic Control*, 50(7):947–957, July 2005.

[123] F. Mondada and M. Bonani. e-puck education robot. `http://www.e-puck.org/`.

[124] A. A Munishkin, A. Hashemi, D. W Casbeer, and D. Milutinović. Scalable markov chain approximation for a safe intercept navigation in the presence of multiple vehicles. *Autonomous Robots*, 43:575–588, 2019.

[125] A. A. Munishkin, D. Milutinović, and D. W. Casbeer. Stochastic optimal control navigation with the avoidance of unsafe configurations. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 211–218, June 2016.

[126] A. A. Munishkin, D. Milutinović, and D. W. Casbeer. Safe Navigation With the Collision Avoidance of a Brownian Motion Obstacle. *ASME Dynamic Systems and Control Conference*, 3, October 2017.

[127] S. G. Nersesov, W. M. Haddad, and Q. Hui. Finite-time stabilization of nonlinear dynamical systems via control vector Lyapunov functions. *Journal of the Franklin Institute*, 345:819–837, April 2008.

[128] Karl J. Obermeyer, Paul Oberlin, and Swaroop Darbha. Sampling-based path planning for a visual reconnaissance unmanned air vehicle. *Journal of Guidance, Control, and Dynamics*, 35(2):619–631, 2012.

[129] B. Øksendal. *Stochastic Differential Equation: An Introduction with Applications*. Berlin: Springer-Verlag, 2000.

[130] Gonzalo Pajares. Overview and current status of remote sensing applications based on unmanned aerial vehicles (UAVs). *Photogrammetric Engineering & Remote Sensing*, 81(4):281–330, 2015.

[131] D. Panagou, D. M. Stipanović, and P. G. Voulgaris. Distributed Coordina-

tion Control for Multi-Robot Networks Using Lyapunov-Like Barrier Functions. *IEEE Transactions on Automatic Control*, 61(3):617–632, 2016.

[132] Hyoshin Park, Dahai Liu, Sirish Namilae, et al. Multi-agent reinforcement learning-based pedestrian dynamics models for emergency evacuation. Technical report, North Carolina A&T Sate University. Transportation Institute. Center For . . . , 2022.

[133] V. S. Patsko and V. L. Turova. *Numerical Study of the 'Homicidal Chauffeur' Differential Game with the Reinforced Pursuer*, volume 12. Nova Scientific, 2007. Chap. 8.

[134] Vladimir Pavlovic, James M Rehg, and John MacCormick. Learning switching linear models of human motion. In *NIPS*, volume 2, page 4, 2000.

[135] W. B. Powell. What You Should Know About Approximate Dynamic Programming. *Naval Research Logistics (NRL)*, 56(3):239–249, April 2009.

[136] Warren B. Powell. *Approximate Dynamic Programming: Solving the curses of Dimensionality*. Wiley Series in Probability and Statistics, 2007.

[137] Warren B Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

[138] Warren B. Powell. What you should know about approximate dynamic programming. *Naval Research Logistics (NRL)*, 56(3):239–249, April 2009.

[139] S. Prajna and A. Jadbabaie. Safety verification of hybrid systems using barrier certificates. *Hybrid Systems: Computation and Control. HSCC 2004. Lecture Notes in Computer Science*, 2993, 2004. Alur R., Pappas G.J. (eds).

[140] S. Prajna, A. Jadbabaie, and G. J. Pappas. Stochastic Safety Verification Using Barrier Certificates. *IEEE Conference on Decision and Control (CDC)*, pages 929–934, December 2004.

[141] S. Prajna, A. Jadbabaie, and G. J. Pappas. A Framework for Worst-Case and Stochastic Safety Verification Using Barrier Certificates. *IEEE Transactions on Automatic Control*, pages 1415–1428, August 2007.

[142] Steven A.P. Quintero and João P. Hespanha. Vision-based target tracking with a small UAV: Optimization-based control strategies. *Control Engineering Practice*, 32:28–42, 2014.

