UAV swarming with collision avoidance and communication constraints

Abdul Hanif Zaini

2020

Abdul Hanif Zaini. (2020). UAV swarming with collision avoidance and communication constraints. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/137101>

<https://doi.org/10.32657/10356/137101>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 17:49:52 SGT

UAV Swarming with Collision Avoidance and Communication Constraints

Abdul Hanif Zaini

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2020

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

11 Feb. 2020

.....

Date

.....

Abdul Hanif Zaini

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

11 Feb. 2020

.....

Date

.....

Prof. Lihua Xie

Authorship Attribution Statement

This thesis contains material from 2 papers published in the following peer-reviewed journal(s) and 1 paper accepted at a conference in which I am listed as an author.

Chapter 3 is published as Bing Zhu, Abdul Hanif Zaini and Lihua Xie, “Distributed guidance for interception by using multiple rotary-wing unmanned aerial vehicles,” IEEE Transactions on Industrial Electronics, 64(7): 5648-5656, 2017.

The contributions of the co-authors are as follows:

- Dr Zhu prepared and revised the manuscript drafts.
- Dr Zhu designed the single UAV proportional navigation guidance law and the task allocation algorithm.
- I co-designed the distributed guidance law with collision avoidance with Dr Zhu.
- All the presented simulations were conducted by Dr Zhu.
- I designed and conducted the physical experiments.
- Prof Xie provided the initial project direction and edited the manuscript drafts.

Chapter 4 is published as Abdul Hanif Zaini and Lihua Xie, “Triggered communication for reciprocal collision avoidance in multi-agent navigation,” Proc. the 2018 IEEE 14th International Conference on Control and Automation (ICCA), Anchorage, Alaska, June 2018.

The contributions of the co-authors are as follows:

- I prepared and revised the manuscript drafts.
- I designed the entire algorithm and conducted the simulations.
- Prof Xie provided direction and edited the manuscript drafts.

Chapter 5 is published as Abdul Hanif Zaini and Lihua Xie, “Distributed drone traffic coordination using triggered communication.,” Unmanned Systems, 08(01): 1-20, 2020.

The contributions of the co-authors are as follows:

- I prepared and revised the manuscript drafts.
- I designed and analyzed the entire algorithm and conducted the simulations.

- Prof Xie provided direction and edited the manuscript drafts.

11 Feb. 2020

.....

Date

.....

Abdul Hanif Zaini

Acknowledgments

First and foremost I would like to express my most heartfelt gratitude and appreciation to my supervisor, Professor Xie Lihua, for his unwavering support, expert guidance and rigorous insight. This thesis would not have been possible without his kind encouragement and dedication to keeping me on track. His generosity of knowledge, time and energy is an inspiration.

Secondly, I am also grateful to Professor Zhu Bing, Professor Hu Jinwen and Dr. Rodney Teo for their assistance and advice for my PhD. research. The discussions were encouraging and immensely helpful.

Thirdly, many thanks also to my fellow laboratory mates Dr. Guo Kexin and Mr. Nguyen Thien Minh for the countless discussions and exchange of ideas as well as their assistance in conducting experiments and other technical issues.

Thanks should also go to Mr. Yuen Sien Huan, the lab technician at the Internet of Things Laboratory, for his prompt valuable assistance on technical and laboratory matters.

I gratefully acknowledge the financial support received from the Nanyang President Graduate Scholarship.

I am also deeply grateful to my family for their support, encouragement and understanding.

Lastly, I would also like to thank everyone else who contributed in any way to the completion of this research thesis and my graduate studies no matter how small.

Abstract

Collision avoidance is critical for the safe and reliable operation of swarms of UAVs (Unmanned Aerial Vehicles). Numerous works have already been reported in literature. Most of these methods can be grouped into three categories: repulsion, partition-based, and trajectory replanning. In this thesis we seek to further develop some of these existing methods with consideration for special scenarios such as intruder UAV interception and dense UAV traffic coordination where physical and communication constraints are often omitted in the design of swarm algorithms. Collision avoidance can be simple to achieve. The challenge is in ensuring that the swarm scenario objectives are not compromised.

The intruder interception problem using UAVs is an interesting problem on its own but also poses a challenge for collision avoidance when multiple agents and intruders are considered. The accessibility of UAV technology motivates us to design a non-destructive method for stopping UAVs that intrude into restricted areas by using our own UAVs to intercept and capture them. For interception, we propose a proportional navigation guidance law, on a double integrator model, for a quadrotor UAV with velocity feedback that globally guarantees interception with non-zero hitting velocity within finite time. Our novel solution to the collision avoidance problem specific to the guidance law is to apply repulsion to the line-of-sight vectors between pairs of agents and intruders. We demonstrate our method's effectiveness with both simulations and a semi-physical experiment.

The experiment results motivate us to consider communication constraints for the subsequent work. Repulsion-based avoidance is unsuitable where other agents' states are obtained via limited communication. We propose instead a significantly modified version of the partition-based velocity obstacle method combined with an event-triggered communication protocol. To limit the scope, we consider the two-dimensional single integrator model and the scenario where agents are tasked with moving to fixed target positions while avoiding collision. Agents are assumed to be connected by a mobile ad hoc network allowing them to communicate with

any other agent in the swarm. However, communication for each agent is limited to only one other agent at a time. The protocol determines for each agent when and to whom to communicate, and reduces the rate of communication. The method's effectiveness is shown with simulations.

We further extend the work to overcome its limitation on agent numbers and spatial configuration by designing a hybrid collision avoidance method combining the spatial partition-based Voronoi Cell and Velocity Obstacle methods. In addition, to overcome deadlocks and livelocks, we introduce a priority system for distributed coordination by having agent pairs make mutual decisions based on each agent's conditional priority. A similar event trigger-based communication protocol is presented. Our method's effectiveness is demonstrated in multiple simulations involving up to 100 agents.

Finally, to apply the collision avoidance methods developed based on the single integrator model to the constrained triple integrator system, we develop a trajectory buffering method using the input partition-based Barrier Certificate collision avoidance method. We feed a virtual single-integrator agent to the previous hybrid collision avoidance method and use the provided input and position as nominal states to be tracked by the actual UAV. Its relative position then determines the virtual agent's progress along the nominal input direction. The barrier certificate method is used to ensure that the agent never exceeds a maximum displacement from the nominal position.

The methods we present in this work are aimed at swarms of quadcopter UAVs and are primarily validated by simulations. Assumptions on the dynamics and communication system are based on literature and current technology. The methods are not necessarily limited to the scenarios discussed and recommendations for future work are discussed.

Contents

Statement of Originality	iii
Supervisor Declaration Statement	v
Authorship Attribution Statement	vii
Acknowledgments	ix
Abstract	xi
List of Figures	xvii
List of Tables	xix
Symbols and Acronyms	xxi
1 Introduction	1
1.1 Motivation	2
1.2 Objectives	5
1.3 Major Contributions	5
1.4 Organization of the Thesis	6
2 Literature Review	9
2.1 Quadrotor Helicopter Dynamics	10
2.2 Collision Avoidance	13
2.2.1 Repulsion-based methods	14
2.2.1.1 Repulsive Control	14
2.2.1.2 Navigation Functions	18
2.2.1.3 Summary of Repulsion-based Methods	19
2.2.2 Partition-based Methods	20
2.2.2.1 Voronoi Cells	20
2.2.2.2 Velocity Obstacles	22
2.2.2.3 Barrier Certificates	25
2.2.2.4 Summary of Partition-based Methods	27

2.2.3	Trajectory Replanning	28
2.2.3.1	Mixed Integer Methods	29
2.2.3.2	Sequential Convex Programming	30
2.2.3.3	Other Methods	30
2.2.3.4	Summary of Trajectory Replanning Methods	32
2.3	Summary	32
3	LOS Repulsion for Multi-UAV Intruder Interception	35
3.1	Motivation	36
3.2	Problem Formulation	37
3.3	Three-Dimensional Guidance Law for Single Quadrotor UAV Inter- ception	39
3.3.1	Guidance Law	39
3.3.2	Simulation	42
3.3.3	Experiment	43
3.4	Multi-Interception with LOS-based CA	44
3.4.1	LOS Repulsion CA	44
3.4.2	Distributed Guidance Law	46
3.4.3	Simulation	49
3.4.4	Semi-physical Experiment	50
3.4.5	Issues with Experimental Implementation	51
3.5	Summary	53
4	VO-based Reciprocal CA with Triggered Communication	55
4.1	Motivation	56
4.2	Problem Formulation	57
4.2.1	System	57
4.2.2	Communication	58
4.2.3	Objective	58
4.3	VO-based Reciprocal CA	59
4.3.1	Preliminaries	59
4.3.2	Feasibility Maintenance of the Quadratic Program	62
4.3.3	Constraint Relief	64
4.3.4	Maximum Velocity Constraint	66
4.4	Communication Protocol Using Measure-based Triggers	68
4.4.1	Trigger Mechanism	68
4.4.2	Constraint Update Selection	71
4.5	Simulations	71
4.6	Summary	76
5	Hybrid Partition-based CA with Triggered Communication	79
5.1	Motivation	80
5.2	Problem Formulation	81
5.2.1	Model	81

5.2.2	Communication	82
5.2.3	Objective	83
5.3	Hybrid CA Algorithm	83
5.3.1	Overview	83
5.3.2	Buffered Voronoi Partitions	87
5.3.3	Additional Repulsive Buffer	88
5.3.4	Collision Cone Partitions	90
5.3.5	Information Exchanged	92
5.3.6	Priority System	94
5.3.7	Mode Switching Conditions	96
5.3.8	CCP Boundary Selection	97
5.3.9	Target Velocity	98
5.3.10	Convex Optimization on Velocity	100
5.4	Analysis	100
5.4.1	Deadlock	102
5.4.1.1	BVP and Repulsion Deadlock	102
5.4.1.2	CCP Overlap Deadlock	104
5.4.2	Livelock	109
5.5	Trigger-based Communication Protocol	111
5.5.1	BVP Trigger	112
5.5.2	CCP Trigger	114
5.6	Simulation	115
5.6.1	Small-Scale Algorithm Demonstration	115
5.6.2	Antipodal Swap of 100 Agents	117
5.6.3	Crowd Shuffling of 50 Agents in 100 Randomized Simulations	117
5.6.4	Simulation Without Priority System	120
5.6.5	Simulation Without Repulsion	122
5.6.6	Effect of Control Time Interval	122
5.7	Summary	123
6	Jerk-limited Holonomic Robot Motion Planning Using Barrier Functions	127
6.1	Motivation	128
6.2	Problem Formulation	130
6.2.1	System	130
6.2.2	Nominal Path Requirement	130
6.2.3	Problem	131
6.3	Discretized CBF Preliminary	131
6.4	Jerk-limited Motion Restriction Algorithm	132
6.4.1	DCBF for Displacement	133
6.4.2	Additional DCBF for Velocity	137
6.4.3	Nominal Position Update	137
6.5	Simulation	138

6.6 Summary	141
7 Conclusion and Recommendations	145
7.1 Conclusion	145
7.2 Recommendations	147
List of Author's Publications	149
Bibliography	151

List of Figures

2.1	Quadrotor helicopter and model	10
2.2	$VO_{i j}^\tau$ and $ORCA_{i j}^\tau$ in relative velocity and position space superimposed.	23
3.1	Interception scenario	36
3.2	Proportional Navigation	39
3.3	Simulation of integrated guidance controller	43
3.4	Experiment of integrated guidance controller	43
3.5	Relative LOS	44
3.6	Multi-UAV interception simulation	50
3.7	Trajectories of semi-physical experiment	51
3.8	Logged acceleration data for semi-physical experiment.	52
4.1	$VO_{i j}$ and the respective $\beta_{i j}$ and $\beta_{j i}$ of agents i and j in relative velocity and position space superimposed.	60
4.2	Five agents on a circle swapping positions two places clockwise. . .	73
4.3	Agent 1 navigating to target position at (6,3) between two stationary agents.	74
4.4	Communication log of agent 4.	74
4.5	Log of trigger Δ_4	75
5.1	Velocity Obstacle with time horizon truncation.	84
5.2	Two CA modes.	85
5.3	Flowchart of the hybrid CA algorithm	86
5.4	Buffered Voronoi Partition positional constraint, $A_{i j}^{pos}$	88
5.5	Velocity Obstacle and Derived CCP constraints.	91
5.6	Rotation and direction rules.	94
5.7	Deadlock scenarios caused by BVP mode.	103
5.8	Example of agents in a deadlock caused by CCP mode.	105
5.9	An example of agent 1 surrounded with no gaps.	106
5.10	CCP and BVP constraints updated such that an opening becomes available which breaks the deadlock.	108
5.11	The hexagonal close-packed scenario.	111
5.12	Flowchart outlining the Trigger-based Communication Protocol. . .	112
5.13	Agent 1 passing between 3 other stationary agents.	116

5.14	100 agents ordered according to agent number on a circle with their target positions set antipodal.	118
5.15	50 agent shuffling simulation initialization and key completion times.	119
5.16	Log of communication instances for agent 47 from the simulation with RNG seed 48.	120
5.17	Deadlocked simulation where the priority system is disabled.	121
5.18	Comparison of group averaged completion times with and without repulsion.	123
5.19	Performance comparison for varying control time interval T	124
6.1	JMR Algorithm Outline.	133
6.2	X-Y plane view and displacement plots.	140
6.3	X-axis position, velocity and acceleration compared to reference.	141
6.4	Y-axis position, velocity and acceleration compared to reference.	142
6.5	Z-axis position, velocity and acceleration compared to reference.	142

List of Tables

3.1	Initial States of Interceptor and Target	42
3.2	Initial Positions and Velocities of Interceptors and Targets	50
4.1	Simulation Parameters	72
4.2	Trigger and Communication Counts	75
5.1	Simulation Parameters	115
5.2	Additional Results of 100 Simulations	119
6.1	Simulation Parameters	139

Symbols and Acronyms

Symbols

$\mathbb{R}^r$	the r -dimensional Euclidean space
$(\cdot)^\top$	the transpose of the matrix or vector
$\ \cdot\ $	the 2-norm of a vector in Euclidean space
∇f	the gradient vector of function f
$\mathcal{C}^k$	the function with continuous partial derivatives up to k th orders
$\mathbf{q}_i(t)$	the position vector $\mathbf{q}$ of agent i at time t
q_j	the scalar position on the j th axis
$\mathcal{N}$	the index set of agents

Acronyms

CA	Collision Avoidance
CBF	Control Barrier Function
DCBF	Discrete Control Barrier Function
GPS	Global Positioning System
JMR	Jerk-limited Motion Restriction
LOS	Line of Sight
OP	Optimization Problem
ORCA	Optimal Reciprocal Collision Avoidance
PN	Proportional Navigation
UAV	Unmanned Aerial Vehicle
VO	Velocity Obstacle
ZCBF	Zeroing Control Barrier Function

Chapter 1

Introduction

The past decade has seen increasing development and deployment of unmanned aerial vehicles (UAV) in various applications. Previously limited to military use, the technology has since proliferated in the civilian domain thanks to the advancement, miniaturization, and lowering costs of components, to the point of being affordable to general hobbyists. Apart from its well-known use for aerial photography, other examples of potential civilian applications include delivery, agriculture, law enforcement, environmental monitoring, and entertainment. The shrinking size and costs of individual units have also prompted attention towards the exploration of deploying these UAVs in swarms, following inspiration from nature.

Swarm robotics is heavily inspired by nature. Some examples include foraging ant colonies, flocks of birds coordinating to avoid predators, and even the movement of humans forming lanes in a crowd [1]. It is an attractive area of research due to its potential robustness, flexibility and scalability [2]. Robustness here refers to redundancy afforded by large numbers and decentralization. The swarm is potentially capable of achieving objectives that a single UAV cannot or would do with less efficiency. From [2], the robotic swarm is defined as a group of physical agents that are designed to produce an emergent collective behavior based on local interactions.

Further details on applications and guiding characteristics of nature-based robotic swarms are found in [3–6]. Examples of recommended applications for robotic swarms include area coverage and foraging. Area coverage or deployment refers to the use of swarm members as mobile sensors that can be quickly deployed to cover large areas efficiently, or as network nodes to provide communication on the battlefield or a disaster-affected area. Foraging refers to tasks where members may operate from a home base to do tasks such as search and rescue, mapping, construction and even intruder interception. [3, 4] specify that swarm members should be distributed in terms of control and do not require global information. Interestingly, [5, 6] instead advocate that some level of centralized control and access to global information may be useful due to the difference in sensing and communication capabilities between natural and artificial swarms.

Another reason for the recent interest in UAV swarm development is the advent of small-sized quadrotor helicopters. These quadrotor helicopters are mechanically simple but versatile in 3D spatial motion. Under appropriate assumptions, they can be considered holonomic. The commercial availability of parts, simple construction, and low maintenance make it a popular platform for researchers.

There exist many aspects of robotic swarms that are still open to further research but in this thesis we focus only on one. The subject of study presented in this thesis is the design of novel collision avoidance (CA) algorithms for a swarm of quadrotor UAVs. We consider this problem in the context of two scenarios with one motivating the other. The motivation for this issue is discussed in the following.

1.1 Motivation

Avoiding collision between agents of a swarm is a basic requirement for its safe operation [7–9]. Although one advantage of a swarm is its redundancy and potential expendability of its members, it is still prudent to design algorithms to minimize

the loss of agents by collision. This is especially important for swarms deployed in civilian locations where collateral damage should be avoided.

Agents may be designed to detect nearby neighbors either by proximity-based sensing such as with vision systems [10] or by exchanging state information over wireless communication [11, 12], and then move to avoid each other by some relevant avoidance algorithm. The former is dependent on the sensor, which may be unreliable if considering relatively fast moving agents which the sensor might not detect in time due to short range, low resolution or low sampling rate. Moreover, sensor information is limited to only physical information such as position and velocity. The latter involves agents measuring their own position, for example by global positioning system (GPS), and transmitting or broadcasting it wirelessly. This method is then subjected to bandwidth limitation, delays, and dropouts [13, 14].

In this thesis, only communication based methods are presented. Neither method is necessarily better than the other and performance is dependent on application and scenario. Chung et al. recommends the hierarchical approach to swarm design due to the variation of time scales of different aspects of decision-making, control and estimation [9]. Sensor-based avoidance can be designed to complement the communication-based one for redundancy, with both acting at smaller time scales than the more computationally intensive trajectory generation or other higher level algorithms.

The first scenario of interest in this thesis is multi-intruder interception. The proliferation of UAV technology comes with some illegal intrusion into restricted areas such as airports, prisons and key government installations. To avoid collateral damage, it is desirable to stop them by non-destructive means. Our initial research work on multi-UAV systems tackles the intruder UAV problem using our own UAVs to intercept and capture them. For the case where there may be more than one intruder, a CA method that minimizes disruption of the interception process is desired.

The second scenario involves the distributed coordination of swarm members in heavily-congested clusters. An example of where such clusters can occur is when a swarm is used for delivery or construction tasks [7, 12]. In such scenarios, the problem of limited bandwidth or data rate in wireless communication becomes significant due to large numbers of agents operating within communication range. Excessive communication may also cause more frequent dropouts or delays. Hence, the CA algorithm should be designed to require less frequent communication while still guaranteeing non-collision. This is also motivated by the challenges experienced in experimental implementation of the first scenario. The alternative solution would be to space agents out further or reduce communication power. The former means less efficient use of space and the latter reduces the distance in which other agents can detect it. Another challenge of this scenario is in ensuring that deadlocks can be avoided.

The constraints of current communication technology are often overlooked in the development of algorithms for aerial swarm applications. A majority of these algorithms are designed based on the assumption of an arbitrarily large bandwidth for communication [15, 16]. Specifically, neighbor information is assumed to be available to an agent continuously or at every time step. The recent surveys [3, 9] state that few experimental results exist that use decentralized topologies due to limitations of current technology. Successful experiments reported in literature usually limit the number of agents [17, 18], or set larger safety distances with relatively low speed to account for any delays or dropouts [12].

The limitations of communication technology may not be simply overcome with future advancements. Moreover, it is beneficial to reduce wireless communication usage if it is possible to do so while guaranteeing non-collision. The bandwidth may then be used to communicate other information to aid swarm cooperation.

1.2 Objectives

The objective of this research is to explore, formulate and develop novel solutions for problems in UAV swarm collision avoidance that are still largely open and unsolved. Collision avoidance is important for the safe operation of UAV swarms. Moreover, UAV swarms have not yet been deployed in useful applications. The primary open issues investigated in this thesis are the problems of limited bandwidth in communication and deadlock resolution. This thesis focuses on the versatile quadrotor helicopter UAV, which may be modeled by double or triple integrators given appropriate assumptions.

1.3 Major Contributions

Our main contributions can be stated as follows:

- *LOS-based repulsion collision avoidance*: A direction-based collision avoidance using potential field repulsion along relative lines of sight (LOS) to a UAV's assigned target is proposed. The method is integrated into the proportional navigation guidance law. This method ensures that the LOS towards an interceptor's target is clear of obstacles, which reduces disruption to its interception.
- *VO modification for event-triggered communication*: The issue of limited bandwidth is tackled using a modified version of the Velocity Obstacle (VO) [19] suitable for event-triggered communication. Modifications are provided which guarantee feasibility and thus non-collision. Measure-based triggers are also tailored to the collision avoidance method.
- *Hybrid collision avoidance for event-triggered communication*: The deadlock problems of the previous work are addressed by using a hybrid collision avoidance formulation consisting of Voronoi Cell, VO and repulsion methods. The

method is suitable for event-triggered communication and thus reduces required communication without compromising non-collision guarantee.

- *Priority system for distributed coordination of deadlock resolution:* A system is introduced which takes advantage of the communication system to compare the priorities of agent pairs. Agents compare priority based on certain conditions to determine how they should move relative to one another to break deadlocks. In this thesis, the priority is based on distance to target and identification number.
- *Discretized barrier function method for adapting single integrator planning to constrained triple integrators:* Discretized barrier functions are formulated to restrict a UAV modeled by a constrained triple integrator to within a maximum displacement from a virtual agent. The virtual agent is used by a higher level planner and its movement is updated according to the actual UAV's motion relative to the command velocity. The UAV tracks the virtual agent's command velocity. This allows use of a relatively simple single integrator model for the higher level planner.

1.4 Organization of the Thesis

The rest of the thesis is organized as follows.

Chapter 2 reviews the quadrotor helicopter dynamics and various collision avoidance methods from literature.

Chapter 3 presents the LOS-based repulsion collision avoidance for multi-intruder interception using UAVs. The proportional navigation guidance law is first introduced followed by the integration of the LOS-based repulsion using the Lyapunov method. Simulation and experimental results are given.

Chapter 4 covers the VO-based collision avoidance for multiple agents with event-triggered communication. Modifications are provided to the VO formulation to

ensure non-collision and feasibility between communication updates. Measure-based triggers are introduced to reduce communication. Simulation results are given.

Chapter 5 extends the method from the previous chapter by utilizing Voronoi Cell and repulsion in a hybrid collision avoidance method. A priority system is introduced to distributedly coordinate agents to avoid deadlocks. A similar event-triggered communication scheme is provided. Extensive analysis and simulation results are given.

Chapter 6 adapts the higher level single integrator algorithm of the previous chapter to the quadrotor UAV approximated by the constrained triple integrator model. Discretized barrier functions are formulated to restrict the UAV's displacement relative to a virtual agent. Simulation results are presented.

Finally, conclusions and recommendations for future work are provided in Chapter 7.

Chapter 2

Literature Review

In this chapter, a review of quadrotor helicopter dynamics and collision avoidance (CA) methods are presented.

The review of quadrotor dynamics is first given in Section [2.1](#). The quadrotor helicopter is chosen for this research due to it being very commonly used in aerial multi-robot research. It is also able to move in any direction in 3D space, which makes it a relatively versatile platform compared to other form factors such as fixed wing aircraft. It is also mechanically simpler than the monorotor helicopter due to the use of 4 rotor blades directly attached to electric motors. Although the full quadrotor model is not used in the works presented in this thesis. Knowledge of it is relevant to how they may be approximated, which is typically by double or triple integrators.

Subsequently, an extensive review of CA methods is provided in Section [2.2](#). This literature review aims to give a comprehensive understanding of the state-of-the-art of CA research work. It also highlights the shortcomings of existing approaches and challenges for future work, which motivate the research in this thesis.

2.1 Quadrotor Helicopter Dynamics

The dynamic model of a quadrotor helicopter UAV is briefly presented here based on [20, 21].

A quadrotor helicopter UAV is an unmanned rotorcraft vehicle that utilizes four rotors instead of the usual one (see Figure 2.1(a)). The four rotors are usually positioned on the vertices of a square and oriented such that their planes of rotation are parallel and on the same plane. The center of the square coincides with the center of gravity of the vehicle. Rotors in a directly opposing pair spin in the same direction but opposite to the other pair (see Figure 2.1(b)). Roll, pitch and yaw are controlled by varying the relative rotation rates of the rotors.

FIGURE 2.1: Quadrotor helicopter and model. (a) Platform used in intruder interception experiments in [22–24]. (b) Overlay of (a) illustrating frames of reference and quadrotor model. The green and purple arrows denote rotor thrust and rotation directions respectively.

Two right-handed frames of reference are defined as shown in Figure 2.1(b): earth-fixed inertial frame E and body-fixed frame B. $\{e_1, e_2, e_3\}$ and $\{b_1, b_2, b_3\}$ denote three-dimensional unit vectors along the first, second and third axes of the earth-fixed and body-fixed frames respectively. These vectors as denoted are expressed in frame E. Frame E corresponds to the well-known NED (North, East, Down) frame commonly used in aerospace-related engineering. The axes b_1 and b_2 correspond to the longitudinal and lateral axes as shown with b_3 completing the right-hand

frame. The translation of frame B, with respect to, and expressed in frame E, is denoted by $\mathbf{q} \in \mathbb{R}^3$.

The orientation of frame B with respect to frame E can be represented by the rotation matrix $R = [\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3] \in SO(3)$ [21]. This rotation transforms vectors expressed in the B frame to frame E. We use the roll-pitch-yaw (ϕ - θ - ψ or 1-2-3) extrinsic rotation sequence to construct the rotation matrix from Euler angles, i.e. rotation first about the first axis of frame E by ϕ with subsequent rotations also about frame E. Expressed in terms of Euler angles we obtain

$$R = \begin{bmatrix} \cos \theta \cos \psi & \sin \phi \sin \theta \cos \psi - \cos \phi \sin \psi & \sin \phi \sin \psi + \cos \phi \sin \theta \cos \psi \\ \cos \theta \sin \psi & \cos \phi \cos \psi + \sin \phi \sin \theta \sin \psi & \cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi \\ -\sin \theta & \sin \phi \cos \theta & \cos \phi \cos \theta \end{bmatrix}. \quad (2.1)$$

Note that Euler angles are used for intuitive representation of attitude but are not used in most applications of control and estimation in quadrotors.

The equations of motion are presented as follows [21].

$$\dot{\mathbf{q}} = \mathbf{v}, \quad (2.2)$$

$$\dot{\mathbf{v}} = g\mathbf{e}_3 - \frac{F\mathbf{b}_3}{m} - \frac{\Delta}{m}, \quad (2.3)$$

$$\dot{R} = R\boldsymbol{\omega}_\times, \quad (2.4)$$

$$\dot{\boldsymbol{\omega}} = I^{-1}(-\boldsymbol{\omega} \times I\boldsymbol{\omega} + \boldsymbol{\tau}), \quad (2.5)$$

where $\boldsymbol{\omega} \in \mathbb{R}^3$ is the body angular rotation vector expressed in frame B. $\boldsymbol{\omega}_\times$ denotes the skew-symmetric matrix such that $\boldsymbol{\omega}_\times \mathbf{x}$ is equivalent to the cross product $\boldsymbol{\omega} \times \mathbf{x}$ for any $\mathbf{x} \in \mathbb{R}^3$. Acceleration due to gravity is denoted by g and mass by m . The vector $\Delta \in \mathbb{R}^3$ encompasses aerodynamic effects such as profile drag, blade-flapping, blade induced drag and other unmodeled effects. $I \in \mathbb{R}^{3 \times 3}$ is the inertia matrix in frame B.

The forces, F , and torques, $\tau \in \mathbb{R}^3$, where $\tau = [\tau_1, \tau_2, \tau_3]^T$, are expressed in frame B and are summarized as follows.

$$\begin{bmatrix} F \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \begin{bmatrix} c_T & c_T & c_T & c_T \\ -dc_T & dc_T & dc_T & -dc_T \\ dc_T & -dc_T & dc_T & -dc_T \\ c_q & c_q & -c_q & -c_q \end{bmatrix} \begin{bmatrix} \eta_1^2 \\ \eta_2^2 \\ \eta_3^2 \\ \eta_4^2 \end{bmatrix}, \quad (2.6)$$

where coefficients c_T and c_q are lumped parameter coefficients of thrust and torque respectively. The moment arm of each rotor about the axes is denoted by constant d . The variables $\eta_i \in \mathbb{R}$, $i = 1, 2, 3, 4$, are the angular velocities of respective rotors. For controller design, this matrix is inverted to obtain η_i from the required force and torques.

The quadrotor helicopter is a non-linear underactuated system. In this regard, equations (2.2)-(2.6) serve to highlight how translational motion is achieved ($\mathbf{b}_3$ in (2.3)), that is, a quadrotor rotates to provide thrust in the lateral direction. Lateral thrust is also coupled with the vertical thrust required to maintain altitude or position on the third axis of the Earth-fixed frame.

Despite the underactuation, the quadrotor dynamics was shown to be differentially flat in [25]. Specifically, it was shown that the angular acceleration and velocity are algebraic functions of flat outputs chosen to be $[\mathbf{q}, \psi]$. Hence, sufficiently smooth trajectories for $\mathbf{q}$ given suitable bounded derivatives can be tracked by the quadrotor UAV. Alternatively, the quadrotor may be approximated by integrator dynamics. If the time response of the angular velocity control can be assumed to be sufficiently fast, then the quadrotor may be approximated by triple integrator dynamics with constrained velocity, acceleration and jerk.

2.2 Collision Avoidance

CA in the context of multi-robot systems refer to algorithms designed to prevent collision with other robots or non-cooperative obstacles. The latter is usually called obstacle avoidance in literature and involves environmental objects that may be static or dynamic. In this section, various relevant works in literature are reviewed. Only works relevant to robots that are free to move in 2D or 3D space are considered. As many relevant key works as possible are covered to the best of the author's knowledge. Open research issues are then highlighted in the summary.

The algorithms may be implemented directly in controller design or hierarchically. In the former, the avoidance and objective controllers are designed concurrently and may be coupled. For the latter, a higher level planner's control input, path or trajectory is modified by the lower level avoidance algorithm. This loop may also be iterated to obtain the best solution. Many variations exist and are pointed out accordingly in the review.

Other variations include online or offline implementation, dynamic models, and sensing restrictions. Typically, online is preferred with the alternative requiring prior knowledge of the environment. The single or double integrator is usually used to model robots due to simplicity. Other models are explored as extensions of existing results. Sensing may be done by communication between robots or by some sensor that can detect distance and size of obstacles. Each method also has variations in restriction such as limited bandwidth or sensing distance.

In this review, CA algorithms are primarily classified into one of three main categories according to the mechanism: repulsion-based, partition-based, and trajectory-replanning-based. These categories are similar to those identified in [9, 26]. The methods are further described in their respective following subsections.

2.2.1 Repulsion-based methods

Repulsion-based CA is sometimes also referred to as the artificial potential field method based on the forces existing between molecules. A monotonically decreasing positive-semidefinite potential function is usually designed with respect to distance to an obstacle. Its gradient is then used to produce a safe path away or as a repulsive control input. The former refers to the navigation function method whereby a potential field is produced from a map of obstacles to obtain a safe path by descent to the minima placed at the objective. The latter is used to modify the objective control input designed to drive the robot to its goal. The use of a potential function and its gradient for CA was first introduced in [27].

2.2.1.1 Repulsive Control

One method where a repulsive control input is used to guarantee non-collision is described in [28], which itself extends one of the earliest works on avoidance control in [29]. In [29], sufficient conditions for the avoidance of a given set $\mathcal{A}$ are formalized. From the theorem therein, in short, a given set of states $\mathcal{A}$ is avoidable if there exists a strategy $p^1(\cdot) \in U_1$ and a Lyapunov-like function $V(\cdot)$ such that

$$V(\mathbf{x}, t) > V(\mathbf{x}^1, t), \quad \forall \mathbf{x}^1 \in \partial\mathcal{A}, \quad \forall t_1 \geq t, \quad (2.7)$$

and for all $u^1 \in U_1$,

$$\partial V(\mathbf{x}, t)/\partial t + \nabla_{\mathbf{x}} V(\mathbf{x}, t) \dot{\mathbf{x}} \geq 0, \quad \forall u^2 \in U_2, \quad (2.8)$$

where $\dot{\mathbf{x}} = f(\mathbf{x}, t, u^1, u^2)$, $\mathbf{x}$ is the state, t is the time, and U_1 and U_2 represent the set of available control inputs respectively for CA and objective control. One key idea here is the design of a Lyapunov-like function that is everywhere higher in value than on the boundary. The other key idea is that the strategy $p^1(\mathbf{x}, t)$

must be designed such that there exists an input u_1 that can ensure $V(\mathbf{x}, t)$ is non-decreasing regardless of the other input u_2 , which may be used to achieve some other objective.

In [28], the following function is used to achieve the requirements of (2.7) and (2.8) but in a manner similar to [27].

$$V(\mathbf{x}, \Delta, \delta) = \left[\min \left(0, \frac{\|\mathbf{x}\|^2 - \Delta}{\|\mathbf{x}\|^2 - \delta} \right) \right]^2, \quad (2.9)$$

where $0 < \delta < \Delta$. δ is the minimum or avoidance radius. Note that V is instead non-increasing with respect to $\|\mathbf{x}\|$. The repulsive force is then given by its derivative,

$$p^1(\mathbf{x}) = \frac{\partial V(\mathbf{x}, \Delta, \delta)}{\partial \mathbf{x}} = \begin{cases} 0 & \text{if } \|\mathbf{x}\| > \Delta, \\ \frac{4\mathbf{x}(\Delta - \delta)(\|\mathbf{x}\|^2 - \Delta)}{(\|\mathbf{x}\|^2 - \delta)^3} & \text{if } \delta < \|\mathbf{x}\| \leq \Delta, \\ \text{undefined} & \text{if } \|\mathbf{x}\| = \delta, \\ 0 & \text{if otherwise.} \end{cases} \quad (2.10)$$

$V(\mathbf{x}, \Delta, \delta) \rightarrow \infty$ as $\|\mathbf{x}\|^2 \rightarrow \delta$. This ensures that $\|\mathbf{x}\|^2 < \delta$ is impossible as long as the corresponding $p^1(\mathbf{x})$ is achievable by the robot. However, in real robots where acceleration is finite, this may be impractical as the distance required to decelerate to zero at the robot's maximum deceleration needs to be accounted for. The method is also subject to deadlocks by local minima. The repulsion may be exactly cancelled out by control laws intended to drive the robot to its objective.

Deadlocks may be overcome by implementing gyroscopic forces acting perpendicular to repulsion. The work in [30] introduced this idea using $F_g = (\dot{\mathbf{q}} \times \mathbf{n}) \times \dot{\mathbf{q}}$ with double-integrator agents, where $\dot{\mathbf{q}}$ and $\mathbf{n}$ denote the current velocity and the vector pointing from the robot to the nearest obstacle respectively. Notably, in [30], a braking force was used in place of repulsion. However, no formal guarantee of non-collision was provided in that work. In [31], an elliptical rotational potential field

surrounding each obstacle was introduced that scales with the magnitude of input from an objective controller and by inverse square law on distance. The repulsion force acts parallel to the ellipse surface and never acts directly opposite the objective control input. The inputs are implemented as acceleration for robots modeled by damped mass objects. However, like [28], the repulsion force is unbounded as the robot approaches the obstacle. Recently, the authors of [12] proposed a similar idea called anisotropic repulsion to reduce oscillatory motion caused by opposite moving agents and implemented as a velocity to be tracked by the robot. The repulsion force scales linearly with distance and is rotated according to the robot's position relative to another robot and the direction of its objective. Unlike [30] and [31], the repulsion is not perpendicular to the obstacle's relative position. Non-collision in [12] is also not explicitly guaranteed and experimental success is likely achieved by conservatively chosen parameters. Positions of neighbours are obtained by wireless broadcast. A queuing algorithm was provided to handle distributed coordination of multiple UAVs sharing the same delivery position.

An alternative to potential field design to generate the repulsive control input is seen in flocking algorithms [32]. In [32], the pairwise potential field is repulsive under a specified distance, d_α , and attractive otherwise to achieve two of the three flocking requirements interpreted from nature in [33]. The double integrator is used to model agents. The potential field presented is obtained using a relatively complicated construction method to achieve a smooth pairwise potential with finite cut-off that has a minimum at distance, d_α , and is differentiable everywhere. A finite cut-off is required due to the finite acceleration ability of real robots and is determined by the structural Hamiltonian of the entire flocking group of agents. This Hamiltonian in [32] represents the sum of initial potential and kinetic energy of the group. This is a crucial issue in potential field design with finite cut-offs. The potential field method in the absence of dissipative forces is a conservative energy system. Hence, the repulsion that an agent provides must account for the entire group's energy. This issue is explored further in [34] with an additional requirement of connectivity preservation. An estimate of the sufficient control force

required given the system's initial energy is provided. Recent works to improve the flocking algorithm include [35] where an adaptive controller is introduced for non-linear systems with uncertain parameters, and [36] where coordination of obstacle avoidance is improved by having agents locally communicate obstacle information.

The unbounded potential in (2.9) can also be modified with a finite cut-off as shown in [37] and applied to the agents modeled by double integrators. This is done using the piecewise $\mathcal{C}^2$ smooth function

$$V_f(\mathbf{x}, \Delta, \delta) = \begin{cases} \Gamma \left[\min \left(0, \frac{\|\mathbf{x}\|^2 - \Delta}{\|\mathbf{x}\|^2 - \delta} \right) \right]^2, & \text{if } \|x\| \geq h, \\ -\mu \|x\| + c, & \text{otherwise,} \end{cases} \quad (2.11)$$

where μ is the maximum permissible acceleration, Γ and c are constants to ensure continuity based on parameters Δ , δ and maximum permissible velocity, and $\delta < h < \Delta$. The formulation also allows for discrete sampling with varying time intervals greater than some minimum $T_0 > 0$ as $\partial V_f / \partial \mathbf{x} = -\mu$ is a constant for $\|x\| \leq h$. However, unlike [32] and [34], (2.11) is limited to only one other agent. The case whereby there are more than one other agent interacting is not explored. The same author subsequently extended the work for sensing uncertainty but with continuous sampling instead in [38].

The issues of limited sensing rates and communication with limited bandwidth in groups of more than two agents were also tackled in another extension of [37] in [39]. To handle the multi-agent case, the unbounded potential function of the form (2.9) is used instead of (2.11). A self-triggered communication scheme was introduced where the next instance of communication with another agent is determined based on the time to collision. The intervals between communication instances become shorter as the agents approach each other. Although the time interval length can vary, an upper bound is still required due to sensing or communication range. There still exists some periodic communication, even if at a slower rate.

Scenarios with agents that have unusual shapes and dynamics were also considered. Examples are found in [40] with ellipsoidal agents of unicycle dynamics, and [41]

with non-linear Lagrangian dynamics. In [42], interactions between agents modeled by double integrator or unicycle dynamics are based on hydrodynamics, which is used to coordinate them to form predefined shapes. The agents act as particles in an incompressible fluid. An unbounded viscosity term proportional to the inverse of the square of distance to another agent is used to ensure non-collision.

2.2.1.2 Navigation Functions

The navigation function is a special artificial potential function designed such that its minimum coincides with the goal and the robot's initial potential energy is lower than those at positions coinciding with any obstacles [43]. Typically, obstacles are static and the environment information is known prior. A bounded-acceleration controller can be obtained from the gradient of the navigation function. Used with a damping term, the robot is guaranteed to reach its objective without collision with obstacles. The navigation function is used primarily to obtain a path or trajectory for a robot to reach its goal. In terms of CA, it is not vastly different from the methods presented earlier. The key difference is that control of the robot towards its goal is explicitly integrated into the potential function and achieved by gradient descent.

Decentralized navigation functions were also applied to the multi-agent case. An example is found in [44] but only applied to agents modeled by single integrators. A more recent work aimed at a multi-UAV application in [45] tackled issues such as local minima due to inter-UAV interaction and jitter around local minimum points due to discrete-time implementation. The method utilizes unbounded potential fields of the form introduced in [27] leaving consideration for constrained dynamics to future work. In [46], Lyapunov functions constructed using a differential game approach were used in a similar manner. The authors of [47] adapted the navigation function to practical application with a UAV swarm by dividing the area of operation into square cells with assigned potential values and may at most contain only one UAV at a time. Swarm members detect obstacles by sensors and

share this information by communication. Cell potentials are raised if an obstacle is located within. The agents then advance to the goal by moving to a cell that has the lowest potential of all the adjacent ones. This work circumvents the finite acceleration problem and provides a novel method for cooperative path finding.

2.2.1.3 Summary of Repulsion-based Methods

Potential field if applied pairwise and not as a navigation function is a simple method for CA. Minimal computation is needed and in most cases only the robot's relative position to obstacles or other robots is needed. However, we can observe from literature that the potential field design problem becomes increasingly difficult when considering more aspects of real world applications such as constrained dynamics with finite acceleration capability, limited communication bandwidth or sensor rates, and larger numbers of interacting robots. Research works are either limited in scope or circumvent the issue altogether such as in [47]. Open research problems in repulsion-based CA are as follows.

Current finite potential field formulations are not easily scalable. The potential function in (2.11) accounts for finite acceleration but is limited to pairs of robots. The works in [32] and [34] provide a means of determining the required force and cutoff that scales with the number of agents. However, from inspection and from some investigation, this either requires an increasingly impractical acceleration requirement or large buffer distance between robots.

Deadlock resolution is not yet explored for situations with heavy congestion. An example is when there are stationary agents that have reached their current goal blocking others moving to theirs. In [31] a single stationary obstacle is considered. In [12] agents are always moving between simulated delivery points. These delivery points may be shared by multiple agents at a time and congestion is resolved by a queuing mechanism. This mechanism may be used as inspiration for designing for other congestion scenarios such as the one mentioned earlier.

Limited sensing rate or communication bandwidth is a significant problem in real systems such as UAV swarms. The work in [39] introduced a self-triggered scheme but utilized an unbounded potential function.

The consideration of multiple requirements as discussed above leads to a complicated potential field design making future works more challenging. This can be observed in [32] and [38]. One of the aims of using this nature-inspired design in swarms is to produce some emergent behavior as a result of simple interactions. However, it is arguable that a deterministic path planning approach is more practical in real applications such as that seen in [47].

2.2.2 Partition-based Methods

The following methods are referred to as partition-based as they involve the formulation of convex constraints on some combination of relative position, velocity, acceleration or higher order dynamics. Control inputs are obtained by solving an optimization problem subject to those constraints at every time step. The key methods from literature are Voronoi Cells, Velocity Obstacles and Barrier Certificates.

2.2.2.1 Voronoi Cells

From [48], the Voronoi Cell for a point $\mathbf{q}_i \in \mathbb{R}^n$ relative to other points $\mathbf{q}_j$, $j \neq i$ is

$$\mathcal{V}_i = \{\mathbf{q} \in \mathbb{R}^n \mid \|\mathbf{q} - \mathbf{q}_i\| \leq \|\mathbf{q} - \mathbf{q}_j\|, \forall j \neq i\}, \quad (2.12)$$

where $i, j \in \mathbb{Z}^+$ are indices representing the positions of agents i and j . $\mathcal{V}_i$ can be interpreted as the subspace of positions within which an agent i at point $\mathbf{q}_i$ can remain without colliding with another. Only position information is required as the space is obtained by partitioning along the plane equidistant between two agents. For robots with 2nd or higher order models, trajectory planning is required

to ensure that partitions are not violated. However, in highly congested scenarios, non-violation of partitions is not easily guaranteed.

The Voronoi diagram obtained from the partitions has also been used for safe path planning [49, 50]. The diagram shows the complete partitioning of a space according to some points $\mathbf{q}_i$. By assigning points $\mathbf{q}_i$ to obstacles, a graph is obtained with partition lines as edges representing the safest points between obstacles. An algorithm such as Fast Marching can then be used to find a safe path through. The method does not consider avoidance of other agents and requires extensive prior knowledge of the environment.

Voronoi Cells are also typically used for coverage control, where sensor-carrying robots are to be deployed optimally to cover some objective or area [51, 52]. Robots may be driven to maximize sensor range usage while avoiding overlaps using Voronoi Cell information. CA was not explicitly considered if the agents have finite size in [51]. In [52], it was noted that when CA is implemented, the resulting coverage may not be optimal.

The more recent work in [53] extends the formulation in (2.12) by adding a buffer to ensure a minimum separation distance between finite-sized agents. However, the Voronoi Cell method is prone to deadlock when agents become closely packed. An algorithm is provided that detects these deadlocks and forces the agent to move parallel to the partition plane. This results in a last-in-first-out resolution where agents at the center of a congestion may remain deadlocked until others are cleared. The authors compared their method with the Optimal Reciprocal CA algorithm from [54]. Similar performance was observed in terms of execution steps in scenarios where agents are required to move to some assigned positions while avoiding collision. For agents of 2nd or higher order, a crude solution is provided where the minimum separation distance is made larger than the robot size without explicit guarantee of non-collision. Robots are allowed to violate the partitions but are driven back when it occurs.

In [55], the authors proposed an alternative method to buffering the Voronoi partition by using dynamic weights for obstacles of varying sizes. The work focuses instead on target tracking using single-integrator agents in a cluttered environment. While collision guarantee is provided, avoidance of deadlocks due to obstacles forming barricades is not tackled.

2.2.2.2 Velocity Obstacles

The Velocity Obstacle (VO) concept was first introduced in [56] for motion planning among obstacles with known linear trajectories and later generalized to static and moving obstacles in [57]. The collision problem is considered in velocity space by defining a cone of collision relative velocities, known as the VO, between a robot and an obstacle represented by a sphere. A collision is avoided by selecting a relative velocity outside of the VO. A similar work using collision cones but for irregular shaped objects was reported in [58].

The method was then extended for cooperative avoidance between robot pairs in [19], which eliminated oscillations inherent in the original method. The formulation in [19] does not allow temporary head-on relative velocities in the multi-agent case, which may result in situations with no feasible velocity. This is rectified in the Optimal Reciprocal CA algorithm in [54], which increases the likelihood that a feasible solution exists.

Based on [54], the VO encountered by agent i with respect to agent j for some time horizon τ is

$$VO_{i|j} = \{\mathbf{v} \mid \exists t \in [0, \tau], \|\mathbf{v}t - (\mathbf{q}_j - \mathbf{q}_i)\| < d_{\min}\}, \quad (2.13)$$

where $i, j \in \mathbb{Z}^+$, $i \neq j$ and $d_{\min}$ is the minimum safe separation distance. This is illustrated in Figure 2.2(a). The superscript "opt" is used to denote the agent's current velocity or a preferred velocity based on an objective. This choice is left to the designer and depends on the availability of the information. The vector

FIGURE 2.2: $VO_{i|j}^\tau$ and $ORCA_{i|j}^\tau$ in relative velocity and position space superimposed. (a) $VO_{i|j}^\tau$ is represented by the shaded truncated cone. $\mathbf{v}_i^{\text{opt}}$ and $\mathbf{v}_j^{\text{opt}}$ may represent respective current velocities or preferred velocities based on objective and is chosen by the designer. $\mathbf{u}_{i|j}$ represents the minimum change required to bring the relative velocity to the boundary of $VO_{i|j}^\tau$. (b) $ORCA_{i|j}^\tau$ is shown shaded in blue. If $\mathbf{v}_i \in ORCA_{i|j}^\tau$ and $\mathbf{v}_j \in ORCA_{j|i}^\tau$, then collision is avoided.

$\mathbf{u}_{i|j}$ represents the minimum change required to bring the relative velocity to the boundary of $VO_{i|j}^\tau$. This is used to obtain convex sets for $\mathbf{v}_i$ and $\mathbf{v}_j$ that would avoid collision. From [54], the set for i given j is

$$ORCA_{i|j}^\tau = \left\{ \mathbf{v} \left| \left(\mathbf{v} - \left(\mathbf{v}_i^{\text{opt}} + \frac{1}{2} \mathbf{u}_{i|j} \right) \right)^\top \frac{(-\mathbf{u}_{i|j})}{\|\mathbf{u}_{i|j}\|} \geq 0 \right. \right\}, \quad (2.14)$$

which is shown in Figure 2.2(b). For agent j , $ORCA_{j|i}^\tau$ is obtained in the same manner and overlaps only on the boundary with $ORCA_{i|j}^\tau$. Each pair of agents selecting velocities within their respective $ORCA$ will thus avoid collision up to horizon τ . The $ORCA_{i|j}^\tau$ is optimal due to maximizing the available velocities near its $\mathbf{v}_i^{\text{opt}}$. The result is a partition on velocity space between agents i and j . In the multi-agent case, an optimization problem is solved to obtain a suitable velocity given all ORCA constraints. However, finite acceleration was not considered.

An infeasible optimization problem can still occur in congested multi-agent cases as the $ORCA_{i|j}^\tau$ obtained does not always include zero as shown in Figure 2.2(b). In such cases, the solution given is to choose the "safest" velocity that minimizes the maximum of distances to ORCA constraint boundaries.

A similar method was reported in [59], where the points of closest approach of a pair of agents is used to determine the required avoidance effort. These points are also determined using relative velocity and position. Ultimately, the method is conceptually the same as [54], but the avoidance effort is divided instead according to agent's current velocity and requires more effort from the slower agent. It was also implemented in simulation of a pair of UAVs with unicycle dynamics. This was extended in [60] with consideration for heading rate constraint using a differential flatness-based trajectory planner but only for a pair of UAVs. An extension using interval analysis to handle sensor uncertainty was reported in [61]. Another method similar to [59] and [60] uses the proportional navigation missile guidance approach instead to guide the UAV towards the boundary of minimum required separation distance [62]. A negative relative velocity is directed towards the boundary instead of along the relative position vector. Likewise, only the two UAV case is considered but with a heading rate constraint.

The Velocity Obstacle has also been extended for non-holonomic models and higher order dynamics. Velocity obstacles for non-holonomic robots were considered in [63], [64] and [65]. Acceleration-velocity obstacles were introduced in [66] with constrained acceleration and can be used with non-holonomic systems such as car-like robots. Various simulations show the method's effectiveness but the authors noted that infeasible situations can still exist in heavy congestion where the resulting safe velocity set is empty. This was subsequently generalized for $\mathcal{C}^n$ motion continuity in [67]. In both [66] and [67], a linear constraint on velocity is still obtained like in the ORCA method but with additional consideration for reachable velocities. A simpler approach using trajectory buffering is presented in [68], where a reference linear trajectory obtained using the method from [54] is buffered by a cylinder to account for the maximum lateral deviation due to tracking error of an acceleration-constrained robot. A 2D version for non-holonomic vehicles was also reported more recently in [69].

Recently reported works include goal-related coordination [70], uncertainty in measurement [71, 72], elliptical instead of spherical robot shape [73], and limited field

of view [74, 75].

2.2.2.3 Barrier Certificates

Barrier certificates have also been used to establish linear constraints on acceleration inputs to avoid collision in [76] and [77]. The barrier certificate also referred to as the barrier function maps a safe set of states to a non-negative scalar value. This property is used to obtain a continuous controller that ensures forward invariance of the safe set. In those works, the linear constraint is formulated to ensure that a pair of agents can decelerate at maximum rate from the chosen relative velocity to avoid collision.

A variant called the zeroing control barrier function (ZCBF) is used in [76] and [77]. To define the ZCBF candidate according to [78], a general linearly controlled system is first considered,

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})u, \quad (2.15)$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state, $\mathbf{u} \in U \subset \mathbb{R}^m$ is the control input, and f and g are locally Lipschitz continuous. Subsequently, a set of safe states is defined as $\mathcal{C} \subset \mathbb{R}^n$ for which forward invariance is desired. The ZCBF candidate, $h : \mathbb{R}^n \rightarrow \mathbb{R}$, is such that

$$\mathcal{C} = \{\mathbf{x} \in \mathbb{R}^n \mid h(\mathbf{x}) \geq 0\}. \quad (2.16)$$

Definition 2.2.1. [78] A continuous function $\kappa : (-b, a) \rightarrow \mathbb{R}$ for some $a, b > 0$ is an extended class- $\mathcal{K}$ function if it is strictly increasing and $\kappa(0) = 0$.

Definition 2.2.2. [78] Given the set (2.16) for a smooth function $h : \mathcal{D} \rightarrow \mathbb{R}$, where $\mathcal{C} \subseteq \mathcal{D} \subset \mathbb{R}^n$, h is called a ZCBF if there exists an extended class- $\mathcal{K}$ function κ such that

$$\sup_{\mathbf{u} \in U} \left\{ \dot{h}(\mathbf{x}) + \kappa(h(\mathbf{x})) \right\} \geq 0 \quad (2.17)$$

for all $\mathbf{x} \in \mathcal{D}$, where

$$\dot{h}(\mathbf{x}) = \frac{\partial h(\mathbf{x})}{\partial \mathbf{x}} \dot{\mathbf{x}}. \quad (2.18)$$

The function κ modifies the speed of approaching the boundary of set $\mathcal{C}$.

The control space is thus

$$S(\mathbf{x}) = \left\{ \mathbf{u} \in U \mid \dot{h}(\mathbf{x}) + \kappa(h(\mathbf{x})) \geq 0 \right\}, \quad \mathbf{x} \in \mathcal{D}. \quad (2.19)$$

Any Lipschitz continuous controller $\mathbf{u} : \mathcal{D} \rightarrow \mathbb{R}$ where $\mathbf{u} \in S(x)$ guarantees $\mathcal{C}$ to be forward invariant and asymptotically stable in $\mathcal{D}$. This is formalized in theorems in [77] and [78].

To apply this method to CA of robots modeled as double integrators with finite acceleration in [77], the following ZCBF is defined for robot i with respect to robot j .

$$h_{ij} = \sqrt{4a_m(\|\mathbf{q}_i - \mathbf{q}_j\| - d_{\min})} + \frac{(\mathbf{q}_i - \mathbf{q}_j)^\top}{\|\mathbf{q}_i - \mathbf{q}_j\|} (\mathbf{v}_i - \mathbf{v}_j), \quad (2.20)$$

where a_m is the maximum absolute acceleration of each robot. The formulation of h_{ij} is based on kinematic equations and the distance required to decelerate at a_m rate to zero velocity along the relative position vector. The method is cooperative but can be modified for non-accelerating obstacles. The result of this method is that robot i is allowed to accelerate towards j at a rate greater than $-a_m$ as long as $h_{ij} > 0$.

In the distributed multi-agent case, the robot's control input is obtained by quadratic program using pairwise (2.19) as constraints. In some cases, the program may become infeasible. Therein a conservative modification to (2.20) is given to ensure that deceleration to zero absolute velocity is always possible without collision. Deadlocks were overcome by using perturbations on the nominal input before solving the program.

In [79], CA of UAVs modeled as fourth order integrator was tackled by using an exponential barrier function. However, the resulting constraint is used to modify a preplanned trajectory by iteratively solving a quadratic program. The input is directly constrained but other constraints such as for velocity are only met after iteration by reparameterizing the trajectory. More recent developments include

application to lane keeping for cars [80], fixed-wing aircraft by prediction over a horizon [81], and 3D ellipsoidal agents with uncertainty [82].

2.2.2.4 Summary of Partition-based Methods

Partition-based methods are relatively easier to design compared to repulsion-based when dynamic constraints are considered. However, partition-based methods require solving an optimization problem subject to linear constraints for the control input. The main works as discussed earlier such as [53], [54], [68] and [77] involve, in some similar form, the solving of a quadratic program with linear constraints to minimize the deviation of the control input from the nominal for one time step (i.e. $\min \|\mathbf{u} - \mathbf{u}_{\text{nominal}}\|$). Such programs can be solved in $O(n)$ time, where n is the number of linear constraints [54]. Thus computational power requirement may not be a concern.

A variety of concerns relevant to physical implementation have been addressed such as acceleration constraints and more complex dynamic models such as fixed wing aircraft. We highlight two other relatively open research problems with regards to the partition-based method in the following.

The first of these problems is the issue of limited bandwidth in communication. In applications such as UAVs, obtaining position and velocity information of other UAVs by proximity sensors or cameras may not be reliable. Thus communication can be used instead. To the best of our knowledge, none of the works in literature have directly addressed communication constraint in the design of the partition-based algorithms.

The second is deadlock resolution. A deadlock occurs when an agent does not move due to the combination of constraints and a non-zero nominal input. This is not desirable as the agent cannot complete its objective. In the Voronoi Cell method from [53], the deadlock resolution proposed resulted in a last-in-first-out resolution where agents at the congestion center may remain stuck until the surrounding

agents are cleared. A first-in-first-out resolution may be desired instead. In contrast, reported VO methods did not specifically address the deadlock problem. In [68], the authors noted that it may theoretically occur but was not encountered in simulations. In the Barrier Certificate method from [77], deadlocks were resolved by perturbing the nominal input before optimization. However, similar to repulsion-based works, the issue of heavy congestion involving blocking stationary agents is not fully explored.

While some solutions have been provided, they have not yet been subjected to heavier congestion such as when agents' assigned objective positions are placed close to each other. The problem may also become amplified when the limited communication constraint is also considered. Thus further investigation of deadlock resolution methods may be warranted.

2.2.3 Trajectory Replanning

Trajectory replanning refers to the adjustment or regeneration of trajectories to avoid collisions with environmental obstacles or other robots as it moves to its assigned goal or tracks a nominal trajectory. This is typically done by solving for a control input sequence by minimizing a cost function subject to dynamic, input, and collision constraints [25]. Unlike the previous two categories, methods here are designed to generate trajectories for some time horizon instead of for just the next time step. However, not all methods mentioned here were specifically formulated for online implementation. It is noted in literature, such as in [53] and [77], that trajectory replanning is relatively more computationally intensive than the previous two categories making it unsuitable for online distributed implementation in swarms.

2.2.3.1 Mixed Integer Methods

One of the more common approaches is to use mixed integer linear programming (MILP), in which collisions are modeled using linear constraints with binary variables. All of the following mixed integer works mentioned here use centralized computing. One of the earlier examples is [83]. Predefined trajectories of a group of robots are reparameterized using MILP to avoid collisions while minimizing overall completion time. Paths are unchanged but start times of each trajectory are adjusted. Binary variables are used to indicate whether a robot crosses a collision point before or after the other robot.

A more common way to model collision constraints using MILP was introduced in [84] by approximating obstacles and other robots with polygons and using binary variables to ensure that at least one edge constraint of the polygon is satisfied. Specifically, only one linear constraint with its boundary on the edge of a polygon enclosing the obstacle needs to be satisfied to ensure non-collision. The trajectory is generated using model prediction while the MILP minimizes fuel usage. This method was also applied to spacecraft with additional relevant considerations such as fuel plume impingement [85]. A non-uniform discretization of the time intervals was then introduced in [86] using an iterative method to reduce computation while still ensuring that trajectories do not pass through obstacles. This reduced computation time by considering only time instances where the robot would be near obstacles.

The use of binary variables with polygonally approximated obstacles was also applied to trajectory generation for multiple quadrotor robots in [87]. The piecewise polynomial trajectory generation method first introduced in [25] was extended to include CA of other quadrotors using mixed integer quadratic programming (MIQP). High order polynomials are used to interpolate between a prespecified number of time steps. Constraints enforce smoothness up to a k_r th derivative. The MIQP then minimizes the k_r th derivative. The authors noted that the method does

not scale well with number of quadrotors but can generate optimal trajectories for small teams.

2.2.3.2 Sequential Convex Programming

A decentralized approach to trajectory generation for a swarm of spacecraft was proposed in [88] using sequential convex programming (SCP) and implemented using model predictive control (MPC). MPC is used to compute control for a trajectory over a finite horizon. Resulting control inputs are applied until the next computation is completed. With SCP, each agent iteratively solves an optimization problem to minimize fuel and obtain a trajectory free of collisions towards its assigned goal. Each agent initializes the synchronized iteration with a guess on a nominal trajectory and transmits it to nearby neighbors. The relative trajectory is used to formulate affine collision constraints at each time step tangent to the sphere enclosing the other agent. The new trajectory from the iteration is then transmitted again and the process is repeated until the error between the new and previous trajectories are below some threshold. To avoid oscillation in the solution, each agent only considers collision constraints with agents whose identity numbers are smaller.

Subsequently in [89], a distributed auction assignment algorithm for target assignment was integrated with the method from [88]. Experiments with quadrotor robots were also reported but was conducted using a central computer. The approach allows for decentralized trajectory generation with CA but requires iteration and extensive communication of trajectories.

2.2.3.3 Other Methods

Other more recent methods that do not fit the previous two categories are presented here.

In [90], a centralized geometric algorithm is used to generate first order linear trajectories which are later refined by quadratic programming to obtain smooth n th-order dynamic trajectories. The algorithm operates in polynomial time. Decentralization is left to future work.

In [91], a token-based distributed coordination algorithm of fixed wing aircraft was reported. Tokens based on criticality order are transmitted to adjacent neighbors on the network to determine the sequence in which agents decide on one of three pre-generated trajectories to avoid collision. The agent then passes on the tokens along with their chosen trajectories to the next ones with lower criticality. Essentially, the agents take turns to select safe trajectories based on those chosen by the higher criticality aircraft. Future work includes consideration of more complex scenarios with clustered aircraft.

In [92], a decentralized MPC method is described which incorporates potential field repulsion in the robot dynamics for CA instead of using the usual linear constraints from previous works. Repulsion from other agents are calculated based on the horizon predictions shared by other agents. Evaluation was done by simulation with real experiments left to future work.

In [93], a method similar to [87] was proposed that uses quadratic programming with a dynamic receding horizon to obtain spline parameters for the trajectories. The method uses instead the buffered voronoi cells formulation of [53] to provide collision constraints. It is distributed as it only requires sensing of other robot's positions. A graph search is used to avoid deadlocks and the trajectories are generated based on the obtained path. However, the authors note that the program's feasibility cannot be formally guaranteed for dynamics of higher than first-order integrator but their experiments show that the robots do not collide as the program becomes feasible again in subsequent steps. Future work would involve flying robots and addressing the infeasibility issue.

2.2.3.4 Summary of Trajectory Replanning Methods

The discussed methods demonstrate superiority in accounting for dynamic constraints but at the cost of requiring more computation as well as communication in the decentralized cases. The increased computation requirement is caused by the increased number of decision variables due to optimizing over multiple time steps. This may be mitigated in cases where precomputed trajectory primitives are used such as in [91].

For decentralized cases, works such as [89], [91] and [92] require transmission of trajectory information. This may be unrealistic where communication bandwidth is limited such as with UAVs. However, it is noted that the authors of [89] did attempt to reduce this requirement by using a smaller horizon with MPC. If communication is not required and other agents can be detected by proximity sensors, such as in [93], then trajectory replanning may be safely utilized.

In this thesis, trajectory planning methods are not considered to reduce the scope as well as due to the aforementioned issues of heavy computation and communication requirement. Moreover, we leave trajectory planning methods as higher level planners and instead focus on more reactive low level algorithms for CA.

2.3 Summary

The brief review of quadrotor dynamics showed how quadrotor UAVs move in 3D space. The quadrotor rotates on the roll and pitch axis to provide lateral acceleration. [25] showed that quadrotors are differentially flat in outputs $[\mathbf{q}, \psi]$. This allows for approximation of the quadrotor model by double or triple integrators given suitable assumptions on velocity, acceleration and jerk bounds, or time responses of control loops.

Three categories of CA methods were reviewed: repulsion-based, partition-based and trajectory replanning. Some trajectory replanning methods were reviewed but

the category is not considered in this thesis to limit its scope. This is mostly due to the relatively heavier computation and communication requirement which may not be suitable for UAV swarms with small and lightweight agents. The main issues of the first two categories are summarized in the following.

The two main common issues are the lack of consideration for limited bandwidth in communication, and deadlock resolution in heavier congestion.

Limited bandwidth in communication is a significant issue as highlighted in Section 1.1. Most works assume that the position information of other agents are obtainable reliably at every time step. However, if wireless communication is to be used between aerial swarm agents, then algorithms need to be designed to be robust to slower communication rates, delays and dropouts. Two recent works by the same author have attempted to address this issue in [37, 39], but still require some periodic communication to ensure non-collision.

Deadlock resolution has been tackled in CA works to varying degrees. In repulsion-based works, it is done by adding a gyroscopic force perpendicular to the repulsion. In partition-based works, it is done by scenario identification and forcing a move parallel to the partition in the Voronoi case or by perturbation in the Barrier Certificate case. Deadlocks in the VO case were acknowledged but not encountered in simulations. However, deadlock resolution in heavily congested scenarios are not fully explored. Moreover, the problem becomes exacerbated when limited or slower rates of communication is used. This is highlighted again later in Chapter 4 and 5. A step in this direction exists in [12] where a distributed queuing mechanism was introduced for agents with conflicting delivery positions.

For repulsion-based CA, other open issues include the problem of scalability due to finite acceleration, and that repulsion has only been primarily reported using position. For the latter, a direction-based repulsion mechanism is presented in the next chapter.

Chapter 3

LOS Repulsion for Multi-UAV Intruder Interception

This chapter proposes a novel collision avoidance (CA) method by repulsion based on the line of sight (LOS) between an interceptor UAV and its target or other targets. The aim of the resulting control law is to guarantee that collisions with other interceptor UAV or targets that it was not assigned do not happen while minimising disruption to the interception process. The content as presented here is primarily based on the works published in [22–24].

The purpose of this chapter is two-fold. Firstly, a novel CA method given the specific scenario is presented. Secondly, the issues highlighted at the end of this chapter give motivation to explore other CA methods as well as the issue of limited communication in the subsequent chapters.

This chapter is organized as follows. The following section further describes the motivation for tackling this scenario and the choice of method based on the literature. Section 3.2 presents the problem formulation. The guidance law for single UAV interception of one intruder is first described in Section 3.3 as a preliminary to the full distributed version with CA. Simulation and experiment results are also

presented therein. The distributed guidance law with CA is then presented in Section 3.4 with simulation and experiment results. Concluding remarks are given in the chapter summary in Section 3.5.

3.1 Motivation

FIGURE 3.1: Interception scenario. An example is shown where the intruding UAV is detected by radar and an interceptor is sent to capture it.

As mentioned in the introduction, the proliferation of UAV or drone technology brings with it a challenge to defend against its misuse by members of the public or hostile organizations. UAVs may be used to intrude into airports to disrupt traffic, prisons to smuggle contraband or key government installations to spy or cause damage. This challenge is ongoing and is evident in a recent call for proposals by the Defense Science Technology Agency of Singapore [94]. One possible solution (see Figure 3.1), is to use UAVs carrying a net or other similar non-destructive neutralizing device to capture the intruding UAVs. This reduces or prevents any collateral damage if the operation is to be carried out in a civilian area.

In this work, the use of quadrotor UAVs as interceptors is considered. Such UAVs may be considered holonomic in 3D space and are thus more versatile than other forms such as fixed wing UAVs.

We also envision that a ground-based radar might be used to obtain and broadcast the position and velocity data of intruders and interceptors such as shown in Figure 3.1. The scope of work in this chapter is limited to the assumption that such data is reliably obtained with no time delays or dropouts. However, it is possible that in real applications additional onboard sensing methods may be required to work in tandem with the ground radar or other systems.

The interception controller in this chapter is designed based on missile guidance laws. Classic guidance laws include pure pursuit, deviate pursuit, and proportional navigation (PN) among others [95]. Of the available methods, PN is the most popular due to simplicity and energy-efficiency [96].

The CA issue comes when multiple intruders exist. The simplest solution to the multiple intruder case is to use multiple interceptor UAVs. Each interceptor is then assigned one intruder and should complete interception without colliding with other targets or interceptors. CA for multiple UAVs is usually based on relative distances as shown in the literature review. In this work, direction-based avoidance is used to keep the LOS between the interceptor and its target clear of obstacles. Direction-based CA is studied in [97], but the technique is heuristic, and no strict proof is provided. Relevant results are limited and some results can be found in [98].

3.2 Problem Formulation

A double integrator model is assumed for the intruding targets and the interceptor UAVs. For the single intruder case the model is defined as follows.

$$\begin{cases} \dot{\mathbf{q}}_T &= \mathbf{v}_T, \\ \dot{\mathbf{v}}_T &= \mathbf{a}_T, \end{cases} \quad (3.1)$$

where $\mathbf{q}, \mathbf{v}, \mathbf{a} \in \mathbb{R}^3$ are respectively the position, velocity and acceleration, and the subscript T or I denotes the target or interceptor respectively. The position of the

target relative to the interceptor is defined as

$$\mathbf{q}_{TI} = \mathbf{q}_T - \mathbf{q}_I \quad (3.2)$$

Assumption 3.1. The inner (or attitude) loop control of the interceptor rotary UAV is designed and implemented perfectly such that the settling time to reach a desired acceleration is finite and insignificantly small relative to the outer loop velocity control time-response.

Remark 3.1. The time-response of the attitude control of the quadrotor can be tuned much faster than the outer loop velocity control. Given appropriate operation parameters such as velocity and acceleration, the inner-loop dynamics may be neglected in high-level design as is typical in guidance design.