[143] H. Jin Kim R. Vidal, O. Shakernia, D. Hyunchul Shim, and Shankar Sastry. Probabilistic Pursuit–Evasion Games: Theory, Implementation, and Experimental Evaluation. *IEEE Transactions on Robotics and Automation*, 18(5):662–669, October 2002.

[144] K. Rafail. *Stochastic Stability of Differential Equations*. Springer-Verlag Berlin Heidelberg, 2012.

[145] MV Ramana, S Aditya Varma, and Mangal Kothari. Motion planning for a fixed-wing uav in urban environments. *IFAC-PapersOnLine*, 49(1):419–424, 2016.

[146] Vicon Motion Systems Ltd. UK registered no. 1801446. Bonita motion capture camera. `http://www.vicon.com/products/camera-systems/bonita`.

[147] Vicon Motion Systems Ltd. UK registered no. 1801446. Motion capture systems — vicon. `www.vicon.com`.

[148] Vicon Motion Systems Ltd. UK registered no. 1801446. Tracker motion capture software. `http://www.vicon.com/products/software/tracker`.

[149] Laminar Research. X-plane 10: Ultra realistic flight simulator. `http://www.x-plane.com/pro/landing/`.

[150] Coppelia Robotics. Robot simulator simulation v-rep virtual experimentation platform. `http://www.coppeliarobotics.com/`.

[151] L. C. G. Rogers and D. Williams. *Diffusions, Markov Processes and Martingales. Volume 1: Foundations*. Cambridge Univ. Press, Cambridge, U.K., 2000.

[152] P. Rydesäter. Tcp/udp/ip toolbox 2.0.6 - file exchange - matlab central. http://www.mathworks.com/matlabcentral/fileexchange/345-tcp-udp-ip-toolbox-2-0-6.

[153] D. M. Stipanović S. Shankaran and C. J. Tomlin. Collision Avoidance Strategies for a Three-Player Game. *Advances in Dynamic Games, Annals of the International Society of Dynamic Games*, 11, 2011.

[154] Jesús Sánchez-García, José Manuel García-Campos, SL Toral, DG Reina, and Federico Barrero. An intelligent strategy for tactical movements of uavs in disaster scenarios. *International journal of distributed sensor networks*, 12(3):8132812, 2016.

[155] Arjan Schaft and Hans Schumacher. *An introduction to hybrid dynamical systems*. Springer, 2000.

[156] Jürgen Scherer and Bernhard Rinner. Short and full horizon motion planning for persistent multi-uav surveillance with energy and communication constraints. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 230–235. IEEE, 2017.

[157] J. Schulman, J. Ho, A. Lee, I. Awwal, H. Bradlow, and P. Abbeel. Finding Locally Optimal, Collision-Free Trajectories with Sequential Convex Optimization. *robotics science and systems*, 2013.

[158] Eduard Semsch, Michal Jakob, Dušan Pavlicek, and Michal Pechoucek. Autonomous UAV surveillance in complex urban environments. In *Proceedings of 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, volume 2, pages 82–85, 2009.

[159] Vitaly Shaferman and Tal Shima. Unmanned aerial vehicles cooperative tracking of moving ground target in urban environments. *Journal of guidance, control, and dynamics*, 31(5):1360–1371, 2008.

[160] Hazim Shakhatreh, Ahmad H Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah, and Mohsen Guizani. Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges. *IEEE Access*, 7:48572–48634, 2019.

[161] Shai Shalev-Shwartz, Nir Ben-Zrihem, Aviad Cohen, and Amnon Shashua. Long-term planning by short-term prediction. *arXiv preprint arXiv:1602.01580*, 2016.

[162] M. Shanmugavel. *Path Planning of Multiple Autonomous Vehicles*. PhD thesis, Cranfield University, 2007.

[163] Tal Shima, Steven J Rasmussen, and Phillip Chandler. Uav team decision and control using efficient collaborative estimation. *Journal of Dynamic Systems, Measurement, and Control*, pages 609–619, September 2007.