Assumption 3.2. The uncertain time-varying target acceleration is bounded: $\|\mathbf{a}_T\| < \bar{a}_T$, where $\bar{a}_T > 0$ is known prior.

Positions and velocities of all targets and interceptors are also assumed to be continuously available to each interceptor.

The objective in the single intruder case is to design a guidance law $\mathbf{a}_I$ so that the interceptor UAV globally intercepts the target in finite time with non-zero relative velocity, i.e. $\|\mathbf{q}_{TI}(t_f)\| = 0$, $\|\dot{\mathbf{q}}_{TI}(t_f)\| > 0$, where $t_f < +\infty$.

Then for N targets and N interceptors, the notation is as follows.

$$\begin{cases} \dot{\mathbf{q}}_i^j &= \mathbf{v}_i^j, \\ \dot{\mathbf{v}}_i^j &= \mathbf{a}_T^j - \mathbf{a}_{Ii}, \end{cases} \quad (3.3)$$

where $\mathbf{q}_i^j = \mathbf{q}_T^j - \mathbf{q}_{Ii}$, $\mathbf{v}_i^j = \mathbf{v}_T^j - \mathbf{v}_{Ii}$, subscripts T and I still denote target and interceptor respectively, and $i, j \in \{1, 2, \dots, N\}$. Similarly, for relative dynamics between targets,

$$\begin{cases} \dot{\mathbf{q}}^{jk} &= \mathbf{v}^{jk}, \\ \dot{\mathbf{v}}^{jk} &= \mathbf{a}_T^j - \mathbf{a}_T^k, \end{cases} \quad (3.4)$$

where $\mathbf{q}^{jk} = \mathbf{q}_T^j - \mathbf{q}_T^k$, $\mathbf{v}^{jk} = \mathbf{v}_T^j - \mathbf{v}_T^k$, and $j, k \in \{1, 2, \dots, N\}$, $j \neq k$.

The objective for the multi-intruder case is similar: design a guidance law $\mathbf{a}_{I_i}$ for each interceptor UAV to intercept its assigned target without inter-agent collisions or interception of targets other than that assigned. The latter is referred to as mis-interception. It is assumed that the numbers of targets and interceptors are equal.

3.3 Three-Dimensional Guidance Law for Single Quadrotor UAV Interception

3.3.1 Guidance Law

The PN method is used for the guidance law presented in this section. A velocity feedback component is appended to address target acceleration and achieve the non-zero hitting velocity.

FIGURE 3.2: Proportional Navigation

In PN, interception is achieved by matching the interceptor's velocity perpendicular to the line of sight (LOS) with the target and reducing the magnitude of the

distance between. Based on Figure 3.2, this requires

$$\|\mathbf{v}_I\| \sin \delta = \|\mathbf{v}_T\| \sin \theta, \quad (3.5)$$

$$\|\mathbf{v}_T\| \cos \theta - \|\mathbf{v}_I\| \cos \delta < 0. \quad (3.6)$$

For the rest of this chapter the LOS unit vector from the interceptor to the target is defined as

$$\boldsymbol{\lambda}_{TI} = \frac{\mathbf{q}_{TI}}{r_{TI}}, \quad (3.7)$$

where $r_{TI} = \|\mathbf{q}_{TI}\|$. Its magnitude is 1 except when the relative distance is zero at collision. $\dot{\boldsymbol{\lambda}}_{TI}$ only changes the direction of $\boldsymbol{\lambda}_{TI}$, i.e. $\dot{\boldsymbol{\lambda}}_{TI}^\top \boldsymbol{\lambda}_{TI} = 0$. The derivatives of $\boldsymbol{\lambda}_{TI}$ are

$$\dot{\boldsymbol{\lambda}}_{TI} = \frac{\dot{\mathbf{q}}_{TI} r_{TI} - \mathbf{q}_{TI} \dot{r}_{TI}}{r_{TI}^2}, \quad (3.8)$$

$$\ddot{\boldsymbol{\lambda}}_{TI} = \frac{\ddot{\mathbf{q}}_{TI} - 2\dot{\boldsymbol{\lambda}}_{TI} \dot{r}_{TI} - \boldsymbol{\lambda}_{TI} \ddot{r}_{TI}}{r_{TI}}. \quad (3.9)$$

Since $\ddot{\mathbf{q}}_{TI} = \mathbf{a}_T - \mathbf{a}_I$, to drive $\dot{\boldsymbol{\lambda}}_{TI} \rightarrow 0$ which then satisfies the condition (3.5) the PN command acceleration can be designed as

$$\mathbf{a}_I = (-2\dot{r}_{TI} + k_1) \dot{\boldsymbol{\lambda}}_{TI}, \quad (3.10)$$

where $k_1 > 0$ is the control gain.

Proposition 3.1. *Consider the interception model (3.1) and (3.2), and suppose that Assumption 3.1 and Assumption 3.2 are satisfied. With the guidance law (3.10), the variation of LOS is ultimately bounded with tunable ultimate bounds. Moreover, if the target acceleration is along the LOS, and initial relative velocity satisfies $\mathbf{v}_{TI}(0)^\top \boldsymbol{\lambda}_{TI}(0) + \bar{a}_T < 0$, then perfect interception can be guaranteed.*

Proof. Select a Lyapunov function candidate $L = \frac{1}{2} \dot{\boldsymbol{\lambda}}_{TI}^\top \dot{\boldsymbol{\lambda}}_{TI}$. Its time derivative is given by

$$\dot{L} = \dot{\lambda}_{TI}^\top \ddot{\lambda}_{TI} = -\frac{k_1}{r_{TI}} \dot{\lambda}_{TI}^\top \dot{\lambda}_{TI} + \frac{1}{r_{TI}} \dot{\lambda}_{TI}^\top \mathbf{a}_T \quad (3.11)$$

This indicates that $\dot{\lambda}_{TI}$ converges to $\left\| \dot{\lambda}_{TI} \right\| \leq \frac{\|\mathbf{a}_T\|}{k_1} < \frac{\bar{a}_T}{k_1}$. Moreover, if $\mathbf{a}_T$ is along the LOS, and $\mathbf{v}_{TI}(0)^\top \lambda_{TI}(0) + \bar{a}_T < 0$, then perfect interception is guaranteed at finite time, t_f .

Remark 3.2. The PN guidance law guarantees the interception, if the target acceleration is along the LOS, and initial velocity along the LOS is negative. Hence, PN is only a local guidance law that is insufficient for intercepting a highly maneuvered target.

To account for the target's acceleration, the global integrated guidance law can be designed by

$$\mathbf{a}_I = (-2\dot{r}_{TI} + k_1) \dot{\lambda}_{TI} + \left(k_2 [(\dot{\mathbf{q}}_{TI}^\top \lambda_{TI}) - v_r] + (\dot{\mathbf{q}}_{TI}^\top \dot{\lambda}_{TI}) \right) \lambda_{TI}, \quad (3.12)$$

where $k_2 > 0$ is the control gain, and v_r is the constant reference relative speed.

Proposition 3.2. Consider the interception model (3.1) and (3.2), and suppose that Assumption 3.1 and Assumption 3.2 are satisfied. Using guidance law (3.12) with reference speed

$$v_r + \frac{\bar{a}_T}{k_2} < -\varepsilon, \quad (3.13)$$

where $\varepsilon > 0$, perfect interception can be guaranteed globally with finite time and non-zero hitting velocity.

Proof. Differentiating $r_{TI} = \mathbf{q}_{TI}^\top \lambda_{TI}$ yields

$$\dot{r}_{TI} = \dot{\mathbf{q}}_{TI}^\top \lambda_{TI} + \mathbf{q}_{TI}^\top \dot{\lambda}_{TI} = \dot{\mathbf{q}}_{TI}^\top \lambda_{TI}, \quad (3.14)$$

$$\ddot{r}_{TI} = \ddot{\mathbf{q}}_{TI}^\top \lambda_{TI} + \dot{\mathbf{q}}_{TI}^\top \dot{\lambda}_{TI} = (\mathbf{a}_T - \mathbf{a}_I)^\top \lambda_{TI} + \dot{\mathbf{q}}_{TI}^\top \dot{\lambda}_{TI}. \quad (3.15)$$

Substitute (3.12) into (3.15) to obtain

$$\ddot{r}_{TI} = -k_2 [(\dot{\mathbf{q}}_{TI}^\top \boldsymbol{\lambda}_{TI}) - v_r] + \mathbf{a}_T^\top \boldsymbol{\lambda}_{TI}. \quad (3.16)$$

The relative speed $(\dot{\mathbf{q}}_{TI}^\top \boldsymbol{\lambda}_{TI})$ thus converges to less than $\frac{\bar{a}_T}{k_2} + v_r < -\varepsilon < 0$. With $(\dot{\mathbf{q}}_{TI}^\top \boldsymbol{\lambda}_{TI}) < 0$, relative distance r_{TI} decreases to zero within finite time. Hence, interception is globally achieved along the LOS with a non-zero hitting velocity.

Remark 3.3. It should be noted that, for interception, $\dot{\boldsymbol{\lambda}}_{TI} \rightarrow 0$ is not strictly required; the interception is guaranteed by strict negativeness of $\dot{r}_{TI}$. However, tunable bounded $\dot{\boldsymbol{\lambda}}_{TI}$ is necessary to avoid circular relative motion.

3.3.2 Simulation

In the simulation, an interceptor UAV modeled by a double integrator like in (3.1) is tasked with intercepting a target that has the same model and is moving with bounded acceleration. The interceptor is controlled using guidance law (3.12) and is allowed to move unconstrained in 3D space. The initial states are given in Table 3.1. The target moves with a constant acceleration. The control gains used are $k_1 = 4$ and $k_2 = 1$. The reference hitting velocity is $v_r = -5\text{m/s}$.

TABLE 3.1: Initial States of Interceptor and Target

$\mathbf{q}_I(0)$	$[0, 0, 0]^\top \text{m}$	$\mathbf{q}_T(0)$	$[-22, -10, 34]^\top \text{m}$
$\mathbf{v}_I(0)$	$[0, 0, 0]^\top \text{m/s}$	$\mathbf{v}_T(0)$	$[5, 6, 0]^\top \text{m/s}$
$\mathbf{a}_I(0)$	$[0, 0, 0]^\top \text{m/s}^2$	$\mathbf{a}_T(0)$	$[-1.2, 1.0, 0.2]^\top \text{m/s}^2$

Figure 3.3 provides the results of the simulation showing relative position convergence with nonzero hitting velocity within finite time. Figure 3.3(c) shows the relative speed along the LOS and is non-zero at interception.

FIGURE 3.3: Simulation of integrated guidance controller. Interception is achieved within finite time.

3.3.3 Experiment

To demonstrate the integrated guidance law, two near identical quadrotor UAVs (see Figure 2.1(a)) were built and programmed. The original scenario for interception involves the use of radar to track the intruder and transmits the data to the intercepting UAV. The radar is emulated by having the target track its position and velocity with GPS (Global Positioning System) and transmit it to the interceptor via 2.4GHz XBee transceiver radios [99] at 50Hz, which is also the frequency of the controller. Acceleration commands were mapped to attitude commands based on level flight, which are then limited to 35° .

FIGURE 3.4: Experiment of integrated guidance controller

The experiment was conducted outdoors on a level empty field. For safety, the experiment was limited to two-dimensional movement, and the interceptor flew at

3m altitude, 1m higher than the target. The target moves at a constant 3m/s and starts at about 15m North and 15m East of the interceptor.

Interception is deemed completed when the interceptor is within some predefined radius of the target. The interceptor stops and transmits a command to the target to also stop.

Figure 3.4 shows a successful interception with the use of an EKF (extended Kalman filter). The EKF was used to estimate states in the presence of stochastic factors such as random target maneuver, measurement noise and time delay. The predefined interception radius is 2m.

3.4 Multi-Interception with LOS-based CA

3.4.1 LOS Repulsion CA

CA is applied by repulsion along relative LOS with other targets or interceptors. The potential field method described in [28] and presented in (2.9) is used here but applied to the relative LOS with respect to every other interceptor and target.

FIGURE 3.5: Relative LOS.

First define the relative LOS for target i and interceptor i with respect to another interceptor j as

$$\lambda_{ij}^i = \lambda_i^i - \lambda_j^i, \quad (3.17)$$

and the relative LOS for target i and interceptor i with respect to another target k as

$$\boldsymbol{\lambda}_i^{ik} = \boldsymbol{\lambda}_i^i - \boldsymbol{\lambda}^{ik}, \quad (3.18)$$

where $\boldsymbol{\lambda}_j^i = \frac{\mathbf{q}_j^i}{r_j^i}$, $\boldsymbol{\lambda}^{ik} = \frac{\mathbf{q}^{ik}}{r^{ik}}$, $r_j^i = \|\mathbf{q}_j^i\|$, $r^{ik} = \|\mathbf{q}^{ik}\|$ and $i, j, k \in \{1, 2, \dots, N\}$. N is the number of intruders and the number of interceptors is assumed to be equal. Note that the relative positions $\mathbf{q}_j^i$ and $\mathbf{q}^{ik}$ are defined according to (3.3) and (3.4) respectively. The relative LOS are shown in Figure 3.5.

Collision and mis-interception is avoided by ensuring that the LOS between the interceptor and its assigned target is kept clear of other interceptors or targets. To ensure this, the following potential functions are utilized. The inter-interceptor potential function is defined as

$$\Phi_i(\boldsymbol{\lambda}_i^i, \boldsymbol{\lambda}_1^i, \dots, \boldsymbol{\lambda}_N^i) = \sum_{k=1, k \neq i}^N \phi(\boldsymbol{\lambda}_{ik}^i), \quad (3.19)$$

where

$$\phi(\boldsymbol{\lambda}_{ij}^i) = \left(\min \left[0, \frac{\|\boldsymbol{\lambda}_{ij}^i\|^2 - \Delta_1}{\|\boldsymbol{\lambda}_{ij}^i\|^2 - \delta_1} \right] \right)^2 \quad (3.20)$$

and $0 < \delta_1 < \Delta_1 \leq 2$. The inter-target potential function is defined as

$$\Psi_i(\boldsymbol{\lambda}_i^i, \boldsymbol{\lambda}^{i1}, \dots, \boldsymbol{\lambda}^{iN}) = \sum_{j=1, j \neq i}^N \psi(\boldsymbol{\lambda}_i^{ij}), \quad (3.21)$$

where

$$\psi(\boldsymbol{\lambda}_i^{ij}) = \left(\min \left[0, \frac{\|\boldsymbol{\lambda}_i^{ij}\|^2 - \Delta_2}{\|\boldsymbol{\lambda}_i^{ij}\|^2 - \delta_2} \right] \right)^2 \quad (3.22)$$

and $0 < \delta_2 < \Delta_2 \leq 2$.

To simplify the problem the following assumption is made which supersedes Assumption 3.2.

Assumption 3.3. The accelerations of the multiple targets are zeros, or namely $\mathbf{a}_T^i = 0$ ($i = 1, 2, \dots, N$), where N denotes the total number of targets.

Assumption 3.4. Initial positions of the interceptors and the intruders satisfy that $\|\boldsymbol{\lambda}_{ik}^i(0)\| > \delta_1$, $\|\boldsymbol{\lambda}_i^{ik}(0)\| > \delta_2$, where δ_1 and δ_2 are positive constants from (3.20) and (3.22) respectively, $i = 1, 2, \dots, N$, $k = 1, 2, \dots, N$, and $i \neq k$.

Remark 3.4. Assumption 3.4 implies that, interceptor UAVs are launched from different initial positions; and the same for the intruding targets.

3.4.2 Distributed Guidance Law

The resulting integrated guidance controller is

$$\begin{aligned}
\mathbf{a}_{Ii} = & (-2\dot{r}_i^i + k_1) \dot{\boldsymbol{\lambda}}_i^i + \left(k_2 [(\dot{\mathbf{q}}_i^{iT} \boldsymbol{\lambda}_i^i) - v_r] + (\dot{\mathbf{q}}_i^{iT} \dot{\boldsymbol{\lambda}}_i^i) \right) \boldsymbol{\lambda}_i^i \\
& + k_3 r_i^i \left(\frac{\partial \Phi_i}{\partial \boldsymbol{\lambda}_i^i} \right)^\top + k_3 r_i^i \sum_{j \neq i} \left[\left(\frac{\partial \Phi_i}{\partial \boldsymbol{\lambda}_j^i} \right) \frac{\dot{\boldsymbol{\lambda}}_j^i \dot{\boldsymbol{\lambda}}_i^{iT}}{\|\dot{\boldsymbol{\lambda}}_i^i\|^2} \right]^\top \\
& - k_3 r_i^i \left(\frac{\partial \Phi_i}{\partial \boldsymbol{\lambda}_i^i} \boldsymbol{\lambda}_i^i \right) - k_3 r_i^i \left(\sum_{j \neq i} \left(\frac{\partial \Phi_i}{\partial \boldsymbol{\lambda}_j^i} \right) \frac{\dot{\boldsymbol{\lambda}}_j^i \dot{\boldsymbol{\lambda}}_i^{iT}}{\|\dot{\boldsymbol{\lambda}}_i^i\|^2} \right) \boldsymbol{\lambda}_i^i \\
& + k_4 r_i^i \left(\frac{\partial \Psi_i}{\partial \boldsymbol{\lambda}_i^i} \right)^\top + k_4 r_i^i \sum_{j \neq i} \left[\left(\frac{\partial \Psi_i}{\partial \boldsymbol{\lambda}_j^i} \right) \frac{\dot{\boldsymbol{\lambda}}_j^i \dot{\boldsymbol{\lambda}}_i^{iT}}{\|\dot{\boldsymbol{\lambda}}_i^i\|^2} \right]^\top \\
& - k_4 r_i^i \left(\frac{\partial \Psi_i}{\partial \boldsymbol{\lambda}_i^i} \boldsymbol{\lambda}_i^i \right) - k_4 r_i^i \left(\sum_{j \neq i} \left(\frac{\partial \Psi_i}{\partial \boldsymbol{\lambda}_j^i} \right) \frac{\dot{\boldsymbol{\lambda}}_j^i \dot{\boldsymbol{\lambda}}_i^{iT}}{\|\dot{\boldsymbol{\lambda}}_i^i\|^2} \right) \boldsymbol{\lambda}_i^i,
\end{aligned} \tag{3.23}$$

where $k_i > 0$ ($i = 1, 2, 3, 4$) are control gains.

Remark 3.5. The physical implication of the direction-based inter-agent collision (or mis-interception) avoidance is that, if the i th interceptor (or target) never

appears along the LOS of the j th interceptor UAV, then the collision between the i th interceptor (or target) and j th interceptor would never occur.

Theorem 3.1. *Consider the system described by (3.3) and (3.4), where a multi-UAVs system is required to intercept multiple targets. Suppose that Assumptions 3.1, 3.3 and 3.4 are satisfied. If the command acceleration for distributed guidance is designed by (3.23) with control gains $k_i > 0$ ($i = 1, 2, 3, 4$) and reference speed $v_r \leq -\epsilon < 0$, then the following statements hold:*

1. *Each UAV is capable of intercepting its target;*
2. *There exist neither interagent collisions nor misinterceptions before interception.*

Proof. First consider the dynamics along the LOS of the i th interceptor with the i th target by substituting the distributed guidance law (3.23) into (3.3):

$$\begin{aligned} \ddot{r}_i^i &= \ddot{\mathbf{q}}_i^{i\top} \boldsymbol{\lambda}_i^i + \dot{\mathbf{q}}_i^{i\top} \dot{\boldsymbol{\lambda}}_i^i = -\mathbf{a}_{Ii}^{\top} \boldsymbol{\lambda}_i^i + \dot{\mathbf{q}}_i^{i\top} \dot{\boldsymbol{\lambda}}_i^i, \\ &= -k_2 \left[(\dot{\mathbf{q}}_i^{i\top} \boldsymbol{\lambda}_i^i) - v_r \right], \end{aligned} \quad (3.24)$$

because $\mathbf{a}_T^i = 0$ by Assumption 3.3, and the repulsion terms in the 2nd to 4th lines of (3.23) cancel when multiplied by $\boldsymbol{\lambda}_i^i$. The result implies that the relative velocity along the LOS with its target converges to the reference speed $v_r \leq -\epsilon < 0$.

Consider next the dynamics perpendicular to the LOS of interceptor i towards target i ,

$$\begin{aligned}
\ddot{\lambda}_i^i = & -\frac{k_1 \dot{\lambda}_i^i}{r_i^i} - \frac{\ddot{r}_i^i}{r_i^i} \lambda_i^i - \left(k_2 [(\dot{\mathbf{q}}_i^{i\top} \lambda_i^i) - v_r] + (\dot{\mathbf{q}}_i^{i\top} \dot{\lambda}_i^i) \right) \frac{\lambda_i^i}{r_i^i} \\
& - k_3 \left(\frac{\partial \Phi_i}{\partial \lambda_i^i} \right)^\top - k_3 \sum_{j \neq i} \left[\left(\frac{\partial \Phi_i}{\partial \lambda_j^i} \right) \frac{\dot{\lambda}_j^i \dot{\lambda}_i^{i\top}}{\|\dot{\lambda}_i^i\|^2} \right]^\top \\
& - k_4 \left(\frac{\partial \Psi_i}{\partial \lambda_i^i} \right)^\top - k_4 \sum_{j \neq i} \left[\left(\frac{\partial \Psi_i}{\partial \lambda^{ij}} \right) \frac{\dot{\lambda}^{ij} \dot{\lambda}_i^{i\top}}{\|\dot{\lambda}_i^i\|^2} \right]^\top \\
& + k_3 \left(\frac{\partial \Phi_i}{\partial \lambda_i^i} \lambda_i^i \right) \lambda_i^i + k_3 \left(\sum_{j \neq i} \left(\frac{\partial \Phi_i}{\partial \lambda_j^i} \right) \frac{\dot{\lambda}_j^i \dot{\lambda}_i^{i\top}}{\|\dot{\lambda}_i^i\|^2} \right) \lambda_i^i \\
& + k_4 \left(\frac{\partial \Psi_i}{\partial \lambda_i^i} \lambda_i^i \right) \lambda_i^i + k_4 \left(\sum_{j \neq i} \left(\frac{\partial \Psi_i}{\partial \lambda^{ij}} \right) \frac{\dot{\lambda}^{ij} \dot{\lambda}_i^{i\top}}{\|\dot{\lambda}_i^i\|^2} \right) \lambda_i^i .
\end{aligned} \tag{3.25}$$

Then select a Lyapunov function candidate $L_{mm} = \sum_{i=1}^N \frac{1}{2} \dot{\lambda}_i^{i\top} \dot{\lambda}_i^i + k_3 \sum_{i=1}^N \Phi_i + k_4 \sum_{i=1}^N \Psi_i$. Its time derivative is

$$\begin{aligned}
\dot{L}_{mm} = & - \sum_{i=1}^N \left(\frac{k_1 \dot{\lambda}_i^{i\top} \dot{\lambda}_i^i}{r_i^i} + k_3 \frac{\partial \Phi_i}{\partial \lambda_i^i} \dot{\lambda}_i^i + k_3 \sum_{j \neq i} \frac{\partial \Phi_i}{\partial \lambda_j^i} \dot{\lambda}_j^i + k_4 \frac{\partial \Psi_i}{\partial \lambda_i^i} \dot{\lambda}_i^i + k_4 \sum_{j \neq i} \frac{\partial \Psi_i}{\partial \lambda^{ij}} \dot{\lambda}^{ij} \right) \\
& + k_3 \sum_{i=1}^N \left(\frac{\partial \Phi_i}{\partial \lambda_i^i} \dot{\lambda}_i^i + \sum_{j \neq i} \frac{\partial \Phi_i}{\partial \lambda_j^i} \dot{\lambda}_j^i \right) \\
& + k_4 \sum_{i=1}^N \left(\frac{\partial \Psi_i}{\partial \lambda_i^i} \dot{\lambda}_i^i + \sum_{j \neq i} \frac{\partial \Psi_i}{\partial \lambda^{ij}} \dot{\lambda}^{ij} \right) \\
= & - \sum_{i=1}^N \left(\frac{k_1 \dot{\lambda}_i^{i\top} \dot{\lambda}_i^i}{r_i^i} \right) \leq 0 ,
\end{aligned} \tag{3.26}$$

which is negative semidefinite, and zero only if $\dot{\lambda}_i^i = \mathbf{0}$. Consequently, each $\dot{\lambda}_i^i$ also asymptotically converges to zero. It follows that the interception can be guaranteed with $\dot{\mathbf{q}}_i^{i\top} \lambda_i^i \leq -\epsilon < 0$ and $\dot{\lambda}_i^i \rightarrow \mathbf{0}$, which proves (1).

Collision and mis-interception avoidance is proven by contradiction. The potential function and consequently L_{mm} diverge to infinity if the relative LOS approaches δ_1 or δ_2 . From (3.20) and (3.22), the potential functions satisfy

$$\lim_{\|\lambda_{ij}^i\| \rightarrow \delta_1^+} \phi(\lambda_{ij}^i) = +\infty, \quad \lim_{\|\lambda_i^{ij}\| \rightarrow \delta_2^+} \psi(\lambda_i^{ij}) = +\infty. \quad (3.27)$$

However, the derivative of L_{mm} is negative semidefinite, which contradicts the infinite L_{mm} . Hence, no collisions or mis-interceptions will occur and (2) is proven. $\square$

Remark 3.6. As can be seen from the proof of Theorem 3.1, since the interception strategy is based on direction (LOS), the direction-based CA is applied herein to cancel positive terms in derivatives of Lyapunov functions, such that interceptions and CA can be guaranteed.

Remark 3.7. Interceptors try to avoid each other whenever their lines of sight with respect to one target approaches each other, even if distances among them might still be large, indicating that the CA would be performed in advance before interceptors get very close to each other.

3.4.3 Simulation

Five interceptor UAVs are tasked with intercepting five targets. Each UAV is assigned a unique target. Targets and interceptor UAVs are modeled by double integrators like in (3.1). The initial positions and velocities are listed in Table 3.2. All targets have zero acceleration in this simulation. The interceptors move using distributed guidance law (3.23) with control gains $k_1 = 40$, $k_2 = 1$ and $k_3 = k_4 = 0.00002$. The reference relative speed is $v_r = -5\text{m/s}$. The parameters for potential functions (3.20) and (3.22) are $\Delta_1 = 0.08$, $\delta_1 = 0.0004$, $\Delta_2 = 0.1$ and $\delta_2 = 0.0004$.

The simulation result is shown in Figure 3.6. Interception is shown with no collisions or mis-interceptions. All interceptors complete interception within finite time and non-zero hitting velocities.

TABLE 3.2: Initial Positions and Velocities of Interceptors and Targets

$\mathbf{q}_T^1(0)$	$[-10, 6, 37]^T \text{m}$	$\mathbf{v}_T^1(0)$	$[2.4, 3.5, 0.1]^T \text{m/s}$
$\mathbf{q}_T^2(0)$	$[-14, 18, 28]^T \text{m}$	$\mathbf{v}_T^2(0)$	$[3.3, 2.5, 0.1]^T \text{m/s}$
$\mathbf{q}_T^3(0)$	$[8, 24, 32]^T \text{m}$	$\mathbf{v}_T^3(0)$	$[2.5, 2.5, 0]^T \text{m/s}$
$\mathbf{q}_T^4(0)$	$[9, 22, 39]^T \text{m}$	$\mathbf{v}_T^4(0)$	$[2.1, 4.2, -0.1]^T \text{m/s}$
$\mathbf{q}_T^5(0)$	$[13, -1, 48]^T \text{m}$	$\mathbf{v}_T^5(0)$	$[1.5, 4.4, -0.1]^T \text{m/s}$
$\mathbf{q}_{I1}(0)$	$[0, 0, 0]^T \text{m}$	$\mathbf{v}_{I1}(0)$	$[0, 0, 0]^T \text{m/s}$
$\mathbf{q}_{I2}(0)$	$[2, -2, 0]^T \text{m}$	$\mathbf{v}_{I2}(0)$	$[0, 0, 0]^T \text{m/s}$
$\mathbf{q}_{I3}(0)$	$[4, -4, 0]^T \text{m}$	$\mathbf{v}_{I3}(0)$	$[0, 0, 0]^T \text{m/s}$
$\mathbf{q}_{I4}(0)$	$[6, -6, 0]^T \text{m}$	$\mathbf{v}_{I4}(0)$	$[0, 0, 0]^T \text{m/s}$
$\mathbf{q}_{I5}(0)$	$[8, -8, 0]^T \text{m}$	$\mathbf{v}_{I5}(0)$	$[0, 0, 0]^T \text{m/s}$

FIGURE 3.6: Multi-UAV interception simulation

3.4.4 Semi-physical Experiment

To demonstrate the distributed multi-UAV interception guidance law, a semi-physical setup was used. The same pair of interceptor and target was used as before under the same conditions.

An additional pair of interceptor and target was simulated online on the Linux mini-computer of the interceptor UAV. The second target moves with constant velocity. The second interceptor was simulated with a point mass model with drag.

The starting and interception points of both real and physical pairs are shown in Figure 3.7. The first target now starts at 2m west and 20m north of the first interceptor. Interceptor 2 starts roughly 1m west of the real interceptor 1.

FIGURE 3.7: Trajectories of all interceptors and targets from the semi-physical experiment

Figure 3.7 shows a successful interception for both pairs. For interceptor 1, it is observed that it undergoes a wavy motion due to the proximity of targets 1 and 2 in the early stages where the λ_1^{12} is small. For interceptor 2, its response can be broken down into stages. In stage 1-2, interceptor 2 moves to avoid interceptor 1 due to small λ_{21}^2 . However, in stage 2-3 this resulted in a small λ_2^{21} , which caused interceptor 2 to move back where interceptor 1 is no longer nearby. It then completes its interception in stage 3-4 where there are no imminent collisions.

3.4.5 Issues with Experimental Implementation

Although interception was completed for both pairs with no apparent problems, further investigation into logged acceleration and control input data suggests otherwise. Firstly, referring to Figure 3.8, a lag can be observed in the actual acceleration behind the controller's output. This is likely due to the rapidly changing acceleration command due to the collision and mis-interception avoidance components of the controller. The rapid changes mean that the perfect inner loop control assumption may not hold under extreme conditions where the relative LOS is close

to δ_1 or δ_2 . Secondly, a rapid rocking motion is also observed in the X and Y axes at 1s to 3s and 2s to 4s respectively. Note that attitude is mapped to the control input up to 35° or a maximum of $6.867ms^{-2}$. Large attitude swings may not be desirable for certain payloads such as nets. Thirdly, actuator saturation is obvious at the beginning and the end where the control inputs are significantly larger than the actual acceleration. Hence, in the physical scenario, collision and mis-interception avoidance is not guaranteed as the non-linear quadrotor dynamics were not accounted for sufficiently. The use of an infinite potential function in (3.20) and (3.22) resulted in large and rapid changes in input, which the UAV cannot produce. As such, the safer scenarios such as in Figure 3.7 served to validate only the principles behind the controller design.

FIGURE 3.8: Logged and post-calculated acceleration data for physical UAV in semi-physical experiment. Calculated drag refers to drag calculated based on logged GPS velocity and not true wind velocity.

The heavy communication requirement also posed a problem in experiment realization, which led to the semi-physical version. Due to the lack of a radar system, XBee communication modules were required to transmit the required data. The data was required to be transmitted at a relatively high 50Hz rate to approximate the continuous stream of data and each interceptor requires knowledge of all other UAVs. Realistically, this would require at least 3 Xbee modules on each interceptor for dedicated channels in a 2 interceptor-target pair system. Experimentally, it is discovered that 50Hz is near the limit on a dedicated channel for the message size used.

3.5 Summary

In this chapter, a novel CA method using repulsion on the directional relative LOS for the multi-intruder interception scenario was presented. The guidance law for single interception was first formulated according to PN principles and velocity feedback, and subsequently validated by simulation and experiments. This law was then modified using the potential field method from [28] but applied to direction using the LOS vectors. The method guaranteed that the LOS of the interceptor towards its target remains clear. Additionally, this guarantees that no inter-interceptor collisions or mis-interceptions can occur. A simulation and semi-physical experiment was presented to demonstrate its effectiveness.

However, implementation was not without issues. The perfect inner loop control assumption does not hold in scenarios with highly dynamic motion. Real dynamic constraints on velocity, acceleration and jerk limited the experiment success. The concerns of dynamic constraints on potential field design highlighted in the literature review proved to be significant in this scenario.

Communication constraints also limited the multi-UAV experiment to the semi-physical one. This algorithm may potentially be implemented where radar is used to determine all the positions and is broadcasted to every interceptor. However, for other decentralized scenarios, it is clear that high frequency communication rates, such as that used here, is impractical for larger numbers of agents. If communication is to be used as a means of determining other agent's positions, then a CA algorithm that is robust to lower rates or dropouts should be designed.

Chapter 4

VO-based Reciprocal CA with Triggered Communication

This chapter proposes a modified version of the VO-based collision avoidance (CA) method to tackle the issue of limited bandwidth in communication by using an event-triggered protocol. The distributed VO CA presented here is designed to ensure non-collision regardless of the varying lengths of time intervals between instances of communication updates between agents in a swarm. Without further communication, non-collision is guaranteed but the agent may not be able to reach its designated goal. The event-triggered protocol is used to determine when an agent should communicate with another so that the respective VO-based constraint may be updated allowing it to move to its designated goal. The content presented here was published in [\[100\]](#).

The purpose of this chapter is to present a novel and alternative method in which CA can be implemented for a swarm with communication constraints considered. Firstly, CA is formulated in velocity space by using VOs instead of repulsion like in Chapter 3. Secondly, as VO is partition-based, collision constraints can be constructed such that they can be maintained indefinitely without collision until the next communication. This results in a much reduced communication rate. This

is in contrast to the repulsion-based method [39], which still requires an upper bound on time intervals.

This chapter is organized as follows. The following section provides a further justification for considering the limited communication problem and motivation for the choice of CA method. Section 4.2 then defines the model, communication system, and the problem to be solved. Section 4.3 describes the VO-based reciprocal CA method and the modifications required. The event-triggered communication protocol is presented in Section 4.4. Effectiveness of the method is demonstrated with simulation in Section 4.5. A summary is provided in Section 4.6.

4.1 Motivation

Control algorithms for multi-agent systems are often designed to run periodically at relatively high frequencies, which is problematic in application if they require inter-agent communication. Thus it is prudent to design algorithms with communication protocols that consider the bandwidth limitation. The research interest is evident in the form of event- or self-triggered related work [101]. In event-triggered algorithms, the main idea is that control inputs should be recalculated only when performance is not satisfactory. This usually refers to comparing the current Lyapunov function value to a fraction of its initial. For self-triggering, continuous performance monitoring is circumvented by predicting the next update time. The latter was adapted to CA in [37, 39].