[164] Qingshuo Song and Gang George Yin. Convergence rates of Markov chain approximation methods for controlled diffusions with stopping. *Journal of Systems Science and Complexity*, 23(3):600–621, jul 2010.

[165] Q.S. Song and G. Yin. Existence of Saddle Points in Discrete Markov Games and Its Application in Numerical Methods for Stochastic Differential Games. In *2006 45th IEEE Conference on Decision and Control*, pages 6325–6330, dec 2006.

[166] M. Z. Spivey and W. B. Powell. Dynamic Assignment Problem. *Transportation Science*, 38(4):399–419, 2004.

[167] Sriraman Sridharan and Rangaswami Balakrishnan. *Discrete Mathematics: Graph Algorithms, Algebraic Structures, Coding Theory, and Cryptography*. CRC Press, 2019.

[168] I. M. Stamova. Vector Lyapunov functions for practical stability of nonlinear impulsive functional differential equations. *Journal of Mathematical Analysis and Applications*, 325:612–623, March 2006.

[169] L. E. Sucar. *Parallel Markov Decision Processes*, volume 214. Springer, 2007.

[170] S. Summers and J. Lygeros. Verification of discrete time stochastic hybrid systems: A stochastic reach-avoid decision problem . *Automatica*, 46(12):1951–1961, December 2010.

[171] Sean Summers, Maryam Kamgarpour, Claire Tomlin, and John Lygeros. Stochastic system controller synthesis for reachability specifications encoded by random sets. *Automatica*, 49(9):2906–2910, 2013.

[172] Zhiyong Sun, Héctor Garcia de Marina, Brian DO Anderson, and Changbin Yu. Collaborative target-tracking control using multiple fixed-wing unmanned aerial vehicles with constant speeds. *Journal of Guidance, Control, and Dynamics*, 44(2):238–250, 2021.

[173] Panagiotis Theodorakopoulos and Simon Lacroix. UAV target tracking using an adversarial iterative prediction. In *Proceedings of 2009 IEEE International Conference on Robotics and Automation*, pages 2866–2871, 2009.

[174] Panagiotis Theodorakopoulos and Simon Lacroix. Uav target tracking using an adversarial iterative prediction. In *2009 IEEE International Conference on Robotics and Automation*, pages 2866–2871. IEEE, 2009.

[175] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[176] John Tisdale, ZuWhan Kim, and J Karl Hedrick. Autonomous uav path planning and estimation. *IEEE Robotics & Automation Magazine*, 16(2):35–42, 2009.

[177] Panagiotis Tsiotras. Bounded rationality in learning, perception, decision-making, and stochastic games. In *Handbook of Reinforcement Learning and Control*, pages 491–523. Springer, 2021.

[178] T. Urakubo. Stability Analysis and Control of Nonholonomic Systems with Potential Fields. *Journal Intelligent Robot Systems*, 89:121–137, June 2018.

[179] Kimon P Valavanis and George J Vachtsevanos. *Handbook of unmanned aerial vehicles*, volume 2077. Springer, 2015.

[180] Marcos A. M. Vieira, Ramesh Govindan, and Gaurav S. Sukhatme. Scalable and practical pursuit-evasion with networked robots. *Intelligent Service Robotics*, 2(4):247, 2009.

[181] K. Virtanen, J. Karelahti, and T. Raivio. Modeling Air Combat by a Moving Horizon Influence Diagram Game. *Journal of Guidance, Control, and Dynamics*, 29(5):1509–1514, September 2006.

[182] A. Von Moll, D. W. Casbeer, E. Garcia, and D. Milutinović. Pursuit-evasion of an Evader by Multiple Pursuers. *Proceedings of the 2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2018.

[183] Alexander Von Moll, Eloy Garcia, David Casbeer, M Suresh, and Sufal Chandra Swar. Multiple-pursuer, single-evader border defense differential game. *Journal of Aerospace Information Systems*, 17(8):407–416, 2020.

[184] Guofang Wang, Ziming Li, Wang Yao, and Sikai Xia. A multi-population mean-field game approach for large-scale agents cooperative attack-defense evolution in high-dimensional environments. *Mathematics*, 10(21):4075, 2022.