In [37, 39], Rodríguez-Seda integrated a self-triggered communication scheme with cooperative potential functions. The time to collision based on the predefined maximum velocity is used. In between updates, the repulsive force is calculated from the last received position of the other agent. Although the method was shown to be effective, the potential function is not trivial to adapt to systems with constrained states. [37] only considered the two agent case while [39] solved the multi-agent case using the infinite potential function similar to (2.9). Moreover, it

is vulnerable to communication failure as an upper bound on update intervals is required.

In contrast, the VO method can be formulated to guarantee non-collision for all time after a single update as long as the relative velocity remains outside of the VO and is not head-on. However, there also exist spatial configurations of agents where the set of safe velocities is empty. If this happens, non-collision is not guaranteed. Hence, modifications are required to ensure that the safe set remains non-empty at each update.

In this work, a communication protocol utilizing a target-relative measure-based trigger with a modified VO avoidance algorithm is presented. The protocol serves to determine, similar to event-triggered control, times at which an agent should communicate with another based on its motion relative to the target position. This is to enable agents to reach their assigned targets. For the limitation mentioned earlier, a sufficient condition is established to ensure that the quadratic program has a feasible solution at every update. A modification to the standard VO method is also provided to allow agents to approach zero velocity which is necessary for stopping at their target.

4.2 Problem Formulation

4.2.1 System

The 2D single integrator model is assumed for a group of N agents. This work explores for the first time the use of trigger-based communication with the VO method. Hence, higher order models are left to future extensions.

$$\dot{\mathbf{q}}_i = \mathbf{v}_i, \quad \mathbf{q}_i \in \mathbb{R}^2, \quad i \in \mathcal{N} = \{1, 2, \dots, N\}, \quad (4.1)$$

where $\|\mathbf{v}_i\| \leq v_{\max}$, and $v_{\max}$ is the maximum velocity constraint. The control input, $\mathbf{v}_i$, is updated discretely with control frequency, f .

Each agent is assumed to know or able to measure only its position and velocity. The positions and velocities of other agents are only obtainable through communication.

Additionally, a constraint on minimum separation distance, $d_{\min}$, is set between every pair of agents. Each agent is considered as a point object in the 2D space. Hence, collision between any pair of agents is defined as

$$\|\mathbf{q}_{ij}\| = \|\mathbf{q}_i - \mathbf{q}_j\| < d_{\min}, \quad i, j \in \mathcal{N}, \quad i \neq j. \quad (4.2)$$

4.2.2 Communication

The communication system assumed in this work is described as follows. Previous VO-related works assume that each agent be able to measure the positions and velocities of every other agent at each time step. In this work, it is instead considered that agents are only able to obtain position and velocity by communication on a wireless ad hoc network [102].

Each instance of communication is restricted to pairwise only. At any control time step, agent i may request communication with any other agent, $j \neq i$. Assuming an instance of communication for agent j at the same time step does not already exist with another agent, the new communication is accepted which blocks any other requests. Likewise, an agent cannot request communication if it is already responding to another.

In this work, zero latency and no dropped messages are assumed. Each instance is a simultaneous exchange of only position and velocity information.

4.2.3 Objective

The aim of this work is to design a distributed inter-agent CA and communication algorithm for agents with dynamics (4.1) to avoid collision as defined in (4.2) under

the communication system described in Section 4.2.2. Additionally, each agent is required to reach its predefined target position, $\mathbf{q}_i^f$, within finite time.

The purpose of the algorithm is to reduce communication rates between agents without compromising ability to avoid collision and reach their target positions.

4.3 VO-based Reciprocal CA

In this section, reciprocal CA using VO is formulated based on the works first introduced in [57] and further developed in [19, 54]. Modifications are then presented, which are designed to meet the requirements of the defined problem.

4.3.1 Preliminaries

Consider two agents i and j with dynamics (4.1) moving at some constant relative velocity. The velocity obstacle $VO_{i|j}$ denotes the set of velocities of i relative to j that would result in collision as defined in (4.2) at some future time $t > t_0$. Similar to [54], the collision cone can be defined as,

$$VO_{i|j} = \{\mathbf{v} | \exists t \geq 0, \|\mathbf{v}t - \mathbf{q}_{ji}\| < d_{\min}\}, \quad (4.3)$$

where $\mathbf{q}_{ji}$ is the position of agent j relative to i . Figure 4.1 illustrates the velocity obstacle in relative position and velocity space.

Note that this formulation forgoes the usual truncation of the collision cone using a time horizon as described in [54] and reviewed in Figure 2.2 due to uneven time intervals between VO updates when considering a trigger-based communication system.

From the VO, respective half-planes of permissible velocities are derived as linear constraints for the optimization of $\mathbf{v}_i$ and $\mathbf{v}_j$ to avoid collision. A vector $\mathbf{u}_{i|j}$, shown in Figure 4.1, can be obtained from the closest point on either straight-line

FIGURE 4.1: $VO_{i|j}$ and the respective $\beta_{i|j}$ and $\beta_{j|i}$ of agents i and j in relative velocity and position space superimposed. $VO_{i|j}$ is represented by the shaded triangular region not inclusive of the straight line clockwise and counterclockwise boundaries with apex at origin. $\hat{\mathbf{n}}_{i|j}^{cw}$ is the unit outward normal to the clockwise boundary. The partially shaded side of the $\beta_{i|j}$ and $\beta_{j|i}$ boundaries indicate the respective half-plane of allowable velocities.

boundary of the VO to the relative velocity $\mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j$. This vector represents either the minimal change required to move $\mathbf{v}_{ij}$ out of the VO, or the maximal permissible change in the direction of the boundary if it was already outside of the VO. The change $\mathbf{u}_{i|j}$ is shared between i and j according to

$$\mathbf{u}_{i|j} = \mathbf{u}_{i|j}^i - \mathbf{u}_{i|j}^j, \quad (4.4)$$

$$\mathbf{u}_{i|j}^i = \rho \mathbf{u}_{i|j}, \quad \rho \in [0, 1]. \quad (4.5)$$

To obtain the half-plane constraints, either of the two straight line boundaries can be chosen if $\mathbf{v}_{ij} \in VO_{i|j}$, but the nearest should be chosen if $\mathbf{v}_{ij} \notin VO_{i|j}$ and $\mathbf{v}_{ij}^\top \mathbf{p}_{ji} > 0$. If $\mathbf{v}_{ij}^\top \mathbf{p}_{ji} \leq 0$, there exists a region of overlap for both half-planes. The definition of Optimal Reciprocal Collision Avoidance (ORCA) in [54] dictates that the half-plane be from the nearest point on the VO boundary.

In this work, for swarm coordination, the clockwise boundary is chosen initially for all agents and a method for coordinated switching is described in the next section.

The resulting permissible set of velocities for i induced by j is defined as

$$\beta_{i|j} = \{ \mathbf{v} \mid (\mathbf{v} - (\mathbf{v}_i + \mathbf{u}_{i|j}^i))^T \hat{\mathbf{n}}_{i|j} \geq 0 \}, \quad (4.6)$$

where $\hat{\mathbf{n}}_{i|j}$ is the outward normal to the chosen boundary of $VO_{i|j}$, i.e.

$$\hat{\mathbf{n}}_{i|j} = \hat{\mathbf{n}}_{i|j}^{\text{cw}} \quad \text{or} \quad \hat{\mathbf{n}}_{i|j}^{\text{ccw}}, \quad (4.7)$$

where superscripts cw and ccw respectively denote clockwise and counterclockwise boundaries. This is also illustrated in Figure 4.1.

Since the VO and $\beta_{i|j}$ are defined in relative velocity space, and that the $VO_{j|i}$ is a mirror of $VO_{i|j}$, agent j can obtain similarly its $\beta_{j|i}$ with the same information as i and an appropriate ρ to satisfy (4.4). Note that $\hat{\mathbf{n}}_{j|i} = -\hat{\mathbf{n}}_{i|j}$. Thus selecting velocities within their respective $\beta_{i|j}$ ensures that $\mathbf{v}_{ij}(t) \notin VO_{i|j}(t)$ and no collision can happen at any future time. Note that it is necessary that both agents select the same boundary.

The following proposition describes how (4.6) maintains non-collision even if it is not updated at every time step.

Proposition 4.1. *Consider two agents i and j of dynamics (4.1) at time $t = t_0$ and moving such that $\mathbf{v}_i(t_0) \in \beta_{i|j}(t_0)$, $\mathbf{v}_j(t_0) \in \beta_{j|i}(t_0)$ and $\|\mathbf{q}_i - \mathbf{q}_j\| \geq d_{\min}$. Then if $\mathbf{v}_{ij}(t)$ is subsequently changed such that $\mathbf{v}_i(t) \in \beta_{i|j}(t_0)$ and $\mathbf{v}_j(t) \in \beta_{j|i}(t_0)$ for any $t > t_0$, no collision, (4.2), can happen for all $t > t_0$.*

Proof. Since $\mathbf{v}_i(t) \in \beta_{i|j}(t_0)$ and $\mathbf{v}_j(t) \in \beta_{j|i}(t_0)$, from (4.6),

$$(\mathbf{v}_i(t) - (\mathbf{v}_i(t_0) + \mathbf{u}_{i|j}^i(t_0)))^T \hat{\mathbf{n}}_{i|j}(t_0) \geq 0, \quad (4.8)$$

$$(\mathbf{v}_j(t) - (\mathbf{v}_j(t_0) + \mathbf{u}_{j|i}^j(t_0)))^T \hat{\mathbf{n}}_{j|i}(t_0) \geq 0. \quad (4.9)$$

Add (4.8) and (4.9), noting that $\hat{\mathbf{n}}_{i|j}(t_0) = -\hat{\mathbf{n}}_{j|i}(t_0)$, to obtain

$$\begin{aligned} (\mathbf{v}_{ij}(t) - (\mathbf{v}_{ij}(t_0) + \mathbf{u}_{i|j}(t_0)))^T \hat{\mathbf{n}}_{i|j}(t_0) &\geq 0 \\ \mathbf{v}_{ij}(t)^T \hat{\mathbf{n}}_{i|j}(t_0) &\geq 0. \end{aligned} \quad (4.10)$$

□

Remark 4.1. Proposition 4.1 simply states that if the constraints $\beta_{i|j}$ and $\beta_{j|i}$ remain unchanged, any change in $\mathbf{v}_i$ or $\mathbf{v}_j$ will not result in collision as long as the respective constraints are satisfied.

The preferred velocity to drive agent i to its goal, $\mathbf{q}_i^f$, is set as

$$\mathbf{v}_i^{\text{pref}} = \min \left[k_p \left\| \mathbf{q}_i^f - \mathbf{q}_i \right\|, v_{\max} \right] \frac{\mathbf{q}_i^f - \mathbf{q}_i}{\left\| \mathbf{q}_i^f - \mathbf{q}_i \right\|}, \quad (4.11)$$

where $k_p > 0$ is the control gain.

Typically, for each agent, the velocity that ensures non-collision is chosen by convex optimization using quadratic programming. The following convex problem is solved at each time step.

Optimization Problem 4.1.

$$\min \left\| \mathbf{v}_i - \mathbf{v}_i^{\text{pref}} \right\|, \quad (4.12)$$

subject to

$$\mathbf{v}_i \in \beta_i = \bigcap_{j \in \mathcal{N} \setminus \{i\}} \beta_{i|j}. \quad (4.13)$$

The expected running time is linear with respect to the number of constraints [54].

4.3.2 Feasibility Maintenance of the Quadratic Program

Agents are required to reach their target positions, while guaranteeing non-collision. From Proposition 4.1, it is already established that no collision can happen between i and j with the appropriate $\mathbf{v}_{ij}$. However, it is possible that $\mathbf{v}_i^{\text{pref}} \notin \beta_i$. Hence, i may either be unable to move in the preferred direction or stop at $\mathbf{q}_i^f$. This is where the communication system is required for updating of pairwise constraints, and is described in Section 4.4. However, earlier it is noted that there exist configurations

of agents and velocities that make the optimization infeasible. It is thus necessary with each update of constraint to ensure that, at the very least, the current velocity $\mathbf{v}_i$ remains in β_i , keeping the program feasible.

Assumption 4.1. Each agent i has initial knowledge of $\mathbf{q}_j$ and $\mathbf{v}_j$, $j \in \mathcal{N} \setminus \{i\}$ at time $t = 0$.

Assumption 4.2. The initial configuration, $\mathbf{q}_i(t = 0)$ and $\mathbf{v}_i(t = 0)$, $\forall i \in \mathcal{N}$ is such that a feasible solution, $\|\mathbf{v}_i\| \leq v_{max}$, for OP 4.1 exists for every agent, and at least one has a non-zero solution. For the latter, this ensures that the initial solution is not a deadlock scenario.

With each instance of communication, the agent will update its knowledge of the other's position and velocity, thus updating the relevant constraint (4.6). With each update, the solution of OP 4.1 must be feasible. Otherwise, CA cannot be guaranteed even if the velocity is set to $\mathbf{v}_i = 0$ as this may result in $\mathbf{v}_{ij} \in VO_{ij}$ for some j .

Theorem 4.1. Under Assumption 4.2, if a communication occurs between $i \in \mathcal{N}$ and $j \in \mathcal{N} \setminus \{i\}$ at time $t_{ij}^k > 0$ and a new $VO_{ij}(t_{ij}^k)$ is obtained according to (4.3), then $\mathbf{v}_{ij}(t_{ij}^{k-1}) \notin VO_{ij}(t_{ij}^k)$, where $\mathbf{v}_{ij}(t_{ij}^{k-1})$ is the old relative velocity before solving OP 4.1 at $t = t_{ij}^k$. Superscripts k and $k - 1$ denote current and previous control time steps respectively. This applies to each subsequent update.

Proof. With Assumption 4.2, and Proposition 4.1, it is established that $\mathbf{v}_{ij}(t_{ij}^{k-1}) \notin VO_{ij}(t = 0)$. Thus $\|\mathbf{q}_{ij}(t_{ij}^k)\| > d_{min}$. From (4.10), $\mathbf{v}_{ij}(t_{ij}^{k-1})^\top \hat{\mathbf{n}}_{ij}(0) \geq 0$. Since $\mathbf{q}_{ji}(0)^\top \hat{\mathbf{n}}_{ij}(0) < 0$, then $\mathbf{q}_{ji}(t_{ij}^k)^\top \hat{\mathbf{n}}_{ij}(0) < 0$. Hence, upon obtaining $VO_{ij}(t_{ij}^k)$, it is ensured that $\nexists t > t_{ij}^k$ where $\|\mathbf{v}_{ij}(t_{ij}^{k-1}) \times (t - t_{ij}^k) - \mathbf{q}_{ji}(t_{ij}^k)\| < d_{min}$, which satisfies the complement of the new $VO_{ij}(t_{ij}^k)$ according to its definition (4.3). The same reasoning is applicable to each subsequent update. $\square$

Theorem 4.1 does not guarantee that the β_{ij} , derived from either the clockwise or counterclockwise boundaries, maintains feasibility. However, based on the VO_{ij}

definition (see Figure 4.1), it is guaranteed that at least one of the boundaries maintains feasibility if chosen. The required boundary selection method is formalized in Algorithm 4.1.

Algorithm 4.1 Boundary selection for agent i relative to j

```

1: Input
2:   boundSelFlagij ∈ {1, 2}, vij, n̂i|jcw, n̂i|jccw
3: Output
4:   boundSelFlagij, n̂i|j
5: if vij · n̂i|jCW ≥ 0 then
6:   if vij · n̂i|jCCW ≥ 0 then
7:     if ||vij|| <  $v_{ij}^{threshold}$  then
8:       boundSelFlagij ← (boundSelFlagij mod 2) + 1
9:     end if
10:  else
11:    boundSelFlagij ← 1
12:  end if
13: else
14:   boundSelFlagij ← 2
15: end if
16: if boundSelFlagij = 1 then
17:   n̂i|j ← n̂i|jCW
18: else
19:   n̂i|j ← n̂i|jCCW
20: end if

```

Simply put, Algorithm 4.1 determines which of the two possible boundaries for $\beta_{i|j}(t)$ is such that the current $\mathbf{v}_i(t) \in \beta_{i|j}(t)$. The variable "boundSelFlag_{ij}" is globally initialized at 1, i.e. every agent starts with the clockwise boundary. $v_{ij}^{threshold} \in (0, v_{max}]$ is a global parameter. Note that similar to $\beta_{i|j}$ definition, the algorithm for j relative to i is mirrored. Hence, both agents, given the same information, will select the same boundary without additional signaling.

4.3.3 Constraint Relief

Constraint relief refers to the choosing of constraints upon a communication update that provides a solution that is closer to $\mathbf{v}_{pref}$ or increases the solution space. In prior works [19, 54], the boundary nearest from $\mathbf{v}_{ij}$ was chosen. For [19], the

situation where $\mathbf{v}_{ij}^\top \mathbf{q}_{ji} \leq 0$ and the two possible half-planes overlap is not considered. In [54], the cone is truncated, which increases the solution space around the origin by allowing head-on velocities but expires after some time τ . Note that head-on velocities are not considered in this work for simplification in communication triggering. In both cases, the VOs are updated at each time step, which likely mitigate the need for careful selection. Any switching from one side of the VO to the other in the latter case, is also smoother due to the truncation. However, in this work, the updates are relatively far apart and only for one other agent at a time. Hence, two techniques are provided here to explicitly manipulate the calculation of constraint $\beta_{i|j}$ to improve the solution. One involves switching between clockwise and counterclockwise constraints and is included in Algorithm 4.1. The other provides a simple rule for *effort sharing* in (4.4).

Switching: The $v_{ij}^{threshold}$ parameter in Algorithm 4.1 is set as some small fraction of v_{max} . The rationale is that as the agents continue to move and solve their respective programs, a situation may be encountered where one agent, i , slows down as the $\mathbf{v}_i^{pref}$ becomes perpendicular to the boundary of β_i . This implies that a particular $\beta_{i|j}$ is in need of update. If a $i \rightarrow j$ communication does occur, then using Algorithm 4.1, with $\|\mathbf{v}_{ij}\| < v_{ij}^{threshold}$, a switch will occur if $\mathbf{v}_{ij}$ is in the overlap region of the new constraints.

Note that although it is possible to instead use $\|\mathbf{v}_i\| < v_{ij}^{threshold}$, $\mathbf{v}_{ij}$ gives a better indication of the situation. If $\|\mathbf{v}_i\|$ is small, $\|\mathbf{v}_j\|$ may not be. Thus, a constraint switch may not be necessary as the new $VO_{i|j}$ can be significantly different from before. A small $\|\mathbf{v}_{ij}\|$ instead suggests that the new $VO_{i|j}$ may not differ much, hence a switch may be useful.

Effort Sharing: From (4.4) and (4.5), it is established that the available solution space can be shared between agent i and j . In previous works such as [54, 67], $\rho = 0.5$ is suggested. However, these works have agents with more frequent updates and use truncated VO with time horizons.

The *effort sharing* is formulated simply as

$$\mathbf{u}_{i|j}^i = \frac{\|\mathbf{v}_i\|}{\|\mathbf{v}_i\| + \|\mathbf{v}_j\|} \mathbf{u}_{i|j}. \quad (4.14)$$

As with the $\beta_{i|j}$, (4.14) is calculated at the update of the pairwise constraint, and held constant until the next update.

In Figure 4.1, it can be observed that for agent j , the origin is within its $\beta_{j|i}$, but not for i with $\beta_{i|j}$. Should $\mathbf{v}_j$ become zero (e.g. $\|\mathbf{q}_j^f - \mathbf{q}_j\| = 0$) at some future time, if $\rho = 0.5$ is used, the origin velocity would not be available to agent i . Without sharing the target positions, agent i does not know whether the other agent has reached its target position or is in a temporary deadlock and may start moving again after. In such a situation, simply setting the $\mathbf{v}_i = \mathbf{0}$ at $\|\mathbf{q}_i^f - \mathbf{q}_i\| = 0$ violates the guarantee of non-collision if $\mathbf{0} \notin \beta_i$.

Instead, (4.14) is used to delegate the effort to ensure that the solution space is not overly conservative. In the above scenario, i will have the full solution space as $\|\mathbf{v}_j\| = 0$, thus allowing its velocity to approach the origin. (4.14) is thus a necessary condition for $\|\mathbf{q}_i^f - \mathbf{q}_i\| \rightarrow 0$.

The combined effect of these two methods is shown by simulation of one agent encountering deadlock with two stationary agents in Section 4.5.

4.3.4 Maximum Velocity Constraint

A quadratic constraint on velocity can be avoided using the following method. $\mathbf{v}_i^{pref}$ is scaled by the factor

$$\alpha_i = \min_{j \in \mathcal{N} \setminus \{i\}} \min(\alpha_{i|j}, 1), \quad (4.15)$$

where

$$\alpha_{i|j} = \begin{cases} 1 & \text{if } \mathbf{v}_i^{pref} \in \beta_{i|j} \text{ or } \mathbf{v}_i^{pref} \parallel \hat{\mathbf{n}}_{i|j}, \\ \Gamma & \text{otherwise.} \end{cases} \quad (4.16)$$

$\Gamma > 0$ is such that

$$\mathbf{v}_{\beta_{i|j}}^{\max} = \Gamma \mathbf{v}_i^{\text{pref}} + K \hat{\mathbf{n}}_{i|j}, \quad (4.17)$$

where

$$\begin{aligned} & \mathbf{v}_{\beta_{i|j}}^{\max} \\ &= \arg \min_{\mathbf{v}} \left\{ \left\| \mathbf{v} - \mathbf{v}_i^{\text{pref}} \right\| \left| \begin{array}{l} \|\mathbf{v}\| = v_{\max}, \\ (\mathbf{v} - (\mathbf{v}_i + \mathbf{u}_{i|j}^i)) \cdot \hat{\mathbf{n}}_{i|j} = 0 \end{array} \right. \right\}, \end{aligned} \quad (4.18)$$

and $K > 0$ is a scaling factor.

The updated convex optimization is

Optimization Problem 4.2.

$$\min \left\| \mathbf{v}_i - \alpha_i \mathbf{v}_i^{\text{pref}} \right\|, \quad (4.19)$$

subject to

$$\mathbf{v}_i \in \beta_i = \bigcap_{j \in \mathcal{N} \setminus \{i\}} \beta_{i|j}. \quad (4.20)$$

Theorem 4.2. Consider agent i with dynamics (4.1) and OP 4.2. Under Assumption 4.2, using Algorithm 4.1 and (4.14), then $\|\mathbf{v}_i\| < v_{\max} \forall t \geq 0$.

Proof. Assumption 4.2 ensures that $\mathbf{v}_i$ satisfies initial β_i , and $\|\mathbf{v}_i\| \leq v_{\max}$.

Using quadratic programming, the solution is the nearest point to $\alpha_i \mathbf{v}_i^{\text{pref}}$ on either an edge or a vertex of β_i , or $\mathbf{v}_i^{\text{pref}}$ if unconstrained.

For the edge scenario, by using α_i in (4.19), it is ensured that the nearest point on any edge is at most of magnitude $v_{\max}$.

For the vertex scenario, the vertex may not be the nearest point from $\alpha_i \mathbf{v}_i^{\text{pref}}$ to the $\beta_{i|j}$ of either adjacent edge. Such points are thus $\notin \beta_i$. However, initial $\|\mathbf{v}_i\| \leq v_{\max}$. Theorem 4.1 enacted with Algorithm 4.1 maintains current velocity $\mathbf{v}_i \in \beta_i \forall t \geq 0$ with each update of a constraint. Hence, any $\mathbf{v}_i$ that is constrained at the vertex, cannot be greater than $v_{\max}$ in magnitude. The situation arises only

because $\mathbf{v}_i^{\text{pref}}$ has reduced in magnitude or changed to point into the vertex, and not due to a new $\beta_{i|j}$ "pushing" $\mathbf{v}_i$ to a larger magnitude.

Additionally, the same reasoning is applicable to case 2 of (4.16). $\square$

4.4 Communication Protocol Using Measure-based Triggers

In this section a mechanism for an agent to determine when to request communication and to which other agent is described.

At each time step, each agent determines first if the trigger happens. If it does, a communication is then requested with the appropriate target. If the communication is not blocked, position and velocity information are exchanged. This is followed by the calculation of the new $VO_{i|j}$ and $\beta_{i|j}$, where j is the agent responding to the communication. Subsequently, OP 4.2 is solved to obtain the command velocity.

4.4.1 Trigger Mechanism

The trigger mechanism is a product of three non-dimensional target-relative measures representing angle discrepancy, distance travelled, and velocity, which is checked with a predefined threshold. Each measure consists generally of a current variable and a reference variable, and is independent of other agents' information. The reference variable in this mechanism is either the agents' preferred velocity or its target position. The measures and the product are defined first. The rationale is explained after.

Angle Discrepancy: Intuitively, this measure represents the discrepancy between the current velocity and preferred velocity. Specifically,

$$\Delta_i^{\text{angle}} = \begin{cases} 0 & \text{if } \|\mathbf{v}_i^{\text{pref}}\| = 0, \\ 1 & \text{else if } \|\mathbf{v}_i\| = 0, \\ \frac{\arccos\left(\frac{\mathbf{v}_i^{\text{pref}} \cdot \mathbf{v}_i}{\|\mathbf{v}_i^{\text{pref}}\| \|\mathbf{v}_i\|}\right)}{0.5\pi} & \text{otherwise.} \end{cases} \quad (4.21)$$

Thus $\Delta_i^{\text{angle}} \in [0, 2]$. For the second case, the angle is assumed to be $\pi/2$ because the only scenario where $\|\mathbf{v}_i\| = 0$ other than at the target, is when $\mathbf{v}_i^{\text{pref}}$ is perpendicular to a boundary of β_i . The divisor 0.5π is chosen in the third case to match the second case smoothly. Hence, Δ_i^{angle} is defined such that moving in a direction other than $\mathbf{v}_i^{\text{pref}}$ increases it.

Distance Travelled: This measure is intended as a damper to prevent triggering at every time step if Δ_i^{angle} is high.

$$\Delta_i^{\text{dist}} = \max\left(\frac{\|\mathbf{q}_i(t) - \mathbf{q}_i(t_{\text{last}})\|}{\|\mathbf{q}_i^f - \mathbf{q}_i(t_{\text{last}})\|}, \varepsilon\right), \quad (4.22)$$

for some small $\varepsilon > 0$. Thus $\Delta_i^{\text{dist}} \in [\varepsilon, 1]$. $\varepsilon > 0$ is necessary to ensure triggering can still happen at near zero velocities. t_{last} is the time of last trigger. It is also updated if the *most offending* constraint is updated during a communication triggered by another agent. The *most offending* constraint is the $\beta_{i|j}$ for some j that will result in the largest change to the solution for OP 4.2 if modified. Identification of this constraint is discussed in the next section.

Velocity: This measure scales the magnitude of the product inversely to the velocity.

$$\Delta_i^{\text{vel}} = \frac{\|\mathbf{v}_i^{\text{pref}}\|}{\|\mathbf{v}_i\|}. \quad (4.23)$$

It works in tandem with Δ_i^{dist} . When $\|\mathbf{v}_i\|$ is extremely low relative to $\|\mathbf{v}_i^{\text{pref}}\|$, it implies that i is moving too slowly and not due to proximity with target position.

Hence, it negates the slow changing Δ_i^{dist} .

The trigger condition:

$$\Delta_i = \Delta_i^{\text{angle}} \Delta_i^{\text{dist}} \Delta_i^{\text{vel}} > \Delta_i^{\text{threshold}}, \quad (4.24)$$

where $\Delta_i^{\text{threshold}} > 0$ is a parameter that can be set globally or individually. From simulations, a good range is $[0.3, 0.6]$. Its effectiveness is shown in Section 4.5.

The primary measure for determining the motion performance of the agent relative to its target is the angle discrepancy (4.21). The further the relative direction, the more critical it becomes to update the *offending* constraint. However, if used by itself, it would cause continuous triggering when its value is high. This can happen if the updates in constraints do not allow the optimization solution to point closer to the target.

The next measure (4.22) is used to prevent continuous triggering. This requires that the agent move some distance before allowing another trigger. It is scaled using the target position so that $\Delta_i^{\text{dist}} \in [\varepsilon, 1]$. Δ_i^{dist} thus serves to dampen (4.21) to some fraction of its value.

Alternatively, a time threshold may be considered as a separate trigger before checking (4.21). However, doing so puts a lower bound on trigger times. In the case where the agent moves slowly relatively to $\mathbf{v}_i^{\text{pref}}$, which is close to deadlock, more frequent updates would be useful to check if the deadlock can be broken. Note that deadlock here is the situation where some subset of agents are at near zero velocity and unable to move towards their target positions. To enable more frequent updates in the above scenario, the third measure (4.23) is used. Consequently, the lower bound on trigger times is the controller time interval.

4.4.2 Constraint Update Selection

Due to the restriction of communication, a method for selecting which agent to communicate with is required. Logically, agent i should communicate with the agent where the $\beta_{i|j}$ constraint is *most offending*. From quadratic programming, suitable candidates are the boundaries of constraints on which lies the point $\mathbf{v}_i$. There is at least one such constraint. Otherwise, from (4.21), a trigger would not have occurred. Of these constraints, selection is according to

$$\arg \min_{j \in \mathcal{N} \setminus \{i\}} \left(\mathbf{v}_i^{\text{pref}} - \mathbf{v}_i \right) \cdot \hat{\mathbf{n}}_{i|j}, \quad (4.25)$$

where $\hat{\mathbf{n}}_{i|j}$ is the unit outward normal of $\beta_{i|j}$.

However, (4.25) only provides the constraint that is most opposite to $\mathbf{v}_i^{\text{pref}} - \mathbf{v}_i$. Ideally, the appropriate constraint $\beta_{i|j}$ is the one $\mathbf{v}_i^{\text{pref}}$ is not in and represents an edge in β_i . The algorithm is not trivial and deferred to future work.

The current solution works well enough when the numbers of agents and the constraints are small (< 10). Moreover, the constraint chosen is one where the nearest point from $\mathbf{v}_i^{\text{pref}}$ is closest to the vertex. Updating it allows for the most change based on the current $\mathbf{v}_i^{\text{pref}}$.

4.5 Simulations

A simulation of five agents swapping positions is presented here to show the effectiveness of the communication protocol and reciprocal CA algorithm. The agents begin at points on a circle of radius 4 uniformly spaced apart with zero initial velocity. Their objective is to reach the starting position of the agent two places away clockwise on the circle using the reciprocal CA algorithm and communication protocol of Sections 4.3 and 4.4 respectively. Algorithm parameters as described earlier are presented in Table 4.1. The controller frequency is denoted by f .

TABLE 4.1: Simulation Parameters

v_{max}	1.50m/s	d_{min}	1.0m	f	20Hz
k_p	0.7	$v_{ij}^{threshold}$	0.30m/s	$\Delta_i^{threshold}$	0.5

The simulation is presented in Figure 4.2. The agents are denoted by differently colored circles and their target positions by crosses of the same color. Their trails are also shown in the same color. In Figure 4.2(c), sharp turns are observed with a straight segment after, which indicate moments where a constraint is updated allowing the solution to be closer to $\mathbf{v}_i^{pref}$. Segments that appear curved are due to low velocities and relatively frequent updates. If between updates, it is instead due to the change of $\mathbf{v}_i^{pref}$ as the agent moves in a direction other than $\mathbf{v}_i^{pref}$. In Figure 4.2(d), agent 4 is the last to reach its target having taken a longer path from overshooting. This is due to *effort sharing* in (4.14). The $\beta_{4|j}$ may have been obtained while other agents were still moving. Thus, for some $\beta_{4|j}$, it is possible for the origin to be outside of β_4 . It is only able to reach its target after identifying the *offending* constraints and updating them.

To show the combined effect of Algorithm 4.1, the *effort sharing* of (4.14), and the constraint selection of Section 4.4.2, a simulation of one agent with two other stationary agents is presented in Figure 4.3. The objective of agent 1 is to reach position (6, 3), while the other two agents remain stationary but still able to communicate. It is observed that as the path agent 1 takes becomes perpendicular to the relative position between the agent and the target, it switches its constraint from clockwise to counterclockwise relative to agent 3. At this point, its velocity is zero. This is repeated later for agent 2.

The communication log for agent 4 from the first simulation is presented in Figure 4.4. The vertical axis of Figure 4.4 denotes the agents with which agent 4 communicated with at their respective points in time. The circles denote points at which agent 4 triggered according to (4.24), and requests and completes a communication with the respective agent. Crosses denote the opposite where agent 4 instead responds to another agent's request. It is observed that the requested

FIGURE 4.2: Five agents on a circle swapping positions two places clockwise while avoiding collision and using the triggered communication protocol. Each agent is denoted by differently colored circles and their target positions by crosses of the same color. Their trails show the past 32s of position history. The minimum separation is denoted by the dotted circles around each agent.

communications are spaced far apart relative to the controller frequency. For the responses, there are instances where the gap with another communication is small because they are triggered by other agents. Nevertheless, the average communication rate is low. This is especially evident with agent 1 where there were only 5 instances.

The total counts of triggers and communications, both requested and responded, are in Table 4.2. Triggers refer to the number of times an agent triggered and

FIGURE 4.3: Agent 1 navigating to target position at (6,3) between two stationary agents. The entire trail of agent 1 is shown. The minimum separation is denoted by the dotted circles around each agent.

FIGURE 4.4: Communication log of agent 4. Circles denote trigger times and completed communication requests to other agents. Crosses denote times at which agent 4 responds to other agents' requests. Note that each agent encounters at most one communication, either request or response, at each time step.

requested communication for the entire simulation of 48s. Communications counts both those triggered by the agent and responded to other agents. The five agents were able to reach their targets within reasonable time even with such low rates of communication. Lowering $\Delta_4^{threshold}$ will of course increase these rates but reduce

FIGURE 4.5: Log of trigger Δ_4 . $\Delta_4^{velocity}$ is omitted due to magnitude.

time of completion.

TABLE 4.2: Trigger and Communication Counts

Agent ID	1	2	3	4	5
Triggers	8	16	12	22	10
Communications	22	23	28	33	25

Agent 4's trigger measure logs are presented in Figure 4.5. The black graph shows the value of the primary measure, Δ^{angle} , which remains above the threshold for most of the simulation except towards the end. The distance measure, Δ^{dist} in blue, is shown increasing from ε with each trigger, which corresponds to the increase in the overall Δ . Note that the trigger does not guarantee that the updated constraint allows for the angle discrepancy to become significantly reduced. However, reductions in Δ^{angle} can be seen for some triggers. Δ^{vel} is omitted as its magnitude at some points dwarfs the rest, but its effect can be seen clearly at $t = 40s$ where both Δ^{angle} and Δ^{dist} are below the threshold. This is when the agent is moving at low velocity. Since it is also low relative to its $\mathbf{v}^{pref}$, Δ^{vel} pulls the value of the overall Δ up.

The simulations have shown through the given scenarios that with the target-relative measures of the triggered communication protocol and the modified VO-based reciprocal CA, non-collision is guaranteed while allowing agents to reach their targets with much reduced communication rates.