[185] L. Wang, A. Ames, and M. Egerstedt. Safety barrier certificates for heterogeneous multi-robot systems. In *2016 American Control Conference (ACC)*, pages 5213–5218, July 2016.

[186] L. Wang, A. D. Ames, and M. Egerstedt. Safety barrier certificates for collisions-free multirobot systems. *IEEE Transactions on Robotics*, 33(3):661–674, June 2017.

[187] Yoko Watanabe and Patrick Fabiani. Optimal guidance design for uav visual target tracking in an urban environment. *IFAC Proceedings Volumes*, 43(15):69–74, 2010. 18th IFAC Symposium on Automatic Control in Aerospace.

[188] Albert Wu and Jonathan P How. Guaranteed infinite horizon avoidance of unpredictable, dynamically constrained obstacles. *Autonomous robots*, 32(3):227–242, 2012.

[189] Yu Wu and Kin Huat Low. An adaptive path replanning method for coordinated operations of drone in dynamic urban environments. *IEEE Systems Journal*, 2020.

[190] Ji Xiaohui, Zhang Xuejun, and Guan Xiangmin. A collision avoidance method based on satisfying game theory. In *2012 4th international conference on intelligent human-machine systems and cybernetics*, volume 2, pages 96–99. IEEE, 2012.

[191] Yang Yang, Xiaorui Xi, Songtao Miao, and Jinran Wu. Event-triggered output feedback containment control for a class of stochastic nonlinear multi-agent systems. *Applied Mathematics and Computation*, 418:126817, 2022.

[192] Y. Yavin. Stochastic Two-Target Pursuit-Evasion Differential Games in the Plane. *Journal of Optimization Theory and Applications*, 56(3):325–343, February 1988.

[193] Y. Yavin and R. De Villers. Stochastic Pursuit-Evasion Differential Games in 3D. *Journal of Optimization Theory and Applications*, 56(3):345–357, February 1988.

[194] Y. Yoon, J. Shin, H. J. Kim, Y. Park, and S. Sastry. Model-predictive active steering and obstacle avoidance for autonomous ground vehicles. *Control Engineering Practice*, 17:741–750, 2009.

[195] Huili Yu, Kevin Meier, Matthew Argyle, and Randal W Beard. Cooperative path planning for target tracking in urban environments using unmanned air and ground vehicles. *IEEE/ASME Transactions on Mechatronics*, 20(2):541–552, 2014.

[196] Huili Yu, Kevin Meier, Matthew Argyle, and Randal W. Beard. Cooperative path planning for target tracking in urban environments using unmanned air and

ground vehicles. *IEEE/ASME Transactions on Mechatronics*, 20(2):541–552, 2015.

[197] X. Yu, X. Zhou, and Y. Zhang. Collision-Free Trajectory Generation and Tracking for UAVs Using Markov Decision Process in a Cluttered Environment. *Intelligent & Robotic Systems*, March 2018.

[198] Ugur Zengin and Atilla Dogan. Real-time target tracking for autonomous UAVs in adversarial environments: A gradient search algorithm. *IEEE Transactions on Robotics*, 23(2):294–307, 2007.

[199] Yihao Zhang, Zhaojie Chai, and George Lykotrafitis. Deep reinforcement learning with a particle dynamics environment applied to emergency evacuation of a room with obstacles. *Physica A: Statistical Mechanics and its Applications*, 571:125845, 2021.

[200] Jiajun Zhao, Yifan Wang, and Qiangde Wang. Event-triggered formation-containment control for multiple euler-lagrange systems with input saturation. *Journal of the Chinese Institute of Engineers*, 45(4):313–323, 2022.

[201] Dingjiang Zhou, Zijian Wang, Saptarshi Bandyopadhyay, and Mac Schwager. Fast, on-line collision avoidance for dynamic vehicles using buffered voronoi cells. *IEEE Robotics and Automation Letters*, 2(2):1047–1054, 2017.