4.6 Summary

In this chapter, a communication protocol with a target-relative measure-based trigger along with a modified version of the VO-based reciprocal CA algorithm was presented. Under Assumptions 4.1 and 4.2, convex program feasibility maintenance was established in Theorem 4.1 and Algorithm 4.1, and thus non-collision, for all time. A necessary condition was also provided for the VO-based technique to allow the agents to reach zero velocity at their targets. Techniques to reduce the chance of deadlocks were described in the forms of constraint switching and selection. Moreover, a sufficient technique was provided to limit velocity without resorting to quadratic constraints. The communication protocol was designed to trigger updates at opportune times without being continuous. Its effectiveness with the avoidance algorithm was demonstrated in a simulation of five agents.

However, deadlocks can still exist where agents can become unable to reach their target positions especially with larger numbers of agents. This is due to disallowing head-on velocities and having active collision constraints for every other agent in $\mathcal{N}$ at all times. Works in literature typically implement CA locally within some range. However, this would require regular checking to see if an agent has entered within some predefined sensing range.

Furthermore, the model used in this work is the single integrator in 2D. The scope was limited to simplify the problem and obtain a working solution first. However, it should be noted that sudden velocity changes with unbounded acceleration are not necessary to maintain non-collision because Algorithm 4.1 and (4.6) by construction ensures $v_i(t) \in \beta_{i|j}(t) \forall j \in \mathcal{N}/\{i\}$ at all times (including before solving OP 4.2 in the same time step) and β_i is convex. Hence, the double integrator model may be used instead with minimal modifications.

A solution that solves these issues to some extent is presented in the next two chapters. However, the work presented in the sequel does not diminish the usefulness of this chapter. The VO method enables more efficient CA maneuvers due

to its formulation in velocity space. The solution in the next chapter forgoes this advantage and uses instead the position space to allow head-on velocities.

Chapter 5

Hybrid Partition-based CA with Triggered Communication

This chapter describes a collision avoidance (CA) algorithm that is designed as a hybrid of the Voronoi Cell, VO and repulsion methods. The Voronoi Cell method is used to allow head-on velocities unlike the work presented in Chapter 4. The method is also partition-based but in position space. The problem of limited communication is still considered in this work and treated in a similar manner. A priority system is also introduced to solve deadlock resolution for large numbers of agents distributedly. The work in this chapter was published in [103, 104].

The purpose of this chapter is two-fold. Firstly, it presents a complete algorithm that overcomes the limitations of the work presented in Chapter 4. Specifically, the limitations refer to the problem of deadlock and restriction in the number of UAVs and spatial configuration. It is shown later in simulations that large numbers (up to 100) of agents are able to reach their assigned target positions in heavy congestion without deadlock. Secondly, a novel integrated distributed traffic coordination method under a constrained communication framework is presented using a priority system. The method determines whether an agent can pass another and on which side based on comparing its priority.

This chapter is organized as follows. Motivation for this work is explained in the following section. Section 5.2 defines the model, communication system, and objective. Section 5.3 describes the hybrid collision avoidance algorithm. Analysis of deadlock and livelock scenarios is provided in Section 5.4. The event-triggered communication protocol is presented in Section 5.5. Simulation results are recorded in Section 5.6. Finally, some concluding remarks are drawn in Section 5.7.

5.1 Motivation

This chapter builds on the work presented in Chapter 4. Thus the motivation for this work is similar. Additional points are discussed in the following.

As mentioned earlier, the CA algorithm presented here utilizes the Voronoi Cell method reviewed in Section 2.2.2.1 to allow head-on velocities. This method may be used to define spatial partitions between pairs of agents that can guarantee non-collision indefinitely without requiring updates. Partition updates are instead used to enable agents to move towards their assigned target positions. This is similar to the use of VO partitions in Chapter 4. This Voronoi-based partition is used to allow head-on velocities while agents are not trying to pass each other.

The Voronoi Cell was used for collision avoidance in [53]. It was noted that that method resolves deadlocks by identifying the specific scenarios and forcing agents to move parallel to the partitions, which resulted in a last-in-first-out resolution. This sequence may not be desirable as it potentially leaves agents deadlocked in the center of the congestion until it clears. Moreover, the limited communication problem was not considered.

The hybrid algorithm presented in this chapter uses instead the combination of VO and repulsion in addition to the Voronoi Cell to improve deadlock resolution. The VO formulation is used to obtain spatial partitions that allow agents to pass as close as possible on either side. The repulsion is used to enlarge the spacing between agents to allow for this passing to occur more easily. The effectiveness

of this combination in allowing higher priority agents to pass first is shown in simulation in Section 5.6.5.

Deadlocks become a significant problem when constraints are not updated as often due to limited communication bandwidth. If the agents are homogeneous and not carrying unique loads, they may be reassigned target positions [105]. Otherwise, a coordination algorithm is needed to resolve congestion and deadlocks. A centralized method may require multiple communications with all agents affected, taking up communication capacity that can instead be used for other tasks. The distributed algorithm proposed here reduces this requirement while still enabling agents to resolve any congestion.

Similar to Chapter 4, agents are required to move to their assigned targets in finite time, while avoiding collisions with others using pairwise interactions. For safety, non-collision has to be ensured between less frequent and non-periodic instances of communication. This problem is relevant to the transportation and construction applications where congestion may occur if large numbers of agents coincidentally operate within the same small area [7].

5.2 Problem Formulation

5.2.1 Model

The UAVs are modeled as N single integrator agents in 2D space with position,

$$\mathbf{q}_i(t_{m+1}) = \mathbf{q}_i(t_m) + T\mathbf{v}_i(t_m), \quad \mathbf{q}_i \in \mathbb{R}^2, \quad i \in \mathcal{N} = \{1, 2, \dots, N\}, \quad (5.1)$$

where $T = t_{m+1} - t_m$ is the constant control time interval, subscript m denotes the current control time step, $\|\mathbf{v}_i\| \leq v_{\max}$, and $v_{\max}$ is the input constraint. Time, t_m , is dropped from the notation where only variables at current time are considered. In simulations, the control time step of every agent is synchronized.

Note that unlike previous chapters, the discrete-time model is used here to clearly differentiate the notation for control time step, t_m , and the communication time to be defined in the next section. The time interval size does not affect the collision avoidance guarantee. Its effect on the simulation completion time is shown later in the simulation results.

Each agent's instantaneous global position and velocity are known or measurable only by itself. Other agent's positions are only obtainable via wireless communication.

Each agent is represented by a point at the drone's center. The agents are treated as homogenous, and size and shape are accounted for by defining non-collision as maintaining a minimum separation distance, $d_{\min}$, between pairs of agents, i.e.

$$\|\mathbf{q}_{ij}\| = \|\mathbf{q}_i - \mathbf{q}_j\| \geq d_{\min}, \quad i, j \in \mathcal{N}, \quad i \neq j. \quad (5.2)$$

5.2.2 Communication

This work tackles the scenario where other agents' positions are obtained via limited bandwidth communication. Agents are assumed to be connected via a wireless ad hoc network [102], which allows communication between any pair.

In this work, each instance of communication is a simultaneous exchange of information between a pair of agents. In physical application this may be done using the Transmission Control Protocol (TCP) [102].

The interval between communication instances for a pair of agents is time-varying and the communication is event-triggered. The time of each instance is denoted by

$$t_h^{ij} = t_h^{ji}, \quad i, j \in \mathcal{N}, \quad i \neq j, \quad (5.3)$$

where h is the positive integer index denoting the most recent instance. In this work, it is assumed that there are no communication delays. Thus t_h^{ij} is considered

to be an integer multiple of interval T for simplicity. Information received at time $t_h^{ij} = t_m$ is stored and used for input calculation from time $t \geq t_h^{ij}$.

Each instance of communication is initiated by one agent due to an event trigger. A successful exchange of information then blocks communication with others within the same step, t_m . Hence, at any time step, an agent communicates with only one other. Physically, this means each agent maintains only one TCP connection at a time. Additionally, an agent may only initiate communication once per time step, i.e. no reattempts. Since the simulation processes all agents' steps simultaneously, communication initiation conflicts are resolved by agent number, i.e. the agent whose number $i \in \mathcal{N}$ is the smallest among those initiating at the same time gets to communicate.

5.2.3 Objective

The main objective is to design a low complexity distributed algorithm for coordinating the movement of agents of dynamics (5.1) under dense traffic conditions and limited communication so that they reach their pre-assigned target positions, $\mathbf{q}_i^f$, $i \in \mathcal{N}$, in finite time while guaranteeing non-collision as defined in (5.2). The aim is to allow for reduced communication rates without compromising CA and reaching of assigned target positions.

5.3 Hybrid CA Algorithm

5.3.1 Overview

The version of VO suitable for multi-agents, taken from [54], is shown in Figure 5.1(a) and is the set

$$VO_{ij}^\tau = \{\mathbf{v} | \exists t \in [0, \tau], \|\mathbf{v}t - (\mathbf{q}_j - \mathbf{q}_i)\| < d_{\min}\}, \quad (5.4)$$

FIGURE 5.1: Velocity Obstacle with time horizon truncation. a) $VO_{i|j}^\tau$ where some head-on velocities are excluded according to (5.4). The unshaded region denotes the set of agent i 's velocities relative to agent j that avoids collision for time horizon τ . b) Three possible linearizations of the non-convex avoidance set marked by unit normals $\hat{\mathbf{n}}_{i|j}^{r,ccw}$, $\hat{\mathbf{n}}_{i|j}^{r,cw}$ and $\hat{\mathbf{n}}_{i|j}^{r,head}$.

which is the VO from the perspective of agent $i \in \mathcal{N}$ with respect to agent $j \in \mathcal{N}/\{i\}$. $\mathbf{q}_i$ and $\mathbf{q}_j$ are the positions of agents i and j respectively at time $t = 0$. Selecting a relative velocity outside this set prevents collision between agents i and j up to time $t = \tau$. The time horizon, τ , allows for head-on relative velocities in the region bounded by the straight dotted lines from the origin and the arc truncating the shaded cone in Figure 5.1(a). However, the use of fixed time intervals between updates is avoided so that non-collision can be guaranteed without requiring periodic or frequent communication. In Chapter 4, the time horizon was omitted in linearizing the non-convex VO set complement by considering only the straight line sections of the boundary, which are illustrated in Figure 5.1(b) and marked by their respective unit normals $\hat{\mathbf{n}}_{i|j}^{r,cw}$ and $\hat{\mathbf{n}}_{i|j}^{r,ccw}$. Superscripts cw and ccw denote the clockwise and counterclockwise avoidance directions respectively. However, without allowing head-on, agents that become surrounded by others may encounter zero velocity as the only solution, leaving them deadlocked.

The head-on requirement is solved by adapting the Voronoi Cell method. Similar to [68], the linearization of the VO is narrowed to three possibilities for easier coordination of agent movement. The three linearizations are shown in Figure 5.1(b)

FIGURE 5.2: Two CA modes. White regions denote the convex set of allowed positions.

and marked by straight lines with unit normals, $\hat{\mathbf{n}}_{i|j}^{r,cw}$, $\hat{\mathbf{n}}_{i|j}^{r,ccw}$ and $\hat{\mathbf{n}}_{i|j}^{r,head}$. The constraint marked by $\hat{\mathbf{n}}_{i|j}^{r,head}$ is replaced by a linear constraint derived from the Voronoi Cell and not necessarily at the same position as that shown. Using this method, the constraint obtained is on position space and not limited to a time horizon.

Agent movement is coordinated, to enable them to reach their assigned targets, by careful selection of the linearization. Selection is done immediately after information exchange between the communicating agent pair and is constant until the next communication instance. The linearization is split into two modes as shown in Figure 5.2. The first and default mode, called the *Buffered Voronoi Partition* (BVP) mode, consists of the head-on Voronoi Cell-based linear constraint and repulsion. The repulsion is used to expand the space between agents and ensures that the agent remains within the feasible set when transitioning between modes. The second mode, called the *Collision Cone Partition* (CCP) mode, consists of either of the clockwise or counterclockwise constraints from Figure 5.1(b). While the first mode ensures that agents can approach each other up to some distance without being deadlocked, the second is used to enable them to pass each other as close as d_{min} . The same mode and subsequently the direction (clockwise or counterclockwise) in the CCP mode, are selected independently by each agent based on the same information obtained from communication.

The mode for agent i with respect to j is temporarily denoted here by

$$\Omega_{i|j} = \begin{cases} 1 & \text{if CCP mode,} \\ 0 & \text{if BVP mode.} \end{cases} \quad (5.5)$$

Mode and direction selection is based on prioritizing the motion of one agent over another. Typically in literature, selection is based on relative velocity or relative target velocity [54, 68]. This equal-weighted method may lead to deadlocks in congested scenarios such as when agents' target positions are placed close together. It is shown later in simulation that this can result in a deadlock. The proposed priority system described later instead causes higher priority agents to move in their preferred directions first at the expense of others. Those that have reached their assigned targets are then given lower priority. This priority-based distributed coordination avoids deadlocks in the same scenarios.

FIGURE 5.3: Flowchart of the hybrid CA algorithm, which repeats at every control time step, t_m . The relevant sections are also labeled.

For each control time step, t_m , each agent follows the steps summarized in Figure 5.3. Every agent is initialized in the default BVP mode with respect to each other. An event-triggered communication protocol then determines when and to whom each agent communicates. One possible protocol is presented in Section 5.5.

If a communication instance is successful between an agent pair, then an information exchange occurs and the rest is as follows in Figure 5.3. Any mode change relative to another agent is held constant until the next communication with that agent. Otherwise, the constraints are unchanged. The agent then determines its target velocity, which is the velocity by which the agent should ideally move at to reach its target position if it is not subjected to any collision constraint. Subsequently, a convex optimization problem is solved to determine its input velocity, $\mathbf{v}_i$, based on the target velocity. Hence, control is periodic while communication is dictated by event-triggering.

5.3.2 Buffered Voronoi Partitions

For each $i \in \mathcal{N}$, given knowledge of the positions of others and according to [48], the Voronoi Cell may be defined by,

$$\mathcal{V}_i = \{ \mathbf{q} \in \mathbb{R}^2 \mid \|\mathbf{q} - \mathbf{q}_i\| \leq \|\mathbf{q} - \mathbf{q}_j\|, \forall j \neq i \}. \quad (5.6)$$

However, due to (5.2), an additional buffer from the partition is necessary. This is achieved by using the Buffered Voronoi Cell (BVC) concept from [53]. Unlike BVC, the Voronoi partition is applied to constrain velocity. Relative to $j \in \mathcal{N}/\{i\}$, a buffered linear constraint on agent i 's velocity, for time $t_m \geq t_h^{ij}$, is defined as

$$A_{i|j}^{\text{vel}}(t_m) = \{ \mathbf{v} \mid (\mathbf{v} - \mathbf{u}_{i|j}^{\text{v}}(t_m))^{\top} \hat{\mathbf{n}}_{i|j}^{\text{v}}(t_m) \geq 0 \}, \quad (5.7)$$

where

$$\hat{\mathbf{n}}_{i|j}^{\text{v}}(t_m) = \frac{(\mathbf{q}_i(t_h^{ij}) - \mathbf{q}_j(t_h^{ij}))}{\|\mathbf{q}_i(t_h^{ij}) - \mathbf{q}_j(t_h^{ij})\|}, \quad (5.8)$$

$$\mathbf{u}_{i|j}^{\text{v,pos}}(t_m) = \frac{(\mathbf{q}_j(t_h^{ij}) + \mathbf{q}_i(t_h^{ij}))}{2} - \mathbf{q}_i(t_m) + \frac{d_{\min}}{2} \hat{\mathbf{n}}_{i|j}^{\text{v}}(t_m), \quad (5.9)$$

$$\mathbf{u}_{i|j}^{\text{v}} = \frac{\mathbf{u}_{i|j}^{\text{v,pos}}}{T}. \quad (5.10)$$

$\mathbf{u}_{i|j}^{v, \text{pos}}$ is a point on the positional constraint line relative to i 's current position. This constraint is defined as

$$A_{i|j}^{\text{pos}}(t_m) = \left\{ \mathbf{q} \mid \left(\mathbf{q} - \left(\mathbf{u}_{i|j}^{v, \text{pos}}(t_m) + \mathbf{q}_i(t_m) \right) \right)^\top \hat{\mathbf{n}}_{i|j}^v(t_m) \geq 0 \right\} \quad (5.11)$$

and illustrated in Figure 5.4. To adapt the positional constraint to velocity, $\mathbf{u}_{i|j}^v(t_m)$ is used. Any feasible velocity within $A_{i|j}^{\text{vel}}$ will thus not cause the agent to violate $\left(\mathbf{u}_{i|j}^{v, \text{pos}} \right)^\top \hat{\mathbf{n}}_{i|j}^v \geq 0$ within control period T . Note that $\mathbf{q}_j(t_h^{ij})$ and $\mathbf{q}_i(t_h^{ij})$ are information stored by both agents i and j from communication time t_h^{ij} . Both agents use the same information to obtain respective BVP constraints.

FIGURE 5.4: Buffered Voronoi Partition positional constraint, $A_{i|j}^{\text{pos}}$.

5.3.3 Additional Repulsive Buffer

The repulsion on each agent, $i \in \mathcal{N}$ is applied by modifying its target velocity and is designed to repel it up to a common predefined distance from the positional $A_{i|j}^{\text{pos}}$ constraint boundary. The target velocity is modified to keep within the constrained optimization framework and ensure that collision constraints are not violated. The predefined distance, referred to as the *neutral repulsion distance*, d_n , is enforced for conformity and control on how far agents are repelled. The following formulation is by no means the only possibility, but it ensures that, in the absence of other

agents, its target velocity points away from the $A_{i|j}^{\text{pos}}$ boundary if it is less than distance d_n from it.

Agent i 's target velocity is modified with respect to $A_{i|j}^{\text{pos}}(t_m)$ by adding to it

$$\mathbf{v}_{i|j}^{\text{rep}} = \mathbf{v}_{i|j}^{\text{rep},0} + \mathbf{v}_{i|j}^{\text{rep},n}, \quad (5.12)$$

where

$$\mathbf{v}_{i|j}^{\text{rep},0} = \max \left(-\mathbf{v}_i^{\beta, \text{rot}^\top} \hat{\mathbf{n}}_{i|j}^v, 0 \right) \min \left[\max \left(1 - \frac{d - d_n}{d_0 - d_n}, 0 \right), 1 \right] \times \hat{\mathbf{n}}_{i|j}^v, \quad (5.13)$$

$$\mathbf{v}_{i|j}^{\text{rep},n} = k_r \max \left(1 - \frac{d}{d_n}, 0 \right) \times \hat{\mathbf{n}}_{i|j}^v, \quad (5.14)$$

$$\text{and } d = |\mathbf{u}_{i|j}^v \cdot \hat{\mathbf{n}}_{i|j}^v|. \quad (5.15)$$

$\mathbf{v}_i^{\beta, \text{rot}}$ is i 's target velocity before repulsion is added and is determined later, $d_0 \geq d_n$ is the designed *zero repulsion distance* or the maximum distance from the $A_{i|j}^{\text{pos}}(t_m)$ boundary at which the repulsion acts, and $k_r > 0$ is the repulsion gain. $\mathbf{v}_{i|j}^{\text{rep},0}$ in (5.13) serves to cancel out the component of the agent's target velocity opposite the direction of $\hat{\mathbf{n}}_{i|j}^v$. This component is exactly cancelled out at $d \leq d_n$. Hence, in the absence of other constraints, the agent is prevented from approaching any closer than d_n from the $A_{i|j}^{\text{pos}}(t_m)$ boundary. Then $\mathbf{v}_{i|j}^{\text{rep},n}$ in (5.14) is added to repel it if $d < d_n$.

To allow passing between agents, the following lower bound is defined:

$$d_n > \frac{d_{\min}}{2}. \quad (5.16)$$

The resulting closest possible distance between i and j when each is repelled to a minimum of d_n distance from the boundaries of $A_{i|j}^{\text{pos}}(t_m)$ and $A_{j|i}^{\text{pos}}(t_m)$ respectively, is $(2d_n + d_{\min})$, which produces a gap of size $(2d_n - d_{\min})$.

As the agent's system is of only first order and in discrete time, it is not necessary that $d_0 > d_n$. d_0 may be set greater than d_n to provide C^0 smoothness when

$\mathbf{v}_i^{\text{pref}^\top} \hat{\mathbf{n}}_{i|j}^v < 0$ as is typical of repulsion formulations. This is so that the change in velocity due to repulsion is gradual with distance, which is required in practice as acceleration is constrained. Otherwise, setting $d_0 = d_n$ does not affect the purpose of the repulsion which is to repel agents to a common distance d_n .

The use of repulsion instead of shifting the constraint outwards from the obstacle to expand the space is to ensure that the constrained optimization of the agent's velocity is always feasible.

5.3.4 Collision Cone Partitions

In this section, the CCP mode constraints are defined. The constraints for i with respect to j are derived from the straight line boundaries of $VO_{i|j}^\tau$ shown in Figure 5.1(b) and marked by unit normals $\hat{\mathbf{n}}_{i|j}^{\text{r,cw}}$ and $\hat{\mathbf{n}}_{i|j}^{\text{r,ccw}}$, which respectively denote clockwise and counter-clockwise avoidance directions.

Truncation by time horizon in (5.4) is omitted. Instead, the VO in this work is defined as

$$VO_{i|j} = \{\mathbf{v} | \exists t \geq 0, \|\mathbf{v}t - (\mathbf{q}_j - \mathbf{q}_i)\| < d_{\min}\}, \quad (5.17)$$

which is shown in Figure 5.5(a).

The unit normals to $VO_{i|j}$, $\hat{\mathbf{n}}_{i|j}^{\text{r,cw}}$ and $\hat{\mathbf{n}}_{i|j}^{\text{r,ccw}}$ shown in Figure 5.5(a), are used to obtain the two possible half-space constraints for agent i with respect to j . $\hat{\mathbf{n}}_{i|j}^{\text{r,cw}}$ and $\hat{\mathbf{n}}_{i|j}^{\text{r,ccw}}$ can be easily calculated given $d_{\min}$ and relative position $\mathbf{q}_j - \mathbf{q}_i$.

First, define for $t_m \geq t_h^{ij}$,

$$\hat{\mathbf{n}}_{i|j}^{\text{r}}(t_m) = \hat{\mathbf{n}}_{i|j}^{\text{r,cw}}(t_h^{ij}) \quad \text{or} \quad \hat{\mathbf{n}}_{i|j}^{\text{r,ccw}}(t_h^{ij}). \quad (5.18)$$

The choice of clockwise or counterclockwise boundary will be addressed later in Section 5.3.8.

FIGURE 5.5: Velocity Obstacle and Derived CCP constraints. a) $VO_{i|j}$ with the boundaries and their respective normals $\hat{\mathbf{n}}_{i|j}^{r,ccw}$ and $\hat{\mathbf{n}}_{i|j}^{r,cw}$ shown. b) The $B_{i|j}^{vel}$ (light area) and its complement $B_{j|i}^{vel}$ (dark area) derived from the counter-clockwise boundary.

Then the relative position of the constraint to i is

$$\mathbf{u}_{i|j}^{r,pos}(t_m) = \mathbf{q}_i(t_h^{ij}) - \mathbf{q}_i(t_m) . \quad (5.19)$$

The constraint is applied to velocity using

$$\mathbf{u}_{i|j}^r = \frac{\mathbf{u}_{i|j}^{r,pos}}{T} . \quad (5.20)$$

Finally, the linear constraint on agent i 's velocity relative to agent $j \in \mathcal{N}/\{i\}$ for time $t_m \geq t_h^{ij}$ is

$$B_{i|j}^{vel}(t_m) = \{ \mathbf{v} \mid (\mathbf{v} - \mathbf{u}_{i|j}^r(t_m))^\top \hat{\mathbf{n}}_{i|j}^r(t_m) \geq 0 \} . \quad (5.21)$$

An example of $B_{i|j}^{vel}(t_m = t_h^{ij})$ is shown in Figure 5.5(b) where the counter-clockwise boundary and avoidance direction was chosen by both agents i and j , i.e. $\hat{\mathbf{n}}_{i|j}^r = \hat{\mathbf{n}}_{i|j}^{r,ccw}$. Given the same relative position information and choice of avoidance direction, agent j obtains the complementary half-space to $B_{i|j}^{vel}$ in $B_{j|i}^{vel}$.

Similar to $A_{i|j}^{\text{pos}}$, define for agent i with respect to j ,

$$B_{i|j}^{\text{pos}}(t_m) = \left\{ \mathbf{q} \mid \left(\mathbf{q} - \left(\mathbf{u}_{i|j}^{\text{r,pos}}(t_m) + \mathbf{q}_i(t_m) \right) \right)^\top \hat{\mathbf{n}}_{i|j}^{\text{r}}(t_m) \geq 0 \right\}. \quad (5.22)$$

(5.18) - (5.21) are defined and maintained in the agent i 's memory while in the CCP mode with respect to j . Henceforth, CCP constraints that are defined are referred to as active.

5.3.5 Information Exchanged

Three pieces of information are exchanged during an instance of communication between agents i and j . These are defined in the following.

Position at Communication Time: For a communication instance between agents i and j at time $t_h^{ij} = t_m$, i transmits to j and simultaneously internally stores its current position to the variable

$$\mathbf{q}_i(t_h^{ij}) = \mathbf{q}_i(t_m). \quad (5.23)$$

The position of agent j that i receives within the same communication instance is then stored in

$$\mathbf{q}_j(t_h^{ij}) = \mathbf{q}_j(t_m). \quad (5.24)$$

This occurs similarly for agent j , i.e. $\mathbf{q}_j(t_h^{ji}) = \mathbf{q}_j(t_m)$ and $\mathbf{q}_i(t_h^{ji}) = \mathbf{q}_i(t_m)$. The information is stored and used for $t \geq t_m$ until the next communication between i and j .

Avoidance Rule: The avoidance rule for each agent is stored in the variable

$$\gamma_i \in \{0, 1, 2\}. \quad (5.25)$$

This rule is used to determine the directions of rotation for the intent vector, $\boldsymbol{\mu}_{i|j}$, to be defined subsequently, the target velocity, $\mathbf{v}_i^{\text{target}}$, to be defined later, and the

choice of CCP boundary. $\gamma_i = 1$ and $\gamma_i = 2$ signal clockwise and counterclockwise rotation respectively. $\gamma_i = 0$ means that a direction has not been chosen yet and is the default value.

The actual value sent by agent i to j is

$$\gamma_{i|j} = \begin{cases} 0 & \text{if } \sum_{k \in \mathcal{N}/\{i,j\}} \Omega_{i|k} = 0, \\ \gamma_i & \text{otherwise,} \end{cases} \quad (5.26)$$

where $\Omega_{i|k}$ was temporarily defined in (5.5). $\gamma_{i|j}$ is thus set to zero if no other CCP constraints, excluding $B_{i|j}^{\text{vel}}$, are active for agent i . The respective internally stored γ_i may then become modified after exchange and the process is described later in Section 5.3.8.

Intent Vector: The intent vector signals the direction the agent wishes to move with respect to the receiving agent and the magnitude of remaining distance to its assigned target position. It is derived as follows.

First, define the constrained target-relative position vector for agent i ,

$$\mathbf{q}_{\text{fi}} = \frac{\mathbf{q}_i^{\text{f}} - \mathbf{q}_i}{\|\mathbf{q}_i^{\text{f}} - \mathbf{q}_i\|} \min(\|\mathbf{q}_i^{\text{f}} - \mathbf{q}_i\|, \mu_{\text{max}}), \quad (5.27)$$

where $\mathbf{q}_i^{\text{f}}$ is its assigned target position, and $\mu_{\text{max}} > 0$ is a design parameter common to all agents in $\mathcal{N}$.

The intent vector of agent i with respect to j , denoted by $\boldsymbol{\mu}_{i|j}$, is obtained by rotating $\mathbf{q}_{\text{fi}}$ in the direction denoted by γ_i until the resulting vector is on the boundary of the set constrained only by active CCP constraints fulfilling the condition $\mathbf{q}_{\text{fi}}^{\text{T}} \hat{\mathbf{n}}_{i|k}^{\text{r}} < 0$, $k \in \mathcal{N}/\{i,j\}$. CCP constraints where $\mathbf{q}_{\text{fi}}^{\text{T}} \hat{\mathbf{n}}_{i|k}^{\text{r}} \geq 0$ are ignored to ensure that $\boldsymbol{\mu}_{i|j}$ remains unchanged under the same conditions regardless of the mode. Otherwise, the mode may flip between BVP and CCP with each communication.

This is summarized in the following.

$$\boldsymbol{\mu}_{i|j} = \begin{cases} R(-\theta_{i|j}^{\max}) \mathbf{q}_{\text{fi}} & \text{if } \gamma_i = 1, \\ R(\theta_{i|j}^{\max}) \mathbf{q}_{\text{fi}} & \text{if } \gamma_i = 2, \end{cases} \quad (5.28)$$

where

$$\theta_{i|j}^{\max} = \max \left\{ \arccos \left(\frac{\mathbf{q}_{\text{fi}}^\perp \hat{\mathbf{n}}_{i|k}^r}{\|\mathbf{q}_{\text{fi}}^\perp\|} \right) \mid \mathbf{q}_{\text{fi}}^\top \hat{\mathbf{n}}_{i|k}^r < 0 \text{ and } k \in \mathcal{N} / \{i, j\} \right\}, \quad (5.29)$$

$$\mathbf{q}_{\text{fi}}^\perp = \begin{cases} R(-90^\circ) \mathbf{q}_{\text{fi}} & \text{if } \gamma_i = 1, \\ R(90^\circ) \mathbf{q}_{\text{fi}} & \text{if } \gamma_i = 2. \end{cases} \quad (5.30)$$

$R(\cdot)$ is the counter-clockwise rotation matrix. A simple case is shown in Figure 5.6(a).

FIGURE 5.6: Rotation and direction rules. In both plots, white areas represent the resulting intersection of CCP constraints with boundaries denoted by solid lines. a) Rotation of $\mathbf{q}_{\text{fi}}$ to obtain $\boldsymbol{\mu}_{i|j}$. Here $\gamma_i = 2$. b) $B_{i|k}^{\text{vel}}$ derived from Algorithm 1 given that agent i has priority, $\gamma_i = 1$ and $\boldsymbol{\mu}_{i|k}^\alpha$ intersects k 's disc.

5.3.6 Priority System

The priority system is a strict total ordering on the agent set $\mathcal{N}$. The ordering is not fixed and is dependent on the relative magnitude of $\boldsymbol{\mu}_{i|j}$ and $\boldsymbol{\mu}_{j|i}$. It is used

to coordinate movement of agents to avoid deadlocks distributively. The higher priority agent dictates the modes and CCP boundary choices with respect to other agents in a manner that minimizes the possibility of deadlock for itself.

For visualization, the ordering specific to the current states of all agents in $\mathcal{N}$ is represented by the one-to-one mapping specific for those agent states,

$$f: \mathcal{N} \rightarrow \mathcal{M}, \quad (5.31)$$

where $\mathcal{M} = \{1, 2, \dots, N\}$ is the numbered rank set. The smaller the number in $\mathcal{M}$, the higher the priority. Thus $f(i)$ denotes the rank of i given the current states.

Without precision errors, priority is determined by

$$\begin{cases} f(i) < f(j) & \text{if } \|\boldsymbol{\mu}_{i|j}\| \geq \|\boldsymbol{\mu}_{j|i}\|, \forall i, j \in \mathcal{N}, i < j. \\ f(i) > f(j) & \text{otherwise} \end{cases} \quad (5.32)$$

To account for precision in digital implementation, agent i determines its priority relative to j according to

$$\begin{cases} f(i) < f(j) & \text{if } (i < j) \text{ and } (\|\boldsymbol{\mu}_{i|j}\| - \|\boldsymbol{\mu}_{j|i}\| \geq -\varepsilon) \\ f(i) > f(j) & \text{if } (i < j) \text{ and } (\|\boldsymbol{\mu}_{i|j}\| - \|\boldsymbol{\mu}_{j|i}\| < -\varepsilon) \\ f(i) > f(j) & \text{if } (i > j) \text{ and } (\|\boldsymbol{\mu}_{j|i}\| - \|\boldsymbol{\mu}_{i|j}\| \geq -\varepsilon) \\ f(i) < f(j) & \text{if } (i > j) \text{ and } (\|\boldsymbol{\mu}_{j|i}\| - \|\boldsymbol{\mu}_{i|j}\| < -\varepsilon) \end{cases}, \quad i, j \in \mathcal{N}, i \neq j, \quad (5.33)$$

where $\varepsilon > 0$ is the tolerance. The tolerance gives a bias towards the lower numbered agent. In this manner the same result is obtained vice versa by agent j .

The tolerance introduces the possibility of cyclic orders, e.g. $f(i) < f(j)$, $f(j) < f(k)$ but $f(i) > f(k)$, $i, j, k \in \mathcal{N}$, $i \neq j$, $j \neq k$ and $k \neq i$. This occurs when at least two of those agents have intent vectors (e.g. $\boldsymbol{\mu}_{i|j}$) with magnitude less than $q_{\max}$.

This possibility is prevented by conservatively setting for (5.27),

$$q_{\max} < d_0 . \quad (5.34)$$

The upper bound is based on the assumption on the minimum distance between any two target positions to be formally defined later in Section 5.4. This distance is a minimum of $2d_0 + d_{\min}$ to avoid repulsion when all agents have reached their assigned targets. Given this minimum distance and condition (5.34), any pair of agents i and j cannot be both obstructing each other and have their respective $\|\boldsymbol{\mu}_{i|j}\| < \mu_{\max}$. Obstruction here means that any possible $A_{i|j}^{\text{pos}}$ or $B_{i|j}^{\text{pos}}$ obtained by an agent relative to the other based on their current positions does not include the target position. When two agents are no longer obstructing each other, their relative priority becomes irrelevant. This implication is made clearer in the next two sections on mode and boundary selection where priority is involved.

The priority is used to determine which information to use for decision-making. For agent i with respect to j ,

$$\boldsymbol{\mu}_{i|j}^{\alpha} = \begin{cases} \boldsymbol{\mu}_{i|j} & \text{if } f(i) < f(j) , \\ -\boldsymbol{\mu}_{j|i} & \text{otherwise,} \end{cases} \quad (5.35)$$

and

$$\gamma_{i|j}^{\alpha} = \begin{cases} \gamma_{i|j} & \text{if } f(i) < f(j) , \\ \gamma_{j|i} & \text{otherwise.} \end{cases} \quad (5.36)$$

5.3.7 Mode Switching Conditions

Two conditions are used to determine if the agent pair, $i, j \in \mathcal{N}, i \neq j$, should switch from BVP to CCP mode. This mode is held constant until the next communication.

The first condition checks if $\boldsymbol{\mu}_{i|j}^\alpha$ for agent i points into the repulsion field with respect to j , i.e.

$$\hat{\mathbf{n}}_{i|j}^{\text{v}^\top} \left(\boldsymbol{\mu}_{i|j}^\alpha - \mathbf{u}_{i|j}^{\text{v,pos}} \right) - d_0 < 0, \quad (5.37)$$

where d_0 is from (5.13). This condition is used to determine if the higher priority agent needs to cross the $A_{i|j}^{\text{pos}}$ boundary. d_0 is used to shift the constraint line in (5.37) to account for summed repulsion from multiple sources.

The second condition checks for proximity, i.e.

$$\|\mathbf{q}_i(t_h^{ij}) - \mathbf{q}_j(t_h^{ij})\| < d_{\text{prox}}, \quad (5.38)$$

where $d_{\text{prox}} \geq (2d_0 + d_{\text{min}})$. This lower bound is based on when agents i and j are both positioned at d_0 distance away from $A_{i|j}^{\text{pos}}$ and $A_{j|i}^{\text{pos}}$ boundaries respectively. d_0 is used for the same reason as in (5.37). This condition is used to restrict the CCP count. Excessive CCP constraints can result in the origin-only solution, or excessive diversion from the target.

(5.5) for agent i 's mode can now be replaced with

$$\Omega_{i|j} = \begin{cases} 1 & \text{if conditions (5.37) and (5.38),} \\ 0 & \text{otherwise,} \end{cases} \quad (5.39)$$

where $\Omega_{i|j} = \Omega_{j|i} = 1$ represents an active CCP constraint, and deactivated BVP constraint and repulsion. For $\Omega_{i|j} = \Omega_{j|i} = 0$, the CCP constraint is not defined and the BVP constraint and repulsion are active. Agent j arrives at the same mode given the same exchanged information.

5.3.8 CCP Boundary Selection

If $\Omega_{i|j} = \Omega_{j|i} = 1$, where $i, j \in \mathcal{N}, i \neq j$, the agents then need to select from clockwise or counterclockwise avoidance directions in (5.18) in order to define $B_{i|j}^{\text{vel}}$ and $B_{i|j}^{\text{vel}}$.

The boundary selection for (5.18) is replaced with Algorithm 1. For $\gamma_{i|j}^\alpha = 0$, the boundary closer to $\boldsymbol{\mu}_{i|j}^\alpha$ is chosen. This allows the higher priority agent to choose a shorter path. This choice then determines the subsequent rule. If for example i has higher priority with respect to a third agent $k \in \mathcal{N}$, $\gamma_{i|k}^\alpha = 1$ and $\boldsymbol{\mu}_{i|k}^\alpha$ intersects the disc of radius $d_{\min}$ at $\mathbf{q}_k - \mathbf{q}_i$, then the same avoidance direction as before is preferred. This is shown in Figure 5.6(b). Selecting the closest boundary of $VO_{i|k}$ from $\boldsymbol{\mu}_{i|k}^\alpha$ will instead result in overlapping constraints that only allow velocities in the opposite direction. From Figure 5.6(b), it is evident that this would have resulted in the zero-only solution due to the fourth agent l .

Algorithm 5.1 Boundary Selection

```

1: switch  $\gamma_{i|j}^\alpha$  do
2:   case 0
3:      $\hat{\mathbf{n}}_{i|j}^r \leftarrow \begin{cases} \hat{\mathbf{n}}_{i|j}^{r,cw} & \text{if } \boldsymbol{\mu}_{i|j}^\alpha \top \hat{\mathbf{n}}_{i|j}^{r,cw} > \boldsymbol{\mu}_{i|j}^\alpha \top \hat{\mathbf{n}}_{i|j}^{r,ccw} \\ \hat{\mathbf{n}}_{i|j}^{r,ccw} & \text{otherwise} \end{cases}$ 
4:      $\gamma_i \leftarrow \begin{cases} 1 & \text{if } \boldsymbol{\mu}_{i|j}^\alpha \top \hat{\mathbf{n}}_{i|j}^{r,cw} > \boldsymbol{\mu}_{i|j}^\alpha \top \hat{\mathbf{n}}_{i|j}^{r,ccw} \\ 2 & \text{otherwise} \end{cases}$ 
5:   case 1
6:      $\hat{\mathbf{n}}_{i|j}^r \leftarrow \begin{cases} \hat{\mathbf{n}}_{i|j}^{r,cw} & \text{if } \boldsymbol{\mu}_{i|j}^\alpha \top \hat{\mathbf{n}}_{i|j}^{r,ccw} < 0 \\ \hat{\mathbf{n}}_{i|j}^{r,ccw} & \text{otherwise} \end{cases}$ 
7:      $\gamma_i \leftarrow 1$ 
8:   case 2
9:      $\hat{\mathbf{n}}_{i|j}^r \leftarrow \begin{cases} \hat{\mathbf{n}}_{i|j}^{r,ccw} & \text{if } \boldsymbol{\mu}_{i|j}^\alpha \top \hat{\mathbf{n}}_{i|j}^{r,cw} < 0 \\ \hat{\mathbf{n}}_{i|j}^{r,cw} & \text{otherwise} \end{cases}$ 
10:     $\gamma_i \leftarrow 2$ 
11: end switch

```

Remark 1: For case 0, no direction preference has been established yet. This is subsequently set in γ_i in line 4. For cases 1 and 2, the clockwise and counterclockwise boundaries are preferred respectively if $\boldsymbol{\mu}_{i|j}^\alpha$ intersects the $VO_{i|j}$ cone.

5.3.9 Target Velocity

An agent's target velocity is obtained by modifying an intermediate preferred velocity proportional to the relative position of its assigned target. This velocity is rotated for deadlock avoidance. Then the repulsion from (5.12) is added.

The intermediate preferred velocity is defined as

$$\mathbf{v}_i^\beta = \frac{\mathbf{q}_i^f - \mathbf{q}_i}{\|\mathbf{q}_i^f - \mathbf{q}_i\|} \min(k_\beta \|\mathbf{q}_i^f - \mathbf{q}_i\|, v_{\max}), \quad (5.40)$$

where $k_\beta > 0$ is the proportional gain.

$\mathbf{v}_i^\beta$ is then rotated using the same method in (5.28)-(5.30) except with j not omitted to obtain

$$\mathbf{v}_i^{\beta, \text{rot}} = \begin{cases} R(-\phi_i^{\max}) \mathbf{v}_i^\beta & \text{if } \gamma_i = 1, \\ R(\phi_i^{\max}) \mathbf{v}_i^\beta & \text{if } \gamma_i = 2, \end{cases} \quad (5.41)$$

where

$$\phi_i^{\max} = \max \left\{ \arccos \left(\frac{\mathbf{v}_i^\perp \cdot \hat{\mathbf{n}}_{i|k}^r}{\|\mathbf{v}_i^\perp\|} \right) \mid \mathbf{v}_i^{\beta \top} \hat{\mathbf{n}}_{i|k}^r < 0 \text{ and } k \in \mathcal{N} / \{i\} \right\}, \quad (5.42)$$

$$\mathbf{v}_i^\perp = \begin{cases} R(-90^\circ) \mathbf{v}_i^\beta & \text{if } \gamma_i = 1, \\ R(90^\circ) \mathbf{v}_i^\beta & \text{if } \gamma_i = 2. \end{cases} \quad (5.43)$$

$R(\cdot)$ is the counter-clockwise rotation matrix.

The sum of all active repulsions (5.12) is added to obtain

$$\mathbf{v}_i^{\text{sum}} = \mathbf{v}_i^{\beta, \text{rot}} + \sum_{j \in \mathcal{N} / \{i\} \wedge \Omega_{i|j} = 0} \mathbf{v}_{i|j}^{\text{rep}}. \quad (5.44)$$

Finally,

$$\mathbf{v}_i^{\text{target}} = \min(\|\mathbf{v}_i^{\text{sum}}\|, v_{\max}) \frac{\mathbf{v}_i^{\text{sum}}}{\|\mathbf{v}_i^{\text{sum}}\|} \quad (5.45)$$

to ensure that the velocity obtained from convex optimization does not exceed the input constraint.

5.3.10 Convex Optimization on Velocity

The input velocity is determined by quadratic programming. Formally, the convex optimization problem for each agent $i \in \mathcal{N}$ at each control time step t_m is as follows.

Optimization Problem 5.1.

$$\min \|\mathbf{v}_i - \mathbf{v}_i^{\text{target}}(t_m)\| \quad (5.46)$$

subject to

$$\mathbf{v}_i \in A_{i|j}^{\text{vel}}(t_m), \quad \forall j \in \mathcal{N} / \{i\}, \quad \Omega_{i|j}(t_m) = 0 \text{ and } |(\mathbf{u}_{i|j}^{\text{v}}(t_m))^{\top} \hat{\mathbf{n}}_{i|j}^{\text{v}}(t_m)| < v_{\max}, \quad (5.47)$$

$$\mathbf{v}_i \in B_{i|j}^{\text{vel}}(t_m), \quad \forall j \in \mathcal{N} / \{i\}, \quad \Omega_{i|j}(t_m) = 1 \text{ and } |(\mathbf{u}_{i|j}^{\text{r}}(t_m))^{\top} \hat{\mathbf{n}}_{i|j}^{\text{r}}(t_m)| < v_{\max}. \quad (5.48)$$

Since only linear constraints are used, its complexity is also linear. All the linear BVP and CCP constraints include the origin. Hence, constraints further than $v_{\max}$ can be ignored.

5.4 Analysis

This section examines how the proposed algorithm avoids deadlocks and livelocks. Non-collision guarantee is trivial for single-integrator systems. However, inter-agent collision interactions are non-linear, which makes guaranteeing convergence to assigned target positions challenging. A proposition summarizing the algorithm's CA guarantee is first presented. Subsequently, scenarios where deadlocks or livelocks may plausibly occur are examined.

The proposed algorithm requires the following assumptions.

Assumption 5.1. Every agent $i \in \mathcal{N}$ is initialized with knowledge of $\mathbf{q}_j(t_{m=0} = 0)$, $j \in \mathcal{N} / \{i\}$.

The algorithm requires agents' knowledge of the existence and initial position of every other agent in $\mathcal{N}$ to ensure non-collision from $t_{m=0} = 0$. Discovery of new agents to the swarm is not in the scope of this work and may be part of another component algorithm of the drone system.

Assumption 5.2. Each agent's initial position at $t_{m=0} = 0$ is at least $d_{\min}$ distance away from every other agent's.

Assumption 5.2 ensures non-collision for $t_{m=0} = 0$.

Assumption 5.3. Each agent's target position is at least $2d_0 + d_{\min}$ distance away from every other agent's target positions and does not change.

Assumption 5.3 ensures that agents are not affected by repulsion when every agent has reached their targets.

The non-collision guarantee of the algorithm is summarized in the following proposition.

Proposition 5.1. *Under Assumptions 5.1 and 5.2, the hybrid algorithm described in Section 5.3 ensures no collision at any time, $t > 0$.*

Proof. Agents in $\mathcal{N}$ begin with the default BVP mode. From Assumptions 5.1 and 5.2, each agent $i \in \mathcal{N}$, from $t_m \geq 0$ with $t_{h=0}^{ij} = t_{h=0}^{ji} = 0$, is constrained by $A_{i|j}^{\text{vel}}$ from (5.7) with respect to all $j \in \mathcal{N} / \{i\}$. Hence, the closest any two agents can be is $d_{\min}$, which satisfies (5.2).

If a communication instance occurs at some time $t_{h>0}^{ij} > 0$ between i and j , then information (5.23), (5.26) and (5.28) are exchanged. Given the same information, they determine their relative priority and subsequently (5.35) and (5.36) using (5.33). Conditions (5.37) and (5.38) are then checked. Note that at any $t_m = t_h^{ij}$,

$\hat{\mathbf{n}}_{i|j}^v = -\hat{\mathbf{n}}_{j|i}^v$ and $\mathbf{u}_{i|j}^v = -\mathbf{u}_{j|i}^v$, which leads to the same result in (5.37) for both agents.

From (5.39), if $\Omega_{i|j} = \Omega_{j|i} = 1$, i and j then determine $\hat{\mathbf{n}}_{i|j}^r$ and $\hat{\mathbf{n}}_{j|i}^r$ respectively using Algorithm 1. Similarly, $\hat{\mathbf{n}}_{i|j}^{r,cw} = -\hat{\mathbf{n}}_{j|i}^{r,cw}$ and $\hat{\mathbf{n}}_{i|j}^{r,ccw} = -\hat{\mathbf{n}}_{j|i}^{r,ccw}$ thus resulting in the same choice. $B_{i|j}^{\text{vel}}$ and $B_{j|i}^{\text{vel}}$ are then obtained from (5.21), which ensures $\mathbf{v}_i - \mathbf{v}_j \notin VO_{i|j}$ and thus non-collision.

If $\Omega_{i|j} = \Omega_{j|i} = 0$, $A_{i|j}^{\text{vel}}$ and $A_{j|i}^{\text{vel}}$ are updated, or made active with the respective CCP constraints deactivated.

Hence, any mode switch between any pair of agents in $\mathcal{N}$ ensures that (5.2) is satisfied and that mode and boundary choices by both agents are the same. Moreover, both BVP and CCP constraints include the origin. Hence, OP 5.1 always has a solution, which ensures non-collision. $\square$

5.4.1 Deadlock

Definition 5.4.1 (Deadlock). An agent $i \in \mathcal{N}$ is considered to be in a deadlock from some time $t_m = t_0$ if $\|\mathbf{v}_i^\beta(t_m)\| \neq 0$ and Problem 1 results in $\mathbf{v}_i = \mathbf{0}$ for $t_m \geq t_0$.

Remark 1: Note that deadlock here is defined to be permanent, i.e. for $t_m \geq t_0$. If instead the condition occurs only for some $t_m \geq t_0$ and $t_m \leq t_1$ with $t_1 > t_0$, it is called a *temporary deadlock*. These are common and unavoidable.

Two types of deadlocks may happen. These are examined in the following.

5.4.1.1 BVP and Repulsion Deadlock

This deadlock occurs when (5.44) is zero. The cases where this occurs are illustrated in Figure 5.7(a) and 5.7(b). If the repulsion is insufficient to cancel $\mathbf{v}_i^{\beta,\text{rot}}$ due to the interference of opposing sources of repulsion near a vertex of the convex set,

FIGURE 5.7: Deadlock scenarios caused by BVP mode. Shaded spaces bordered by solid lines denote the complement of A_{1j}^{pos} , where $j = 2$ or 3 . B_{1j}^{pos} has a dotted line boundary instead. The unshaded space denotes the set of allowed positions for agent 1. a) $\mathbf{v}_1^{\beta, \text{rot}} = \mathbf{v}_{1|2}^{\text{rep}}$ and is perpendicular to the boundary. b) $\mathbf{v}_1^{\beta, \text{rot}} = \mathbf{v}_{1|2}^{\text{rep}} + \mathbf{v}_{1|3}^{\text{rep}}$ and agent 1 is near the vertex. c) $\mathbf{v}_1^{\beta, \text{rot}} > \mathbf{v}_{1|2}^{\text{rep}} + \mathbf{v}_{1|3}^{\text{rep}}$ and thus $\mathbf{v}_1^{\text{target}}$ points into the vertex with non-zero magnitude. d) Diversion of $\mathbf{v}_1^{\beta}$ to $\mathbf{v}_1^{\text{target}}$ after switching to CCP mode with agents 2 and 3 and $\gamma_i = 2$.

then a deadlock occurs when the origin of the solution space is coincident with that vertex and the optimization solution is zero. This is illustrated in Figure 5.7(c).

Proposition 5.2. *The deadlock cases described above and shown in Figure 5.7(a), 5.7(b) and 5.7(c) are temporary if the agent in deadlock has higher priority than the agents causing the obstructing repulsion and BVP constraints.*

Proof. First assume that the agents obstructing a deadlocked agent $i \in \mathcal{N}$ are stationary, i.e. $\mathbf{v}_j = 0, j \in \mathcal{N} / \{i\}$. Otherwise, their motion may break its deadlock.

When i communicates with one of the other agents, conditions (5.37) and (5.38) are checked. If (5.38) is not met then the respective $A_{i|j}^{\text{pos}}$ boundary is shifted away from i due to $d_{\text{prox}} > (2d_0 + d_{\text{min}})$ and $\left| \left(\mathbf{u}_{i|j}^{\text{v,pos}} \right)^\top \hat{\mathbf{n}}_{i|j}^{\text{v}} \right| < d_0$. This allows i to approach closer and is thus not deadlocked.

If (5.37) and (5.38) are met then a switch to CCP mode occurs for i and j . (5.37) is satisfied because $\boldsymbol{\mu}_{i|j}^\alpha = \boldsymbol{\mu}_{i|j}$ from (5.35) due to i 's higher priority and it points towards the $A_{i|j}^{\text{pos}}$ boundary. This disables the repulsion and diverts $\mathbf{v}_i^\beta$ according to (5.41) - (5.43) making it parallel to the new $B_{i|j}^{\text{vel}}$ boundary and in the direction obtained from Algorithm 5.1.

If agent i remains deadlocked due to $\mathbf{v}_i^{\beta,\text{rot}}$ pointing towards the other $A_{i|j}^{\text{pos}}$ boundary, the same process as above occurs and $\mathbf{v}_i^\beta$ is diverted accordingly. Note that $\boldsymbol{\mu}_{i|j}$ also points in the same direction as $\mathbf{v}_i^{\beta,\text{rot}}$ thus satisfying (5.37). An example of this result is shown in Figure 5.7(d) where $\gamma_i = 2$.

Hence, the algorithm ensures that a non-zero solution to OP 5.1 is obtained by agent i by disabling repulsion and diverting $\mathbf{v}_i^\beta$ to be parallel to the $B_{i|j}^{\text{vel}}$ boundary. $\square$

5.4.1.2 CCP Overlap Deadlock

It is possible for multiple overlapping CCPs to be simultaneously defined resulting in zero as the only solution to OP 5.1. Figure 5.8 shows a scenario illustrating this overlap.

The scenario described occurs when the agent is surrounded by others. The agent may be surrounded such that there are either obvious gaps between each pair of surrounding agents such as in Figure 5.8, no gaps such as in Figure 5.9, or some variation in between. The latter two cases are more complicated to analyze as the agents are likely in motion due to Assumption 5.3. Hence, the former is examined first.

FIGURE 5.8: Example of agents in a deadlock caused by CCP mode where agent 1 is surrounded by 7 other agents. Colored spaces bordered by dotted lines of the same color denote the complement of $B_{1|j}^{pos}$, where $j \neq 1$. Complements of $A_{1|j}^{pos}$ are denoted similarly but with solid lines. The resulting white space denotes the set of allowed positions for agent 1. The combination of constraints in (b) and (c) result in deadlock. a) Agent configuration without CCP or BVP constraints shown. b) CCP constraints that include target q_1^f . c) CCP constraints that do not include target q_1^f . d) Only BVP constraints.

The purpose of examining the scenario in Figure 5.8 is to show how the algorithm breaks the CCP overlap deadlock by systematic selection of constraint mode and boundaries.

Proposition 5.3. *Consider the scenario where agent $i \in \mathcal{N}$ is surrounded by lower priority agents such that gaps between discs of radius d_{min} exist where a line from q_i can pass without intersecting them, its target position is on the opposite side of the obstructing agents, and that the current set of CCP and BVP constraints are*

FIGURE 5.9: An example of agent 1 surrounded with no gaps between discs of minimum separation distance, $d_{\min}$, denoted by dotted circles. Agents are denoted by colored circles and numbered.

such that the only solution to OP 5.1 is zero. A representative example is shown in Figure 5.8 where agent 1 is surrounded. Then given Assumption 5.3, the algorithm in Section 5.3 ensures that the deadlock is temporary.

Proof. For this proof Figure 5.8 is used as an example scenario. In this scenario, the highest priority agent is 1.

First assume that the surrounding agents are stationary, i.e. $\mathbf{v}_j = 0, j \in \mathcal{N} / \{i\}$. Otherwise, their motion may break the deadlock.

Then the condition (5.38) is also assumed to be satisfied with respect to the surrounding agents. Otherwise, a switch to BVP occurs which means that there is room to move.

The direction in which agent 1 escapes the group is dependent on the communication sequence. The constraints grouped in Figure 5.8(c) are such that $\mathbf{q}_{\text{fl}}^T \hat{\mathbf{n}}_{1|j}^r < 0$. From (5.28) - (5.30), any changes to these constraints will modify $\boldsymbol{\mu}_{1|j}$, which affects the choice of mode and CCP boundary with respect to the remaining agents.

First, consider that agent 1 sequentially communicates, in any order, with all the surrounding agents except those whose constraints are grouped in Figure 5.8(c). For agents 3, 4 and 5, if (5.37) is not satisfied due to $\boldsymbol{\mu}_{1|j}$ pointing away, the mode is switched to BVP and is shown in Figure 5.10(b). Alternatively, if (5.37) and (5.38)

are met for the same agents, and since agent 1 has higher priority and $\boldsymbol{\mu}_{1|j}$ does not point towards any of the respective $d_{\min}$ discs, then the CCP boundaries are chosen using Algorithm 1 such that $\boldsymbol{\mu}_{1|j}$ remains within the new $B_{1|j}^{\text{vel}}$. In either case, $\boldsymbol{\mu}_{1|j}$ remains unchanged. For agent 2, $B_{1|2}^{\text{vel}}$ remains unchanged by the same process. For agent 8, a switch to CCP occurs and the clockwise boundary is chosen. The new constraints relative to 2 and 8 are shown in Figure 5.10(a). The resulting convex set is shown in Figure 5.10(c) where a clear exit is shown. Note that repeated communications with the same agents produce the same results again since $\boldsymbol{\mu}_{1|j}$ remains unchanged.

Next, consider if agent 1 communicates with 6 or 7. For agent 6, $\boldsymbol{\mu}_{1|6}$ is the same as $\boldsymbol{\mu}_1$ in Figure 5.8(c) based on (5.28) - (5.30). Regardless of mode, and following Algorithm 1, there will be no change to $\boldsymbol{\mu}_{1|j}$ for the other agents, which then produces the same result as before. However, for agent 7, $\boldsymbol{\mu}_{1|7}$ is the vector $\mathbf{q}_{f1}$ rotated counterclockwise ($\gamma_1 = 1$) until it is parallel to the boundary of $B_{1|6}^{\text{vel}}$ in Figure 5.8(c) according to (5.28) - (5.30). This results in a boundary switch for $B_{1|7}^{\text{vel}}$ from counterclockwise to clockwise following Algorithm 1. Subsequently, $\boldsymbol{\mu}_{1|j} = \boldsymbol{\mu}_{1|7}$ for $j \neq 1$, and $j \neq 7$. Following the same process of checking (5.37) and (5.38) and boundary selection a new exit between agents 6 and 7 is obtained.

In summary, the algorithm determines mode and CCP boundary selection for j where $\mathbf{q}_{f1}^\top \hat{\mathbf{n}}_{1|j}^r \geq 0$ using the same $\boldsymbol{\mu}_{1|j}$. Since $\boldsymbol{\mu}_{1|j}$ points towards a gap, mode and boundary selection are designed to enable an exit through it. Agents may be added to the set of agents satisfying $\mathbf{q}_{f1}^\top \hat{\mathbf{n}}_{1|j}^r \geq 0$ such as with agent 7, which may then modify the shared $\boldsymbol{\mu}_{1|j}$ as highlighted in the example. However, due to the definition of $\boldsymbol{\mu}_{1|j}$, agents are never removed from that set in this scenario. By repeating communications with all the surrounding agents, the algorithm can eventually reach a constraint combination where $\boldsymbol{\mu}_{1|j}$ is no longer modified. $\square$

The process described becomes partially irrelevant when considering the latter two surround cases described earlier. Where such discs overlap, Algorithm 1 along with the diversion of $\mathbf{v}_i^\beta$ ensures that the outside boundary of the overlapping discs is

FIGURE 5.10: CCP and BVP constraints updated such that an opening becomes available which breaks the deadlock scenario in Figure 5.8. a) CCP constraints that include target $\mathbf{q}_1^f$. b) Only BVP constraints. c) Superimposition of constraints in (a) and (b) showing the resulting convex set of allowed positions.

chosen. This maximizes the agents ability to keep moving as long as it has priority. However, in some cases this may be insufficient to avoid deadlock. This is apparent in the extreme case in Figure 5.9.

The higher priority agent then depends on the others to move first. These cases are difficult to examine in a general manner. Instead, the following aspects of the algorithm explains why the surrounding agents may move first.

1. Target Separation: Assumption 5.3 ensures that a minimum gap of $2d_0 - d_{\min}$ between discs of $d_{\min}$ centered at each agent when agents have reached their target. Hence, some of the surrounding agents will have $\mathbf{v}_i^\beta \neq 0$.
2. Repulsion: Overlapping $d_{\min}$ discs means that the pair of agents are less than $2d_0 + d_{\min}$ apart. In BVP mode, both agents will be less than d_0 distance from the respective A_{ij}^{pos} boundary and thus encounter repulsion.
3. Velocity Diversion: (5.41)-(5.43) diverts the velocity of both agents in CCP mode relative to each other. This forces the blocking agent to also move.

Based on these factors, agent configurations such as that shown in Figure 5.9 are not permanent. Surrounding agents will move apart which opens up gaps between them.

5.4.2 Livelock

Definition 5.4.2 (Livelock). A group of agents is said to be in a livelock if a set of motions or constraint changes by the group over some period of time is repeated such that no progress is made towards their respective targets.

By the definition given, livelock scenarios may be too complex to be examined in generalized scenarios. The assumption and algorithm design choices are instead explained here.

Target Separation: As mentioned earlier for (5.34) and in Section 5.4.1.2, the minimum value $2d_0 + d_{\min}$ between agents' assigned target position set in Assumption 5.3 is necessary to ensure that each agent is not affected by repulsion. If target positions are set closer, an oscillatory-like motion can occur.

Permanent Target: Assumption 5.3 also states that the target positions do not change. If target positions can be changed without explicit control, it can lead to livelock in some scenarios. One example is where an agent's target switches back and forth between two points without first reaching either point.

Priority System: The algorithm takes advantage of the communication system to exchange the intent vector, $\mu_{i|j}$, from (5.28) and the avoidance direction rule, $\gamma_{i|j}$, used in Algorithm 1.

By comparing priority and using the $\mu_{i|j}$ and $\gamma_{i|j}$ of the higher priority agent, consistency is enforced in the distributed decision-making. $\mu_{i|j}$ is used to signal the direction in which agent i is supposed to move as its magnitude and direction do not change quickly. In comparison, agent i 's velocity may change quickly subject to constraint interactions which makes it unreliable as a signal. From Algorithm 1,

γ_i is established with the first CCP constraint and then subsequently kept constant as long as i has higher priority than the agents it communicates with. This also maintains a consistent rotation direction for (5.28)-(5.30) and (5.41)-(5.43). This consistency prevents back-and-forth switching between modes or CCP boundaries.

Minimum Spacing: The required minimum spacing between assigned target positions is defined here. Consider the hexagonally close-packed extreme scenario where agent 1 has its start position in the middle of layers of stationary surrounding agents positioned at $2d_0 + d_{\min}$ distance apart as shown in Figure 5.11(a). If agent 1 is unable to find a free path out, it may end up in a livelock moving in the manner shown.

From Figure 5.11(b), if the separation distance is too small, a free straight line path, which is required to escape, will become unavailable due to the discs of minimum separation distance denoted by dotted circles. From (5.38), $d_{\text{prox}} \geq 2d_0 + d_{\min}$. Since, agents are spaced only $2d_0 + d_{\min}$ apart, in Figure 5.11(b), communications between agent 9 and 1 will likely satisfy (5.37) and (5.38) and establish a $B_{1|9}^{\text{vel}}$ that bridges the gap between agents 2 and 9 or 3 and 9. For the latter, this limits its movement to the corridor denoted by the pair of dotted lines. To ensure that a straight line path exists, a lower bound on d_0 is established as follows.

$$\begin{aligned} \left(\cos \frac{\pi}{6}\right) (2d_0 + d_{\min}) &> 2d_{\min} \\ d_0 &> \left(\frac{2 - \cos \frac{\pi}{6}}{2 \cos \frac{\pi}{6}}\right) d_{\min}. \end{aligned} \tag{5.49}$$

An upper bound on d_{prox} is also necessary to avoid preventing the agent from entering the free path. To increase the range of angles at which it can enter the free path, it is observed that its condition (5.38) should only be satisfied for at most three agents ahead of it that are adjacent to the free path at a time. Otherwise, for some positions, both CCP constraints with respect to 1's nearest two agents may intersect with the disc of the fourth agent ahead, which would lead to a zero-only

FIGURE 5.11: The hexagonal close-packed scenario where agents are positioned $2d_0 + d_{\min}$ apart from their neighbors. Each agent is denoted by a colored circle and numbered. The discs of distance $d_{\min}$ are denoted by dotted circles. Agent 1's target is denoted by a cross. a) Full view. The circular arrow denotes the livelock motion for agent 1. b) Zoomed in view. The two dotted lines bound one possible free path agent 1 can take between layers.

solution. By this reasoning,

$$2d_0 + d_{\min} \leq d_{\text{prox}} < \frac{3}{2}(2d_0 + d_{\min}). \quad (5.50)$$

5.5 Trigger-based Communication Protocol

In this section, the distributed communication protocol used in simulations is described. Communications with other agents are needed to update the constraints so that it can advance towards its assigned target position. The method described here may not be the most efficient, but it was sufficient in ensuring all agents reach their target positions.

A mixture of time interval and proximity measures are used, which are compared against some threshold. These comparisons are referred to as triggers. The triggered agent then initiates communication as described in Section 5.2.2.

The protocol is outlined in Figure 5.12. A "combined" timer is checked first. This timer checks if the time interval since the last successful communication with any

FIGURE 5.12: Flowchart outlining the Trigger-based Communication Protocol.

agent exceeds the set threshold, $t_{\text{comb}} > 0$, i.e. for agent $i \in \mathcal{N}$

$$t_m - \max \{t_h^{ij} \mid j \in \mathcal{N}, j \neq i\} > t_{\text{comb}} \quad (5.51)$$

If (5.51) is not met, then no communication is initiated. Otherwise, the agent determines which of the two triggers to check. The BVP and CCP triggers are applicable only to the respectively named modes. If the previous successful communication was caused by the CCP trigger, then the BVP trigger is checked first, and vice versa. Subsequently, if the first checked trigger is not met, the other is checked. If both are not met, no communication is initiated. Otherwise, the agent proceeds to initiate the communication with the other agent as identified by the trigger.

5.5.1 BVP Trigger

The BVP trigger is used to check i 's proximity to the $A_{i|j}^{\text{pos}}$ boundaries for which $\Omega_{i|j} = 0$, $j \in \mathcal{N}/\{i\}$. Proximity to $A_{i|j}^{\text{pos}}$ boundaries are checked along a predefined direction. The closest is then compared to a threshold. Three directions are identified and the order in which they are checked is cycled according to a queue. For agent $i \in \mathcal{N}$, the measure to be compared to a threshold is defined as

$$\Lambda_i^m = \max \left\{ \Lambda_{i|j}^{\text{prox},m} \mid j \in \mathcal{N}/\{i\}, \Omega_{i|j} = 0, \text{ and } \Lambda_{i|j}^{\text{dir},m} > 0 \right\}, \quad (5.52)$$

where superscript $m \in \{1, 2, 3\}$ denotes one of three directions, $\Lambda_{i|j}^{\text{dir},m}$ is the direction component, and $\Lambda_{i|j}^{\text{prox},m}$ is the proximity component.

The proximity component is defined as

$$\Lambda_{i|j}^{\text{prox},m}(t_m) = \max \left[\left(1 - \frac{-\mathbf{u}_{i|j}^{\text{v,pos}}(t_m)^\top \hat{\mathbf{n}}_{i|j}^{\text{v}}(t_m) - d_0}{d_\Lambda \Lambda_{i|j}^{\text{dir},m}(t_m)} \right), 0 \right], \quad (5.53)$$

where $d_\Lambda > 0$ is a design parameter. Note that the values of $\mathbf{u}_{i|j}^{\text{v,pos}}(t_m)$ and $\hat{\mathbf{n}}_{i|j}^{\text{v}}(t_m)$ are based on $\mathbf{q}_i(t_h^{ij})$ and $\mathbf{q}_j(t_h^{ij})$, $t_h^{ij} < t_m$, i.e. before any potential update to $A_{i|j}^{\text{pos}}$ within the same time step, t_m . The value of $\Lambda_{i|j}^{\text{prox}}$ increases linearly from zero as the agent approaches the $A_{i|j}^{\text{pos}}$ boundary closer than $d_\Lambda + d_0$, and $\Lambda_{i|j}^{\text{prox}} = 1$ when $-\mathbf{u}_{i|j}^{\text{v,pos}\top} \hat{\mathbf{n}}_{i|j}^{\text{v}} = d_0$. This formulation simplifies threshold selection to $]0, 1]$.

The direction component is the cosine of the angle between the boundary normal and the chosen direction, i.e.

$$\Lambda_{i|j}^{\text{dir},m}(t_m) = -\hat{\mathbf{n}}_{i|j}^{\text{v}}(t_m)^\top \frac{\mathbf{w}_i^m(t_m)}{\|\mathbf{w}_i^m(t_m)\|}, \quad m \in \{1, 2, 3\}, \quad (5.54)$$

where

$$\mathbf{w}_i^1(t_m) = \mathbf{v}_i^\beta(t_m), \quad (5.55)$$

$$\mathbf{w}_i^2(t_m) = \mathbf{v}_i^{\text{sum}}(t_{m-1}), \quad (5.56)$$

$$\mathbf{w}_i^3(t_m) = -\hat{\mathbf{n}}_{i|j}^{\text{v}}(t_m). \quad (5.57)$$

$\Lambda_{i|j}^{\text{dir},m}(t_m)$ is used to determine the distance to the constraint boundary along w_m . $m = 1$ to 3 respectively represent the direction towards the agent's target, the previous diverted and repelled velocity at t_{m-1} , and the nearest boundary. Each direction on its own is insufficient as without tracking which agents it has communicated with, a single direction might lead to repeated attempts to update a constraint without producing any change, leaving it deadlocked.

Definition 5.5.1 (Queue). A queue is a first-in-first-out data structure used in the algorithm to track the order for processing a list of items. An item that is

first in the queue is checked first. If the algorithm's conditions are satisfied, the item is either removed or moved to the back of the queue while the rest are shifted forward. Otherwise, the next item may be checked and so on. The queue length may vary. Any new items are added to the back.

A queue is used to process the trigger. The three directions, $m = 1$ to 3, are initially queued in any order. The items in this queue are not removed but moved to the back.

The first condition checked is

$$\Lambda_i^m > \Lambda^{\text{threshold}}, \quad (5.58)$$

where $\Lambda^{\text{threshold}} \in]0, 1]$. If (5.58) is satisfied, then agent i initiates the communication with the agent j that produced the largest $\Lambda_{i|j}^{\text{prox},m}$ in (5.52). Otherwise, the next in queue is checked. If none satisfies (5.58), the queue order is unchanged.

The second condition required is a successful communication where the requester and responder exchange information. If this is not satisfied, the queue is unchanged.

5.5.2 CCP Trigger

A queue is also used by agent $i \in \mathcal{N}$ to cycle through the list of $j \in \mathcal{N} / \{i\}$ where $\Omega_{i|j} = 1$. The queue is initialized empty with agents added as the respective CCP constraints are defined. As before, two conditions are checked.

The first condition is

$$t_m - \max \{t_h^{ij} \mid j \in \mathcal{N} / \{i\} \wedge \Omega_{i|j} = 1\} > t_{\text{CCP}}, \quad (5.59)$$

where $t_{\text{CCP}} > 0$. If (5.59) is satisfied, then agent i initiates the communication with the first agent j in the queue.

The second condition, as before, checks if the initiated communication is successful. If it is, the agent is removed from the queue. Otherwise, the queue is unchanged and the next in line is not checked.

The number of active CCP constraints at any time is small. This method ensures that all are eventually checked.

5.6 Simulation

Three main scenarios are presented here: a small-scale demonstration, a circle antipodal swap, and a shuffling crowd. In addition, the effects of the following are separately examined: no priority system, no repulsion, and different control time intervals T . Videos of the simulations are available online¹. Table 5.1 lists parameters with the relevant equations.

TABLE 5.1: Simulation Parameters

T (5.1)	0.04s	$v_{\max}$ (5.1)	3.0m/s
$d_{\min}$ (5.2)	1.0m	d_n (5.13), (5.16)	0.7m
d_0 (5.13), (5.49)	0.84m	k_r (5.14)	1.0
$q_{\max}$ (5.27), (5.34)	0.82m	ε (5.33)	0.01
d_{prox} (5.38), (5.50)	3.1m	k_β (5.40)	0.7
t_{comb} (5.51)	$3T$	d_Λ (5.53)	0.5m
$\Lambda^{\text{threshold}}$ (5.58)	0.01	t_{CCP} (5.59)	$5T$

5.6.1 Small-Scale Algorithm Demonstration

Agent 1 is tasked with a target located on the other side of three stationary agents positioned in a triangle. Figure 5.13 is used to show step-by-step how agent 1 passes through the group. Agent 1 first approaches agent 2 while in BVP mode at $t = 0.1s$. At $t = 0.6s$, agent 1 switches to CCP mode relative to agent 2 because

¹<https://www.youtube.com/playlist?list=PLv1PUvyuHeFFySbsIw5EYrURTA9S7zi03>

FIGURE 5.13: Agent 1 passing between 3 other stationary agents. Agent 1's target position is denoted by the cross. Minimum separation boundaries are denoted by dotted circles. Only constraints $A_{1|j}^{\text{pos}}$ and $B_{1|j}^{\text{pos}}$, $j \in \mathcal{N}/\{1\}$, are shown and the boundaries are denoted respectively by thick and thin lines of corresponding colors. While in CCP mode, the corresponding BVP constraint is converted to dashed lines. The white region is the intersection of all the constraints.

conditions (5.37) and (5.38) are met. Its velocity is then diverted parallel to it by (5.41) - (5.43). As it approaches 3, at $t = 1.2s$, it switches to CCP mode relative to agent 3. This repeats with 4 as it passes between 2 and 3 at $t = 1.8s$. From $t = 1.8s$ onwards, triggered communications continue to update the constraints. Agent 1 has priority and $\mu_{1|2}$ and $\mu_{1|3}$ point away from 2 and 3. This causes a switch back to BVP with respect to both. $B_{1|4}^{\text{vel}}$ is updated until it no longer blocks agent 1's motion to the target as seen at $t = 3.1s$.

5.6.2 Antipodal Swap of 100 Agents

100 agents are positioned at start on vertices of a regular polygon with edge length 3m and target positions set antipodal. This demonstrates the algorithm's feasibility with a large agent count as well as the ability to push through the center of the congestion instead of detouring around. All agents reached within 0.2m of their target in under 264s.

The simulation is presented in Figure 5.14. At $t = 20s$, the effect of repulsion can be observed as the agents are spaced semi-regularly outwards from the center. Agents from the outside are not able to switch to CCP mode with inner agents as they are of higher agent number and thus lower priority. Hence, they remain in BVP mode and maintain the repulsion spacing. The lower numbered agents (in blue, and 1, 19 and 35) are able to choose their preferred boundaries for the CCP constraints and thus move in a relatively direct path to their target. Agents (66 and 82) who were surrounded by higher priority agents are diverted backwards due to the rotated $\mathbf{v}_i^{\beta, \text{rot}}$ from (5.41).

5.6.3 Crowd Shuffling of 50 Agents in 100 Randomized Simulations

100 randomized simulations with 50 agents were conducted to demonstrate the algorithm's effectiveness in heavily congested scenarios. The start and target positions for 50 agents were randomized within a $24 \times 24m$ square with minimum spacing of 2.8m between agents as shown in Figure 5.15(a). The random number generator (RNG) seed was varied from 1-100. All simulations were completed successfully with the slowest taking 141s. A simulation is complete when all agents have reached within 0.2m of their target. Selected simulations are available online. Figure 5.15(b) shows the overall convergence progress of the 50 agents towards their targets. Figure 5.16 shows the log of communications for the agent that was last to reach its target in the slowest simulation.

FIGURE 5.14: 100 agents ordered according to agent number on a circle with their target positions set antipodal. Agents are colored according to the blue-green-red spectrum starting with blue for agent 1. Position trails for agents 1, 19, 35, 66 and 82 are shown.

Table 5.2 highlights communication and maximum instantaneous CCP counts. The first and second columns refer to successful communications with the latter summing both requests and responses for an agent. The highest count for one agent is 1454, which belongs to agent 47 in simulation 48 that took 141s to complete. This means an average communication rate of 10.3Hz, which is less than half the control frequency of 25Hz. The communication constraint described in Section

TABLE 5.2: Additional Results of 100 Simulations

	Initiated Comm.	Total Comm.	Max. CCPs
Max.	778	1454	6
Min.	21	86	0
Median	554	947	3

5.2.2 already limits an agent's communication rate with the control frequency as the upper bound. Moreover, agents are only able to communicate with one other agent at each control step. The results show that the algorithm is able to operate at much lower communication rates, while still ensuring agents reach their target positions without collision. The third column refers to the maximum number of active CCP constraints for an agent in a simulation. This count shows the extent by which conditions (5.37) and (5.38) limit the number of agents which it is in CCP mode with.

FIGURE 5.15: 50 agent shuffling simulation initialization and key completion times. a) 50 agents starting positions randomized using RNG seed 48 within a 24×24 m square with minimum 2.8m spacing. b) Sum of distance remaining to respective targets for 50 agents as a fraction of the sum of start to target distances.

FIGURE 5.16: Log of communication instances for agent 47 from the simulation with RNG seed 48. Red circles and blue crosses denote successful communication instances with the corresponding agent on the vertical axis that were requested or responded to respectively by agent 47.

5.6.4 Simulation Without Priority System

A specific scenario was designed to show that the CCP overlap deadlock described in Section 5.4.1.2 can occur without the priority system described in Section 5.3.6.

The priority system is first disabled by making the following changes. The intent vector in (5.28) is replaced by

$$\boldsymbol{\mu}_{i|j} = \begin{cases} R(-\theta_i^{\max}(\gamma_i = 1))\mathbf{q}_{\text{fi}} & \text{if } \theta_i^{\max}(\gamma_i = 1) < \theta_i^{\max}(\gamma_i = 2) , \\ R(\theta_i^{\max}(\gamma_i = 2))\mathbf{q}_{\text{fi}} & \text{otherwise.} \end{cases} \quad (5.60)$$

This means that the smaller diversion of the two directions is chosen. The variable $\boldsymbol{\mu}_{i|j}^{\alpha}$ from (5.35) is replaced with

$$\boldsymbol{\mu}_{i|j}^{\alpha} = \boldsymbol{\mu}_{i|j} - \boldsymbol{\mu}_{j|i} , \quad (5.61)$$

giving equal weight to both. Algorithm 1 is simplified to

$$\hat{\mathbf{n}}_{i|j}^{\text{r}} = \begin{cases} \hat{\mathbf{n}}_{i|j}^{\text{r,ccw}} & \text{if } \boldsymbol{\mu}_{i|j}^{\alpha} \top \hat{\mathbf{n}}_{i|j}^{\text{r,cw}} < 0 , \\ \hat{\mathbf{n}}_{i|j}^{\text{r,cw}} & \text{otherwise.} \end{cases} \quad (5.62)$$

Similar to (5.60), (5.41) is replaced by

$$\mathbf{v}_i^{\beta, \text{rot}} = \begin{cases} R(-\phi_i^{\max}(\gamma_i = 1))\mathbf{v}_i^\beta & \text{if } \phi_i^{\max}(\gamma_i = 1) < \phi_i^{\max}(\gamma_i = 2), \\ R(\phi_i^{\max}(\gamma_i = 2))\mathbf{v}_i^\beta & \text{otherwise.} \end{cases} \quad (5.63)$$

FIGURE 5.17: Deadlocked simulation where the priority system is disabled. Agent 1 moves to its target denoted by the cross while agents 2-5 are already coincident with their own target positions. Only constraints $A_{1|j}^{\text{pos}}$ and $B_{1|j}^{\text{pos}}$, $j \in \mathcal{N} / \{1\}$, are shown and are denoted respectively by thick and thin lines of corresponding colors. The white region is the constraint intersection.

In this simulation, agents 1-5 are positioned initially at coordinates $(0, 1.5)$, $(3, 1.5)$, $(3, -1.5)$, $(5.3, 0)$ and $(5.7, -2.7)$ respectively. Agents 2-5 have target positions coincident with their initial. Agent 1's target position is $(9, 0)$. The transition to deadlock is shown in Figure 5.17. At $t = 2.0s$, agent 1 is in CCP mode relative to agents 2 and 3, and is approaching the $A_{1|4}^{\text{pos}}$ boundary. Between $t = 2.0s$ and $t = 2.1s$, conditions (5.37) and (5.38) are met resulting in a switch to CCP mode. From (5.62), the counterclockwise boundary is chosen. The new $B_{1|4}^{\text{vel}}$ results in the zero-only solution. Further communication with 2 and 3 produces no changes to the constraints which leaves agent 1 deadlocked. The algorithm with priority system was not deadlocked.

5.6.5 Simulation Without Repulsion

For this simulation, the algorithm's performance when the repulsion is disabled is compared. Repulsion is disabled by replacing (5.44) with

$$\mathbf{v}_i^{\text{sum}} = \mathbf{v}_i^{\beta, \text{rot}} . \quad (5.64)$$

The antipodal swap scenario in Section 5.6.2 is used for comparison, except with 50 agents instead of 100.

For a clearer comparison of the effect of repulsion on overall completion time and the order in which agents reach their target positions, the agents are grouped in fives according to their number order and their completion times are averaged. This result is presented in Figure 5.18. Repulsion is not critical to ensuring that agents can reach their target positions, but it affects the ability of a higher priority agent to push through a congestion instead of going around. However, it should be noted that the parameters and target position spacing remain the same.

From Figure 5.18 it is observed that for the case without repulsion, all groups except the first reach their targets faster than the corresponding groups in the case with repulsion. Without repulsion, the highest priority agents in group 1 become blocked as it attempts to pass through the center of the congestion. They become diverted into a long detour or reversal which slows their progress. Other groups are more quickly diverted from the center when encountering lower priority agents causing them to move around the congestion instead of through. Interestingly, despite slowing down the highest priority group, the entire simulation is completed more quickly without repulsion.

5.6.6 Effect of Control Time Interval

The effect of varying the control time interval, T , is presented in Figure 5.19. As before, the antipodal swap scenario in Section 5.6.2 is used for comparison, except

FIGURE 5.18: Comparison of group averaged completion times with and without repulsion for an antipodal swap of 50 agents. Agents are grouped according to their numbers in fives, i.e. 1-5 in group 1, 6-10 in group 2 and so on.

with 50 agents instead of 100. From Figure 5.19(a), decreasing the interval T has the simple effect of shortening the time to completion of the simulation but with diminishing returns. In Figure 5.19(b), it is also observed that reducing the interval T reduces the variance on the order in which agents converge to their targets. This means that the movement of agents with smaller interval T is more orderly and follows the priority order more closely. This is expected as the smaller interval T approximates the case where agents get updates from every other agent at each time step.

5.7 Summary

This chapter presented a low complexity distributed multi-agent traffic coordination system using a hybrid CA algorithm that is able to operate under limited communication while still guaranteeing non-collision. The proposed method adapts the VO methodology from literature to a trigger-based communication system where updates on other agents' information are only obtained one agent at a time and infrequently. The Voronoi cell method was adapted to enable head-on velocities. A priority system for distributed coordination and decision-making was also introduced, which was shown to prevent deadlocks in analysis and simulation.

FIGURE 5.19: Performance comparison for varying control time interval T . a) Comparison of the sum of distance remaining to respective targets for 50 agents as a fraction of the sum of start to target distances. b) Comparison of individual completion times.

The hybrid algorithm with the communication protocol was demonstrated in two key scenarios. The antipodal swap of 100 agents demonstrated algorithm effectiveness for a large number of agents and the effect of the priority system on agent movement. The 100 randomized simulations of 50 agents demonstrated robustness in a relatively tight congested area.

A comparison was also made to a version of the hybrid algorithm where priority was removed. A specific simulation was provided which showed how the CCP overlap deadlock can be easily encountered. The same simulation with the priority system re-enabled did not encounter a deadlock.

The comparison with and without repulsion showed that a first-in-first-out resolution can be achieved in the 50 agent antipodal swap scenario using the combination of repulsion and VO-based partitions. However, this came at the cost of slowing other higher-numbered agents, which are forced to wait if they are unable to go around the congestion.

One issue that can be the topic of future work is that agents are currently initialized with the position information of every other agent in $\mathcal{N}$. This limits the scalability

of the swarm and agents cannot be added online. A direction for solving this issue would be to make the interactions local to some distance range from the agent. Perhaps a separate communication layer at a different frequency can be used where agents broadcast their identity number for a fixed range. Agents nearby then use this information to determine whom to communicate with as described in this chapter.

Another issue is that currently target positions are assumed to be fixed and non-moving. However, this may be a consideration for a higher level planner that decides where the agents of the swarm go.

A third issue is that only the 2D case has been considered in this work. The 3D case may be more useful especially for UAVs. The current deadlock avoidance method is insufficient for the 3D case. The 2D-based formulation such as with the velocity rotation in (5.41) needs to be updated. However, the congestion in the 3D case may be easier to resolve as conflicts may also be resolved by altitude separation.

The fourth issue is that the agents are currently modeled as single integrators. This problem is also the subject of the next chapter. The agents in this work may be used as virtual ones to be tracked by the actual UAVs. The safe separation distance, $d_{\min}$, can be enlarged to account for UAV dynamics. An algorithm can then be designed to ensure that UAVs remain within a fixed displacement from the virtual agent while tracking its movement. As the motion in the hybrid algorithm is such that zero velocity is always a feasible solution, perfect velocity tracking is not necessary. Further details are provided in the next chapter.

Chapter 6

Jerk-limited Holonomic Robot Motion Planning Using Barrier Functions

This chapter describes an algorithm that may be used to adapt single-integrator based higher level motion planning or collision avoidance (CA) algorithm such as the one described in Chapter 5 for higher order holonomic agents such as quadrotor UAVs. Barrier functions are used to restrict a robot’s motion to within a maximum displacement from a nominal position with consideration for constraints on velocity, acceleration and jerk. This work has been submitted to a conference at the time of writing this thesis.

The purpose of this chapter is to provide a means by which the work in Chapter 5 can be adapted to quadrotor UAVs without compromising its congestion and deadlock resolution capability or requiring extensive modification. This chapter also extends the barrier-function-based CA method described in [77] and (2.20) to a higher order. The extended method is used to “avoid collisions” with the boundary at maximum allowed displacement relative to the nominal position instead of other UAVs.

This chapter is organized as follows. The motivation for this work is provided in the following section. Section 6.2 defines the model, nominal path, and objective. The barrier function discretization is described in Section 6.3. The main results are presented in Section 6.4. A simulation example is provided in Section 6.5. Finally, some concluding remarks are drawn in Section 6.6.

6.1 Motivation

Coordination of multi-robot systems that operate under heavy congestion is relatively easier to solve when robots are modeled as single integrators such as in Chapter 5. Consideration of higher order models such as in [67] can easily result in infeasible scenarios. The use of model predictive methods instead requires extensive communication and iteration to obtain a suitable solution, which may not be practical [89]. A hierarchical system as suggested in [9] is instead used. In this work, algorithms such as the one in Chapter 5 are treated as higher level planners operating on a virtual agent. The algorithm implementation is shown in Figure 6.1. The virtual agent moves as a single integrator with a minimum required separation distance from other agents and is allowed to change directions instantaneously. The lower level algorithm is then tasked with tracking the agent while maintaining a maximum lateral displacement smaller than the separation distance and subjected to dynamic constraints. This is achieved by applying input constraints to a tracking controller using barrier functions.

A similar idea is found in [68] where a robot's chosen reference trajectory is buffered by a cylinder with radius equal to the maximum possible tracking error. However, this method results in an excessive buffer as lateral error reduction is not prioritized. Moreover, the reference trajectory begins from the robot's current position. The actual trajectory may still intersect with an obstacle if it begins on the boundary of an obstacle, which is mitigated by applying a repulsive force.

Jerk-limited trajectory generation within safe corridors defined between waypoints such as in [106] is another potential solution. However, the polynomial-based trajectory generation may not allow sudden changes of waypoints midway between the current ones. The alternative method presented here is also computationally simpler as dynamic constraints are instead encoded in the barrier functions, which then provide jerk input constraints.

With regards to limited acceleration, the online scaling method was developed to track a given trajectory with minimal lateral error [107, 108]. However, a predefined trajectory is required, which is reparameterized to account for constraints.

The computationally simple barrier function method is chosen here to account for dynamic constraints. In [77], the barrier function method is used to obtain an upper bound on acceleration inputs based on current relative position and velocity that prevents a collision between a pair of robots with limited acceleration and velocity. This is extended here to find the upper bound on jerk based on current displacement, velocity and acceleration.

In this work, single-integrator path planning and collision avoidance methods are adapted to jerk-limited holonomic robots by using an online discrete-time modification of the barrier function approach. The method is referred to as the Jerk-limited Motion Restriction (JMR) algorithm and ensures that the robot does not deviate by more than a predefined lateral distance from a nominal path provided by the higher level path planner. The algorithm bounds a tracking controller input to explicitly ensure that dynamic constraints are met. Additionally, a virtual agent is used to represent the nominal path whose velocity is modified by the higher level planner. Its nominal position is advanced by the JMR algorithm and fed back to that planner. The virtual agent behaves as a single integrator and is allowed to change directions or stop instantaneously.

6.2 Problem Formulation

6.2.1 System

The robot is modeled as a 3rd order integrator with r holonomic degrees of freedom,

$$\ddot{q}_i = u_i, \quad q_i \in \mathbb{R}, \quad i \in \{1, 2, \dots, r\}, \quad (6.1)$$

where q_i , $v_i = \dot{q}_i$, $a_i = \ddot{q}_i$ and u_i denote the scalar position, velocity, acceleration and jerk input respectively for axis i . Additionally, constraints are imposed: $|v_i| \leq v_{m,i}$, $|a_i| \leq a_{m,i}$, and $|u_i| \leq u_{m,i}$. $v_{m,i}$, $a_{m,i}$ and $u_{m,i}$ are positive values representing maximum absolute velocity, acceleration and jerk respectively for each axis i .

This model may be used to approximate robot systems with limited acceleration and jerk such as quadrotor robots [79, 106]. Quadrotors may not be able to produce a required lateral acceleration immediately due to drag effects and having to rotate on its roll or pitch axis to produce lateral thrust [21]. The finite rotation speed is modeled by constrained jerk in (6.1). This is sufficient for approximating quadrotors and is shown in simulation.

6.2.2 Nominal Path Requirement

The objective is to constrain a robot's motion relative to some nominal path dictated by a higher level controller or planner.

The nominal path is defined as the pair $\sigma_n, \gamma_n \in \mathbb{R}^r$ and referred to as the *nominal position and velocity* respectively. σ_n represents a virtual position that can only be moved in the direction dictated by γ_n . The maximum allowed change is $\gamma_n T_n$, where T_n is the sampling time interval of the higher level controller. The robot is required to track the path from σ_n to $\sigma_n + \gamma_n T_n$ without deviating by more than a predefined lateral distance.

σ_n is updated by the JMR algorithm prior to calculating the constrained control input at its own time interval T . It is then sampled by the higher level controller to be used as the virtual position. This update is described in Section 6.4.3.

The nominal velocity, γ_n , is the command provided by the higher level controller given the latest sampled σ_n . γ_n is not constrained by the requirements of the robot dynamics and is allowed to change direction and magnitude instantaneously like with a single integrator. However, the actual robot velocity is not required to match the magnitude of γ_n at all times. γ_n is only used as a reference for speed and to dictate the required direction.

6.2.3 Problem

The main objective is to design a motion restriction algorithm that restricts the displacement of a robot of dynamics (6.1) relative to the nominal path from σ_n to $\sigma_n + \gamma_n T_n$ at all times while subjected to its constraints on velocity, acceleration and jerk, i.e.

$$|q_i - \sigma_{n,i}| \leq D_{m,i}, \quad (6.2)$$

where $\sigma_{n,i}$ is the nominal position on the i th axis and $D_{m,i}$ is the maximum permissible distance. The algorithm is also required to provide an update on the virtual position, σ_n , so that the robot may progress along γ_n .

6.3 Discretized CBF Preliminary

The zeroing control barrier function (ZCBF) described in [77, 78] and reviewed in Section 2.2.2.3 is modified here for discrete-time use in a manner similar to [109].

The continuous formulation described in Section 2.2.2.3 may not be directly applicable to discrete-time systems. Some systems may operate with slow high level control frequencies (e.g. 5-20Hz) such as with computationally complex path planners or CA methods, use sensors with low sampling rates such as GPS or cameras,

or have low communication rates such as with UAV swarms [14]. The input obtained from current states may not be sufficient to ensure set invariance until the next state sampling time.

A simple discrete-time modification of the ZCBF method suitable for the application in this work is proposed instead. The predicted state at the next sampling step is used here like in [109]. Doing so in this application maximizes the permissible control space while ensuring set invariance at the next step.

Consider that forward invariance of set $\mathcal{C}$ from (2.16) may only be needed at the sampling time. Also, it is possible to select $\kappa(h(\mathbf{x})) = h(\mathbf{x})$ for (2.17). If there exists a ZCBF candidate, $h(\mathbf{x})$, such that the set $S(\mathbf{x})$ is not empty for $\mathbf{x} \in \mathcal{C}$, then inputs $\mathbf{u} \in U$ can be solved for such that

$$h(\mathbf{x}(t+T)) = h\left(\mathbf{x}(t) + \int_t^{t+T} (f(\mathbf{x}) + g(\mathbf{x})\mathbf{u}) dt\right) \geq 0, \quad (6.3)$$

where t and T are the current sampling time and interval respectively. (6.3) is referred to here as the Discrete Control Barrier Function (DCBF). The challenge remains in formulating a suitable candidate function $h(x)$ and obtaining the solution set of inputs $\mathbf{u} \in U$.

Depending on the application, it may be difficult to ensure that $h(\mathbf{x}) \geq 0$ between sampling times. In this work, this is done by setting buffers near the set boundary in the formulation of $h(\mathbf{x})$.

6.4 Jerk-limited Motion Restriction Algorithm

Typically, CA algorithms for holonomic robots, such as in Section 5 and [77], specify a minimum safe distance between agents which guarantees non-collision. This translates to a sphere whereby the geometric centers of other agents are not allowed to enter. In the Jerk-limited Motion Restriction (JMR) algorithm, a subset

FIGURE 6.1: JMR Algorithm Outline.

safe space is defined within the sphere,

$$\mathcal{C} = \{\mathbf{p}, \mathbf{v}, \mathbf{a} \in \mathbb{R}^r \mid h(\mathbf{p}, \mathbf{v}, \mathbf{a}) \geq 0\}, \quad (6.4)$$

where $\mathbf{p}$ is the displacement from the sphere center, $\mathbf{v}$ is velocity, $\mathbf{a}$ is acceleration, r is the number of holonomic axes and $h(\cdot)$ is the DCBF to be defined. Set $\mathcal{C}$ is used to account for the dynamic constraints defined in Section 6.2.

In the proposed method, a virtual point agent is positioned at the center of the sphere, which is also the nominal position σ_n . The robot, modeled by (6.1), is initialized in set $\mathcal{C}$ and is required to stay within while tracking the nominal velocity, γ_n , from the higher level controller. The robot's progress is then used to update its nominal position along the nominal velocity without exiting $\mathcal{C}$. The algorithm is summarized in Figure 6.1.

6.4.1 DCBF for Displacement

A separate DCBF is defined for each of positive and negative directions for every axis. The axes are decoupled to avoid excessive centripetal acceleration due to any non-zero tangential motion relative to σ_n . The inequality constraints on jerk input, u_i , obtained from each of the positive and negative directions for each axis forms the upper and lower bounds respectively. These constraints are then used to bound the control input from a tracking controller.

Formally, the safe states set for axis i and direction j (positive or negative) is

$$\mathcal{C}_{ij} = \{p_{ij}, v_{ij}, a_{ij} \in \mathbb{R} \mid h_{ij}(p_{ij}, v_{ij}, a_{ij}) \geq 0\}, \quad (6.5)$$

where $p_{i1} = q_i - \sigma_{n,i}$ and $p_{i2} = -p_{i1}$, $v_{i2} = -v_{i1} = -v_i$, $a_{i2} = -a_{i1} = -a_i$, and $j = 1$ and 2 are positive and negative directions respectively.

h_{ij} measures the excess distance between the robot's current displacement, p_{ij} , and the permissible displacement, $D_{m,i}$, from σ_n after subtracting a buffer distance. The buffer distance is the required distance to decelerate to zero velocity and negative acceleration given v_{ij} and a_{ij} .

$$h_{ij}(t) = D_{m,i} - p_{ij}(t) - \Delta p_{ij}^{v\wedge}(t) - \Delta p_{ij}^{a\wedge}(t), \quad (6.6)$$

where $\Delta p_{ij}^{v\wedge}(t)$ and $\Delta p_{ij}^{a\wedge}(t)$ are the buffers to be defined.

The required deceleration, given the dynamic constraints, is a two-step process. The robot is first required to adjust its acceleration to the maximum deceleration, $-a_{m,i}$, at maximum negative jerk, $-u_{m,i}$. The first buffer is formulated as

$$\Delta p_{ij}^a(t) = v_{ij}(t)t_a + \frac{1}{2}a_{ij}(t)t_a^2 + \frac{1}{6}(-u_{m,i})t_a^3, \quad (6.7)$$

where

$$t_a = \frac{a_{m,i} + a_{ij}(t)}{u_{m,i}} \quad (6.8)$$

is the time needed to reach $-a_{m,i}$. Note that as a buffer, the upper bound on the distance traveled is required. This is approximated by

$$\Delta p_{ij}^{a\wedge}(t) = \max(v_{ij}(t)t_b, 0) + \max\left(\frac{1}{2}a_{ij}(t)t_b^2, 0\right), \quad (6.9)$$

where

$$t_b = \max(t_a, T), \quad (6.10)$$

and T is the the sampling time interval. Alternatively, the true upper bound may

be used without affecting the result but not described here for simplicity. If the velocity,

$$v_{ij}^a(t) = v_{ij}(t) + a_{ij}(t)t_a + \frac{1}{2}(-u_{m,i})t_a^2, \quad (6.11)$$

is non-positive, then no further deceleration is required. Otherwise, in the second step, the robot should maintain the maximum deceleration until its velocity is zero. The distance required is

$$\Delta p_{ij}^{v\wedge}(t) = -\frac{[\max(v_{ij}^a(t), 0)]^2}{2(-a_{m,i})}, \quad (6.12)$$

from standard kinematic equations.

The maximum permissible displacement, $D_{m,i}$, is determined based on (6.9), (6.12), and the maximum speed $v_{m,i}$.

$$D_{m,i} \geq \Delta p_{ij}^{v\wedge}(t_0) + \Delta p_{ij}^{a\wedge}(t_0) + v_{m,i}T, \quad (6.13)$$

where velocity, acceleration and displacement are set as $v_{ij}(t_0) = v_{m,i}$ and $a_{ij}(t_0) = 0$ at $p_{ij}(t_0) = 0$. This setting allows the robot to move at the maximum speed while at the nominal position. Then if $p_{ij}(t) > v_{m,i}T$, deceleration occurs. The third term is used to ensure that $v_{m,i}$ is reachable at $p_{ij}(t) = 0$ due to the use of (6.3) to obtain input bounds.

For certain combinations of $u_{m,i}$, $a_{m,i}$ and $v_{m,i}$, the distance from (6.13) may be overly conservative. Starting from the same conditions, the robot's velocity may reach zero before the maximum deceleration, $-a_{m,i}$, is reached. From (6.8) and (6.11), the combination is such that $u_{m,i} < \frac{(a_{m,i})^2}{2v_{m,i}}$. However, the consideration of this case is not necessary for the objective of this work and is left to future work.

From (6.3), the upper bound on inputs u_{ij} until the next sampling time is

$$u_{ij}^\wedge = \sup \{u_{ij}(t) \mid h_{ij}(t+T) \geq 0, \|u_{ij}(t)\| \leq u_{m,i} \text{ and } \|a_{ij}(t+T)\| \leq a_{m,i}\}, \quad (6.14)$$

where $p_{ij}(t+T)$, $v_{ij}(t+T)$, $a_{ij}(t+T)$ are obtained by integrating (6.1), and T is the sampling time interval. $u_{ij}^\wedge$ may be obtained by checking discrete increments of $u_{ij} \in [-u_{m,i}, u_{m,i}]$. The search increment size does not affect the barrier's effectiveness, but it may result in a conservative $u_{ij}^\wedge$.

The resulting bounds on input u_i is thus

$$-u_{i2}^\wedge \leq u_i \leq u_{i1}^\wedge. \quad (6.15)$$

The result as per the objective in Section 6.2.3 is summarized in the following proposition.

Proposition 6.1. *Consider a robot with dynamics (6.1) initialized at $t = 0$ such that $h_{ij}(t = 0) \geq 0$ for an axis i and a direction j . If the robot's control input u_{ij} in that axis direction is perfectly tracked and also constrained by the upper bound (6.14), then its displacement, $p_{ij}(t) \leq D_{m,i}$ for all $t > 0$ including between sampling times of interval T .*

Proof. Inputs are chosen such that $h_{ij}(t+T) \geq 0$ in (6.14). Moreover, the buffer (6.9) denotes the distance needed to reduce $a_{ij}(t)$ to $-a_{m,i}$ using the lowest possible $u_{ij} = -u_{m,i}$. (6.9) also reduces at a faster rate than the actual required distance when $u_{ij} < 0$. Hence, $-u_{m,i}$ is always a feasible input while $a_{ij}(t) > -a_{m,i}$. Subsequently, $u_{ij} = 0$ maintains the maximum deceleration for the buffer (6.12). Thus $p_{ij}(kT) \leq D_{m,i}$, $\forall k \in \mathbb{Z}^+$.

$p_{ij}(t) \leq D_{m,i}$ is also true between sampling times. For $p_{ij}(t)$ near $D_{m,i}$, if $a_{ij}(t) = -a_{m,i}$, then the required buffer distance is accounted for by (6.12). Otherwise, if $a_{ij}(t) > -a_{m,i}$, the buffer (6.9) accounts for any forward movement in the time interval. (6.9) is the upper bound value, i.e. the maximum forward distance the robot moves while $a_{ij}(t)$ decreases to $-a_{m,i}$. Since t_b is lower bounded by interval T in (6.10), then any distance due to forward acceleration or velocity is accounted for in that interval. $\square$

6.4.2 Additional DCBF for Velocity

A DCBF for velocity is also needed for a smooth approach to the maximum $v_{m,i}$ under jerk constraints. The barrier function $h_{ij}(t)$ does not limit the maximum velocity. v_{ij} may exceed $v_{m,i}$ for $p_{ij} < 0$.

Hence, define in a similar manner

$$h_{ij}^v(t) = v_{m,i} - v_{ij}(t) - \Delta v_{ij}^{a^\wedge}(t) - a_{ij}(t)T, \quad (6.16)$$

where

$$\Delta v_{ij}^{a^\wedge}(t) = -\frac{[\max(a_{ij}(t), 0)]^2}{2(-u_{m,i})}. \quad (6.17)$$

(6.17) is the same as (6.12) but applied to velocity. Similar to (6.9), the last term of (6.16) accounts for any forward velocity change when $u_{ij} < 0$. This is needed when a_{ij} crosses zero during the interval. Hence, the same conclusions from Proposition 1 can be drawn but applied to velocity instead.

Similar to (6.14), the upper bound of input is

$$u_{ij}^{v^\wedge} = \sup \left\{ u_{ij}(t) \left| \begin{array}{l} h_{ij}^v(t+T) \geq 0, \\ \|u_{ij}(t)\| \leq u_{m,i}, \\ \|a_{ij}(t+T)\| \leq a_{m,i} \end{array} \right. \right\}. \quad (6.18)$$

Input u_i is then further constrained by

$$-u_{i2}^{v^\wedge} \leq u_i \leq u_{i1}^{v^\wedge}. \quad (6.19)$$

6.4.3 Nominal Position Update

The update of the nominal position, σ_n , occurs before $u_{ij}^{a^\wedge}$ and $u_{ij}^{v^\wedge}$ are calculated in the same sampling time. The nominal position is advanced in the direction γ_n based on the actual position q_i without violating the displacement DCBF (6.6).

The minimum of three possible shift distances in the direction γ_n are calculated and used to move σ_n . The first is

$$d_{\max} = \gamma_n T_n - (\sigma_n^- - \sigma_n(t_{\text{last}})), \quad (6.20)$$

where σ_n^- is the latest nominal position before this update and t_{last} is the time when the last γ_n update by the higher level planner was first sampled by the JMR algorithm. The second is the positive projection of the robot displacement relative to σ_n^- along γ_n , i.e.

$$d_p = \max \left(p^T \frac{\gamma_n}{\|\gamma_n\|}, 0 \right). \quad (6.21)$$

The last is based on the values of (6.6) opposite the direction of γ_n for each axis i . Shifting σ_n is equivalent to shifting p_i in the opposite direction.

$$d_h = \min \left\{ h_{i3}(p_{i3}(t), v_{i3}(t), a_{i3}(t)) \frac{\|\gamma_n\|}{|\gamma_{n,i}|} \right\}, \quad i \in \{1, 2, \dots, r\}, \quad (6.22)$$

where subscript $i3$ denotes the i th axis and opposite the direction of γ_n , and $\gamma_{n,i}$ is the nominal velocity along the i th axis. d_h scales $\frac{\gamma_n}{\|\gamma_n\|}$ up to the maximum allowed distance such that $h_{ij}(t) \geq 0$ is still satisfied for all axes.

The updated nominal position is thus

$$\sigma_n = \sigma_n^- + \frac{\gamma_n}{\|\gamma_n\|} \min(d_{\max}, d_p, d_h), \quad (6.23)$$

which is then used to determine jerk input bounds as described in Sections 6.4.1 and 6.4.2 at the same sampling time.

6.5 Simulation

Demonstration of the JMR algorithm is done by simulation of a quadrotor robot following predefined waypoints. The MATLAB Simulink quadrotor model obtained from [110] is used, which includes rotor drag effects. In the JMR algorithm, the

quadrotor robot is modeled by (6.1) with $r = 3$ degrees of freedom for 3D space. To simplify simulation, the higher level controller providing γ_n operates synchronously with the JMR algorithm with time interval $T_n = T = 0.05s$.

The waypoints are $[0 \ 0 \ 3]$, $[3 \ 0 \ 7]$, $[-3 \ 3 \ 3]$, $[-2 \ -3 \ 5]$, $[3 \ 3 \ 3]$ and shown in Figure 6.2(a). The nominal velocity is generated according to Section 6.2.2 using a proportional controller. The target waypoint is switched to the next when σ_n is less than some arbitrarily small distance from the current position.

To track the nominal path, the reference acceleration used is

$$\mathbf{a}^- = k_p [\sigma_n + \gamma_n T - \mathbf{q}(t)] + k_d [\gamma_n - \mathbf{v}(t)], \quad (6.24)$$

where $k_p, k_d \geq 0$ are control gains, and $q(t)$ and $v(t)$ are the current actual quadrotor position and velocity vectors.

The constraints are applied separately for each axis i :

$$u_i^p = \max \left[\min \left(\frac{1}{T} (a_i^- - a_i(t)), u_{i1}^\wedge \right), -u_{i2}^\wedge \right], \quad (6.25)$$

$$u_i = \max [\min (u_i^p, u_{i1}^{v\wedge}), -u_{i2}^{v\wedge}]. \quad (6.26)$$

The bounded u_i is then integrated to obtain the required velocity at the next time step. This is then tracked by a cascaded PID controller such as the one provided in the Simulink model from [110]. Slight tracking errors can still occur. For cases where $u_{ij}^\wedge$ or $u_{ij}^{v\wedge}$ do not exist, it is set to the extreme $-u_{m,i}$ with $\|a_{ij}(t + T)\| \leq a_{m,i}$.

TABLE 6.1: Simulation Parameters

kp	0.2	kd	0.2	$v_{m,i}$	1.5ms^{-1}
$a_{m,i}$	1.1ms^{-2}	$u_{m,i}$	3.0ms^{-3}	$D_{m,i}$	1.4m

The simulation was conducted using parameters in Table 6.1. The same dynamic constraints are applied to all three axes. The results are presented in Figure 6.2 and further details are in Figure 6.3, 6.4 and 6.5. The robot's initial actual and

FIGURE 6.2: a) X-Y plane view of waypoints and quadrotor motion. Crosses denote waypoints with nominal paths traced by blue dotted lines. The quadrotor's positions are traced by the solid orange line. b) Displacement with respect to nominal position along the three axes. The blue dotted line denotes the maximum and minimum permissible displacements.

nominal positions are initialized at the first waypoint. The robot's nominal position reached the final waypoint at about 25.5s. From Figure 6.3(a), 6.4(a) and 6.5(a), the robot's actual position eventually converges to the nominal at the final waypoint where $\|\gamma_n\| = 0$. Figure 6.2(b) shows that the robot's maximum displacement is mostly maintained between -1.4m and 1.4m.

Although not visible, the robot's z-axis displacement does temporarily exceed by at most 0.1m at $t=5.7$ s. This is due to accumulated tracking error and coupling between axes in the quadrotor drag dynamics. However, this error is small and may be compensated with additional buffering or better tracking methodologies. Figure 6.3(b), 6.4(b) and 6.5(b), show that the quadrotor closely tracked the reference velocity.

In Figure 6.3(c), 6.4(c) and 6.5(c), the effect of the JMR algorithm can be observed. Firstly, in cases where the tracking controller acceleration from (6.24) is too large such as when the robot is approaching the nominal position or its velocity is approaching maximum, then the acceleration is reduced. Secondly, if the controller

deceleration is too low such as when its absolute displacement is approaching $D_{m,i}$, then it is increased. The former is observed where the magnitude of the acceleration is lower than the controller's and the latter where there are sharp increases. Note that such changes occur up to the maximum jerk rate, $u_{m,i}$.

6.6 Summary

The JMR algorithm ensures a robot with dynamics (6.1) does not exceed a maximum permissible displacement from a nominal position while subjected to dynamic constraints including jerk. The control barrier certificate method was adapted for discrete application by considering the barrier function's value at the next time step. Moreover, the barrier function was formulated such that the displacement bound is not exceeded between time intervals as long as inputs are perfectly tracked by the robot. The method for advancing the nominal position, which is used as a

FIGURE 6.3: X-axis position, velocity and acceleration compared to reference.

FIGURE 6.4: Y-axis position, velocity and acceleration compared to reference.

FIGURE 6.5: Z-axis position, velocity and acceleration compared to reference.

high level controller's virtual agent, was also presented. An implementation example was given by simulation. The modeled quadrotor was able to track the required velocity with minor errors.

Future work may include compensating for tracking error and reducing the conservativeness of the presented formulation. The former is due to the assumption that the quadrotor UAV is able to track the velocity perfectly. It was noted that there still exist some error in the tracking and the quadrotor exceeded the maximum displacement by an insignificant distance. This may be addressed by considering the current velocity error in calculating the next point in the trajectory or by using a more robust tracking controller.

The issue of conservativeness refers to the buffer formulation. For the case where $u_{m,i} < \frac{(a_{m,i})^2}{2v_{m,i}}$, it is not necessary for the acceleration to reach $-a_{m,i}$ in order to reduce velocity to zero. Hence, the resulting buffer is larger than needed. Furthermore, the maximum forward movement due to the current velocity and acceleration are only approximated by (6.9).

Another possible future work would include comparison of computation times with other solutions such as in [106].

Chapter 7

Conclusion and Recommendations

7.1 Conclusion

This thesis has presented several contributions to UAV swarm collision avoidance and distributed coordination research. Two scenarios for UAV swarms were considered. The first involved multi-intruder interception using UAVs. The second involved general distributed coordination of UAVs with a focus on limited bandwidth in communication and deadlock avoidance. Examples of this scenario's application include delivery and construction. The second scenario was motivated by the issues of the first. Regardless, all works presented here involve the problem of collision avoidance of multiple UAVs or agents, which are solved with novel methods.

The first scenario was covered in Chapter 3 where a novel LOS-based repulsion method was used to avoid collisions. The repulsion is integrated into the guidance law using Lyapunov functions. The use of repulsion on relative LOS prevents collisions for the double integrator model as well as ensures that the LOS towards its assigned target remains clear. Simulations showed that the agents were able to intercept their assigned targets in finite time while avoiding collisions. However, the potential field used is unbounded at the minimum allowed relative LOS. This resulted in unrealistically high control inputs in some cases as shown with the

semi-physical experiment. Moreover, the experiment was limited by heavy wireless communication requirement.

Subsequently, the work in Chapter 4 represents an attempt to tackle the aforementioned issues of limited available acceleration and communication bandwidth. The VO CA method was adapted instead of repulsion. The resulting partitions in velocity space between a pair of agents were shown to guarantee non-collision if maintained indefinitely. Updates on these partitions are obtained by event-triggered communication to enable agents to reach their target destination. This greatly reduced the communication required. Moreover, since the velocity from the previous control step is always a feasible solution and the solution space is convex, the method can be used for agents with finite acceleration. However, the lack of head-on velocities limited the number of agents that can operate in the swarm without causing deadlock.

The focus in Chapter 5 was to build on the ideas presented in Chapter 4 and provide a means to avoid or resolve deadlocks distributedly. The Voronoi Cell and repulsion methods were integrated to overcome the limitations of the previous method in a hybrid collision avoidance algorithm. Partitions in this work are formulated in terms of position instead of velocity space. The Voronoi-based partition was used to allow head-on velocities. To resolve deadlocks and congestion in a distributed manner, a priority system was introduced which took advantage of the communication system to determine if an agent can pass the other and the direction to do so. Like before, communications are event-triggered. The algorithm's effectiveness was shown in both analysis and simulations. Various simulation scenarios were provided which showed all agents are able to reach their assigned target positions without collision or deadlock in finite time. Simulations involved up to 100 agents at a time in heavy congestion scenarios. However, the method is only applicable to the single integrator agent. This issue is addressed in the Chapter 6.

Finally, Chapter 6 presented a novel method to adapt higher level planning algorithms for single integrator agents to the constrained triple integrator model,

which is a better approximation of a quadrotor UAV. This is achieved by treating the agent in the higher level planner as a virtual agent to be tracked by the actual UAV. The UAV is then kept within some maximum displacement from the virtual agent using discretized barrier functions. The barrier functions are formulated to limit the UAV's jerk at any time based on its current displacement, velocity and acceleration while respecting the respective constraints. The UAV's relative position is then used to update the virtual's along the higher level algorithm's command velocity direction. Simulations showed that the UAV is able to track the virtual agent with minimal error. This method allows relatively simpler higher level motion planners such as that in Chapter 5 to be used with quadrotor UAVs with minimal modification but at the cost of setting larger minimum separation distances between agents. The advantage of this method over simply setting an estimated buffer distance is that the required separation distance is fixed and known. Moreover, the separation distance can be set to be smaller than if an unconstrained tracking controller is used.

7.2 Recommendations

The work presented in this thesis is by no means complete. Recommendations for extending the work presented here are provided in the following. It should be noted that some of these recommendations were not considered in the thesis as presented due to scope limiting and resource availability.

For the multi-intruder interception work, one main recommendation to improve on reducing disruption to interception due to collision avoidance is to partition space such that both the intruder's and interceptor's movement is isolated from every other agent's and their assigned intruders. Currently, only the LOS is isolated by repulsion. However, this is not a trivial problem as such partitions may not exist when there are many intruders. It might also require centralized coordination and assignment.

One of the directions not pursued is in adapting the partition-based collision avoidance methods presented in this thesis to 3D. The collision cones of Chapter 4 and 5 will result in infinite possible avoidance directions in the 3D case. One way of simplifying this may be to limit it to 4 directions by approximating the cone with a 4-sided pyramid like in [68]. For Chapter 5, a significantly different coordination process is needed. Another simpler possibility might be to separate agents by altitude, while using the same algorithms. Agents may then be allowed to switch altitudes given appropriate conditions if the current altitude is heavily congested.

Another possible research direction would be to explore other types of distributed coordination under trigger-based communication. These other types include, but are not limited to, coordinated avoidance of environmental obstacles, navigation through narrow passages, formation flight, and queuing where multiple agents share the same goal position.

Lastly, the works in Chapter 4 and 5 assume that all agents can be reached by point-to-point communication using a mobile ad hoc network. The current formulation is such that every agent has a partition for every other agent in the swarm. This makes it difficult to scale online, i.e. to add new agent to the swarm while the rest are already operating. Moreover, in the case of Chapter 4, the combination of no head-on velocity and large numbers of agents easily leads to deadlock. This may be overcome by investigating the use of a second communication layer that uses short-range broadcasts of small messages carrying the identity of the broadcaster. The small message packet with short-range reduces the likelihood of flooding the immediate area which would reduce likelihood of communication drop outs. This may be used to detect which agents are nearby and therefore require communication with to establish CA partitions. This allows the established algorithms to be used in a local manner.

List of Author's Publications

Journal Articles

- **Abdul Hanif Zaini** and Lihua Xie, “Distributed drone traffic coordination using triggered communication,” *Unmanned Systems*, 08(01): 1-20, 2020. [\[104\]](#)
- Bing Zhu, **Abdul Hanif Zaini** and Lihua Xie, “Distributed guidance for interception by using multiple rotary-wing unmanned aerial vehicles,” *IEEE Transactions on Industrial Electronics*, 64(7): 5648-5656, 2017. [\[24\]](#)
- Kexin Guo, Zhirong Qiu, Cunxiao Miao, **Abdul Hanif Zaini**, Chun-Lin Chen, Wei Meng and Lihua Xie, “Ultra-wideband-based localization for quad-copter navigation,” *Unmanned Systems*, 04(01):23–34, 2016.

Conference Proceedings

- **Abdul Hanif Zaini** and Lihua Xie, “A Hybrid Collision Avoidance Algorithm for Multi-Agent Traffic Coordination Under Limited Communication,” *Proc. of the 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Singapore, Dec. 2018. [\[103\]](#)
- **Abdul Hanif Zaini** and Lihua Xie, “Triggered communication for reciprocal collision avoidance in multi-agent navigation,” *Proc. of the 2018 IEEE 14th International Conference on Control and Automation (ICCA)*, Anchorage, Alaska, June 2018. [\[100\]](#)

- Nguyen Thien Minh, **Abdul Hanif Zaini**, Chen Wang, Kexin Guo and Lihua Xie, “Robust Target-relative Localization with Ultra-Wideband Ranging and Communication,” *Proc. of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia, May 2018.
- Bing Zhu, **Abdul Hanif Zaini**, Lihua Xie and Guoan Bi, “Distributed guidance for interception by using multiple rotary-wing unmanned aerial vehicles,” in *Proc. of the 2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Banff, Canada, July 2016. [23]
- Bing Zhu, Juanjuan Xu **Abdul Hanif Zaini** and Lihua Xie, “A three-dimensional integrated guidance law for rotary UAV interception,” in *Proc. of the 2016 12th IEEE International Conference on Control and Automation (ICCA)*, Kathmandu, Nepal, June 2016. [22]
- Nguyen Thien Minh, **Abdul Hanif Zaini**, Kexin Guo and Lihua Xie, “An Ultra-Wideband-based Multi-UAV Localization System in GPS-denied Environments,” *Proc. of the 2016 International Micro Air Vehicle Conference and Competition*, Beijing, China, Oct. 2016.

Bibliography

[Citing pages are listed after each reference.]

- [1] Roland Bouffanais. *Design and control of swarm dynamics*. Springer, Singapore, 2016. [Page [1](#).]
- [2] Erol Şahin. Swarm robotics: From sources of inspiration to domains of application. In Erol Şahin and William M. Spears, editors, *Swarm Robotics*, pages 10–20. Springer Berlin Heidelberg, 2005. [Page [1](#).]
- [3] Levent Bayındır. A review of swarm robotics tasks. *Neurocomputing*, 172: 292–321, 2016. [Pages [2](#) and [4](#).]
- [4] Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, 2013. [Page [2](#).]
- [5] Veysel Gazi and Barış Fidan. Coordination and control of multi-agent dynamic systems: Models and approaches. In Erol Şahin, William M. Spears, and Alan F. T. Winfield, editors, *Swarm Robotics*, pages 71–102. Springer Berlin Heidelberg, 2007. [Page [2](#).]
- [6] Jan Carlo Barca and Y. Ahmet Sekercioglu. Swarm robotics reviewed. *Robotica*, 31(3):345–359, 2013. [Page [2](#).]
- [7] Samira Hayat, Evşen Yanmaz, and Raheeb Muzaffar. Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint.

- IEEE Communications Surveys & Tutorials*, 18(4):2624–2661, 2016. [Pages 2, 4, and 81.]
- [8] Jorge Cortés and Magnus Egerstedt. Coordinated control of multi-robot systems: A survey. *SICE Journal of Control, Measurement, and System Integration*, 10(6):495–503, 2017.
- [9] Soon-Jo Chung, Aditya Avinash Paranjape, Philip Dames, Shaojie Shen, and Vijay Kumar. A survey on aerial swarm robotics. *IEEE Transactions on Robotics*, 34(4):837–855, 2018. [Pages 2, 3, 4, 13, and 128.]
- [10] Martin Saska, Tomas Baca, Justin Thomas, Jan Chudoba, Libor Preucil, Tomas Krajník, Jan Faigl, Giuseppe Loianno, and Vijay Kumar. System for deployment of groups of unmanned micro aerial vehicles in gps-denied environments using onboard visual relative localization. *Autonomous Robots*, 41(4):919–944, 2017. [Page 3.]
- [11] Xiwang Dong, Bocheng Yu, Zongying Shi, and Yisheng Zhong. Time-varying formation control for unmanned aerial vehicles: Theories and applications. *IEEE Transactions on Control Systems Technology*, 23(1):340–348, 2014. [Page 3.]
- [12] Boldizsár Balázs and Gábor Vásárhelyi. Coordinated dense aerial traffic with self-driving drones. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6365–6372. IEEE, May 2018. [Pages 3, 4, 16, 19, and 33.]
- [13] Samira Hayat, Evşen Yanmaz, and Christian Bettstetter. Experimental analysis of multipoint-to-point uav communications with ieee 802.11 n and 802.11 ac. In *2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1991–1996. IEEE, 2015. [Page 3.]

- [14] Evşen Yanmaz, Markus Quaritsch, Saeed Yahyanejad, Bernhard Rinner, Hermann Hellwagner, and Christian Bettstetter. Communication and coordination for drone networks. In Yifeng Zhou and Thomas Kunz, editors, *Ad Hoc Networks*, pages 79–91. Springer International Publishing, 2017. [Pages 3 and 132.]
- [15] Bing Zhu, Lihua Xie, Duo Han, Xiangyu Meng, and Rodney Teo. A survey on recent progress in control of swarm systems. *Science China Information Sciences*, 60(7):070201, 2017. [Page 4.]
- [16] Xiangyu Meng, Lihua Xie, and Yeng Chai Soh. Asynchronous periodic event-triggered consensus for multi-agent systems. *Automatica*, 84:214–220, 2017. [Page 4.]
- [17] Xiwang Dong, Yan Zhou, Zhang Ren, and Yisheng Zhong. Time-varying formation control for unmanned aerial vehicles with switching interaction topologies. *Control Engineering Practice*, 46:26–36, 2016. [Page 4.]
- [18] Quan Yuan, Jingyuan Zhan, and Xiang Li. Outdoor flocking of quadcopter drones with decentralized model predictive control. *ISA transactions*, 71: 84–92, 2017. [Page 4.]
- [19] Jur Van den Berg, Ming Lin, and Dinesh Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *2008 IEEE International Conference on Robotics and Automation*, pages 1928–1935, May 2008. [Pages 5, 22, 59, and 64.]
- [20] Samir Bouabdallah, Pierpaolo Murrieri, and Roland Siegwart. Design and control of an indoor micro quadrotor. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 5, pages 4393–4398. IEEE, 2004. [Page 10.]
- [21] Robert E. Mahony, Vijay Kumar, and Peter Corke. Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robotics & Automation Magazine*, 19(3):20–32, Sep. 2012. [Pages 10, 11, and 130.]

- [22] Bing Zhu, Juanjuan Xu, Abdul Hanif Bin Zaini, and Lihua Xie. A three-dimensional integrated guidance law for rotary uav interception. In *2016 12th IEEE International Conference on Control and Automation (ICCA)*, pages 726–731. IEEE, June 2016. [Pages [10](#), [35](#), and [150](#).]
- [23] Bing Zhu, Abdul Hanif Bin Zaini, Lihua Xie, and Guoan Bi. Distributed guidance for interception by using multiple rotary-wing unmanned aerial vehicles. In *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 1034–1039, July 2016. [Page [150](#).]
- [24] Bing Zhu, Abdul Hanif Bin Zaini, and Lihua Xie. Distributed guidance for interception by using multiple rotary-wing unmanned aerial vehicles. *IEEE Transactions on Industrial Electronics*, 64(7):5648–5656, July 2017. [Pages [10](#), [35](#), and [149](#).]
- [25] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation*, pages 2520–2525, May 2011. [Pages [12](#), [28](#), [29](#), and [32](#).]
- [26] Alexander Alexopoulos, Amr Kandil, Piotr Orzechowski, and Essameddin Badreddin. A comparative study of collision avoidance techniques for unmanned aerial vehicles. In *2013 IEEE International Conference on Systems, Man, and Cybernetics*, pages 1969–1974. IEEE, 2013. [Page [13](#).]
- [27] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *Autonomous robot vehicles*, pages 396–404. Springer, 1986. [Pages [14](#), [15](#), and [18](#).]
- [28] Dušan M. Stipanović, Peter F. Hokayem, Mark W. Spong, and Dragoslav D. Šiljak. Cooperative avoidance control for multiagent systems. *Journal of Dynamic Systems, Measurement, and Control*, 129(5):699–707, 2007. [Pages [14](#), [15](#), [16](#), [44](#), and [53](#).]
- [29] George Leitmann and Janislaw Skowronski. Avoidance control. *Journal of optimization theory and applications*, 23(4):581–591, 1977. [Page [14](#).]

- [30] Dong Eui Chang, Shawn C. Shadden, Jerrold E. Marsden, and Reza Olfati-Saber. Collision avoidance for multiple agent systems. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, volume 1, pages 539–543, Dec 2003. [Pages 15 and 16.]
- [31] Hamed Rezaee and Farzaneh Abdollahi. A decentralized cooperative control scheme with obstacle avoidance for a team of mobile robots. *IEEE Transactions on Industrial Electronics*, 61(1):347–354, 2013. [Pages 15, 16, and 19.]
- [32] Reza Olfati-Saber. Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420, March 2006. [Pages 16, 17, 19, and 20.]
- [33] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, page 25–34, New York, NY, USA, 1987. Association for Computing Machinery. [Page 16.]
- [34] Zhiyong Chen, Ming-Can Fan, and Hai-Tao Zhang. How much control is enough for network connectivity preservation and collision avoidance? *IEEE Transactions on Cybernetics*, 45(8):1647–1656, Aug 2015. [Pages 16, 17, and 19.]
- [35] Qing Zhang, Ping Li, Zhengquan Yang, and Zengqiang Chen. Adaptive flocking of non-linear multi-agents systems with uncertain parameters. *IET Control Theory & Applications*, 9(3):351–357, Feb. 2015. [Page 17.]
- [36] Weiwei Zhao, Hairong Chu, Mingyue Zhang, Tingting Sun, and Lihong Guo. Flocking control of fixed-wing uavs with cooperative obstacle avoidance capability. *IEEE Access*, 7:17798–17808, 2019. [Page 17.]
- [37] Erick J. Rodríguez-Seda and Dušan M. Stipanović. Guaranteed collision avoidance with discrete observations and limited actuation. In *Advances in Intelligent Vehicles*, pages 89–110. Elsevier, 2014. [Pages 17, 33, and 56.]

- [38] Erick J. Rodríguez-Seda, Dušan M. Stipanović, and Mark W. Spong. Guaranteed collision avoidance for autonomous systems with acceleration constraints and sensing uncertainties. *Journal of Optimization Theory and Applications*, 168(3):1014–1038, 2016. [Pages 17 and 20.]
- [39] Erick J. Rodríguez-Seda. Self-triggered collision avoidance control for multi-vehicle systems. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 461–466, May 2015. [Pages 17, 20, 33, and 56.]
- [40] Khac Duc Do. Coordination control of multiple ellipsoidal agents with collision avoidance and limited sensing ranges. *Systems & control letters*, 61(1):247–257, 2012. [Page 17.]
- [41] Peter F. Hokayem, Dušan M. Stipanović, and Mark W. Spong. Coordination and collision avoidance for lagrangian systems with disturbances. *Applied Mathematics and Computation*, 217(3):1085–1094, 2010. [Page 17.]
- [42] Luciano C.A. Pimenta, Guilherme A.S. Pereira, Nathan Michael, Renato C. Mesquita, Mateus M. Bosque, Luiz Chaimowicz, and Vijay Kumar. Swarm coordination based on smoothed particle hydrodynamics technique. *IEEE Transactions on Robotics*, 29(2):383–399, 2013. [Page 18.]
- [43] Elon Rimon and Daniel E. Koditschek. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, Oct 1992. [Page 18.]
- [44] Dimos V. Dimarogonas. Sufficient conditions for decentralized potential functions based controllers using canonical vector fields. *IEEE Transactions on Automatic Control*, 57(10):2621–2626, 2012. [Page 18.]
- [45] Jiayi Sun, Jun Tang, and Songyang Lao. Collision avoidance for cooperative uavs with optimized artificial potential field algorithm. *IEEE Access*, 5:18382–18390, 2017. [Page 18.]

- [46] Thulasi Mylvaganam, Mario Sassano, and Alessandro Astolfi. A differential game approach to multi-agent collision avoidance. *IEEE Transactions on Automatic Control*, 62(8):4229–4235, 2017. [Page 18.]
- [47] Ema Falomir, Serge Chaumette, and Gilles Guerrini. A mobility model based on improved artificial potential fields for swarms of uavs. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8499–8504. IEEE, 2018. [Pages 18, 19, and 20.]
- [48] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. *Spatial tessellations: concepts and applications of Voronoi diagrams*, volume 501. John Wiley & Sons, 2009. [Pages 20 and 87.]
- [49] Santiago Garrido, Luis Moreno, and Dolores Blanco. Voronoi diagram and fast marching applied to path planning. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 3049–3054, May 2006. [Page 21.]
- [50] Priyadarshi Bhattacharya and Marina L. Gavrilova. Roadmap-based path planning-using the voronoi diagram for a clearance-based shortest path. *IEEE Robotics & Automation Magazine*, 15(2):58–66, 2008. [Page 21.]
- [51] Jorge Cortes, Sonia Martinez, Timur Karatas, and Francesco Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on robotics and Automation*, 20(2):243–255, 2004. [Page 21.]
- [52] Luciano C.A. Pimenta, Vijay Kumar, Renato C. Mesquita, and Guilherme A.S. Pereira. Sensing and coverage for a network of heterogeneous robots. In *2008 47th IEEE Conference on Decision and Control*, pages 3947–3952, Dec 2008. [Page 21.]
- [53] 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. [Pages 21, 27, 28, 31, 80, and 87.]

- [54] Jur Van Den Berg, Stephen J. Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. In *Robotics research*, pages 3–19. Springer, 2011. [Pages [21](#), [22](#), [23](#), [24](#), [27](#), [59](#), [60](#), [62](#), [64](#), [65](#), [83](#), and [86](#).]
- [55] Alyssa Pierson and Daniela Rus. Distributed target tracking in cluttered environments with guaranteed collision avoidance. In *2017 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pages 83–89. IEEE, Dec 2017. [Page [22](#).]
- [56] Paolo Fiorini and Zvi Shiller. Motion planning in dynamic environments using the relative velocity paradigm. In *[1993] Proceedings IEEE International Conference on Robotics and Automation*, pages 560–565 vol.1, May 1993. [Page [22](#).]
- [57] Paolo Fiorini and Zvi Shiller. Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, 17(7):760–772, 1998. [Pages [22](#) and [59](#).]
- [58] Animesh Chakravarthy and Debasish Ghose. Obstacle avoidance in a dynamic environment: A collision cone approach. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 28(5):562–574, 1998. [Page [22](#).]
- [59] Jung-Woo Park, Hyon-Dong Oh, and Min-Jea Tahk. Uav collision avoidance based on geometric approach. In *2008 SICE Annual Conference*, pages 2122–2126. IEEE, Aug 2008. [Page [24](#).]
- [60] Yu Fu, Xiang Yu, and Youmin Zhang. Sense and collision avoidance of unmanned aerial vehicles using markov decision process and flatness approach. In *2015 IEEE International Conference on Information and Automation*, pages 714–719, Aug 2015. [Page [24](#).]
- [61] James A. Douthwaite, Allan De Freitas, and Lyudmila S. Mihaylova. An interval approach to multiple unmanned aerial vehicle collision avoidance. In

- 2017 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, pages 1–8. IEEE, Oct 2017. [Page 24.]
- [62] Su-Cheol Han, Hyochoong Bang, and Chang-Sun Yoo. Proportional navigation-based collision avoidance for uavs. *International Journal of Control, Automation and Systems*, 7(4):553–565, 2009. [Page 24.]
- [63] Jamie Snape, Jur Van Den Berg, Stephen J Guy, and Dinesh Manocha. Smooth and collision-free navigation for multiple robots under differential-drive constraints. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4584–4589, Oct 2010. [Page 24.]
- [64] Javier Alonso-Mora, Andreas Breitenmoser, Paul Beardsley, and Roland Siegwart. Reciprocal collision avoidance for multiple car-like robots. In *2012 IEEE International Conference on Robotics and Automation*, pages 360–366, May 2012. [Page 24.]
- [65] Javier Alonso-Mora, Andreas Breitenmoser, Martin Rufli, Paul Beardsley, and Roland Siegwart. Optimal reciprocal collision avoidance for multiple non-holonomic robots. In Alcherio Martinoli, Francesco Mondada, Nikolaus Correll, Grégory Mermoud, Magnus Egerstedt, M. Ani Hsieh, Lynne E. Parker, and Kasper Støy, editors, *Distributed Autonomous Robotic Systems: The 10th International Symposium*, pages 203–216. Springer Berlin Heidelberg, 2013. [Page 24.]
- [66] Jur Van Den Berg, Jamie Snape, Stephen J. Guy, and Dinesh Manocha. Reciprocal collision avoidance with acceleration-velocity obstacles. In *2011 IEEE International Conference on Robotics and Automation*, pages 3475–3482, May 2011. [Page 24.]
- [67] Martin Rufli, Javier Alonso-Mora, and Roland Siegwart. Reciprocal collision avoidance with motion continuity constraints. *IEEE Transactions on Robotics*, 29(4):899–912, 2013. [Pages 24, 65, and 128.]

- [68] Javier Alonso-Mora, Tobias Naegeli, Roland Siegwart, and Paul Beardsley. Collision avoidance for aerial vehicles in multi-agent scenarios. *Autonomous Robots*, 39(1):101–121, 2015. [Pages [24](#), [27](#), [28](#), [84](#), [86](#), [128](#), and [148](#).]
- [69] Javier Alonso-Mora, Paul Beardsley, and Roland Siegwart. Cooperative collision avoidance for nonholonomic robots. *IEEE Transactions on Robotics*, 34(2):404–420, 2018. [Page [24](#).]
- [70] Zhiguo Ren, Panayiotis Charalambous, Julien Bruneau, Qunsheng Peng, and Julien Pettr. Group modeling: A unified velocity-based approach. *Computer Graphics Forum*, 36(8):45–56, 2017. [Page [24](#).]
- [71] Bharath Gopalakrishnan, Arun Kumar Singh, Meha Kaushik, K Madhava Krishna, and Dinesh Manocha. Prvo: Probabilistic reciprocal velocity obstacle for multi robot navigation under uncertainty. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1089–1096, Sep. 2017. [Page [24](#).]
- [72] Cherry Cheung, Yazdi I. Jenie, and Erik-Jan Van Kampen. Experimental validation of the selective velocity obstacle method for autonomous collision avoidance. In *AIAA Guidance, Navigation, and Control Conference*, page 1912, 2017. [Page [24](#).]
- [73] Beom H. Lee, Jae D. Jeon, and Jung H. Oh. Velocity obstacle based local collision avoidance for a holonomic elliptic robot. *Autonomous Robots*, 41(6):1347–1363, 2017. [Page [24](#).]
- [74] Hyo Hyun Choi, Areeya Rubenecia, Paul Vincent Contreras, and Myungwhan Choi. Collision avoidance scheme for micro uavs delivering information. In *2016 International Conference on Information Networking (ICOIN)*, pages 45–50. IEEE, Jan 2016. [Page [25](#).]
- [75] Steven Roelofsen, Denis Gillet, and Alcherio Martinoli. Collision avoidance with limited field of view sensing: A velocity obstacle approach. In *2017*

- IEEE International Conference on Robotics and Automation (ICRA)*, pages 1922–1927, May 2017. [Page 25.]
- [76] Urs Borrmann, Li Wang, Aaron D. Ames, and Magnus Egerstedt. Control barrier certificates for safe swarm behavior. *IFAC-PapersOnLine*, 48(27): 68–73, 2015. [Page 25.]
- [77] Li Wang, Aaron D. Ames, and Magnus Egerstedt. Safety barrier certificates for collisions-free multirobot systems. *IEEE Transactions on Robotics*, 33(3): 661–674, 2017. [Pages 25, 26, 27, 28, 127, 129, 131, and 132.]
- [78] Xiangru Xu, Paulo Tabuada, Jessy W. Grizzle, and Aaron D. Ames. Robustness of control barrier functions for safety critical control. *IFAC-PapersOnLine*, 48(27):54–61, 2015. [Pages 25, 26, and 131.]
- [79] Li Wang, Aaron D. Ames, and Magnus Egerstedt. Safe certificate-based maneuvers for teams of quadrotors using differential flatness. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3293–3298, Singapore, May 2017. [Pages 26 and 130.]
- [80] Xiangru Xu, Jessy W Grizzle, Paulo Tabuada, and Aaron D. Ames. Correctness guarantees for the composition of lane keeping and adaptive cruise control. *IEEE Transactions on Automation Science and Engineering*, 15(3): 1216–1229, 2017. [Page 27.]
- [81] Eric Squires, Pietro Pierpaoli, and Magnus Egerstedt. Constructive barrier certificates with applications to fixed-wing aircraft collision avoidance. In *2018 IEEE Conference on Control Technology and Applications (CCTA)*, pages 1656–1661, Aug 2018. [Page 27.]
- [82] Christos K. Verginis and Dimos V. Dimarogonas. Closed-form barrier functions for multi-agent ellipsoidal systems with uncertain lagrangian dynamics. *IEEE Control Systems Letters*, 3(3):727–732, 2019. [Page 27.]
- [83] Srinivas Akella and Seth Hutchinson. Coordinating the motions of multiple robots with specified trajectories. In *Proceedings 2002 IEEE International*

- Conference on Robotics and Automation (Cat. No. 02CH37292)*, volume 1, pages 624–631, May 2002. [Page 29.]
- [84] Tom Schouwenaars, Bart De Moor, Eric Feron, and Jonathan How. Mixed integer programming for multi-vehicle path planning. In *2001 European Control Conference (ECC)*, pages 2603–2608. IEEE, Sep. 2001. [Page 29.]
- [85] Arthur Richards, Tom Schouwenaars, Jonathan How, and Eric Feron. Spacecraft trajectory planning with avoidance constraints using mixed-integer linear programming. *Journal of Guidance, Control, and Dynamics*, 25(4):755–764, 2002. [Page 29.]
- [86] Matthew G Earl and Raffaello D’andrea. Iterative milp methods for vehicle-control problems. *IEEE Transactions on Robotics*, 21(6):1158–1167, 2005. [Page 29.]
- [87] Daniel Mellinger, Alex Kushleyev, and Vijay Kumar. Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams. In *2012 IEEE International Conference on Robotics and Automation*, pages 477–483, May 2012. [Pages 29 and 31.]
- [88] Daniel Morgan, Soon-Jo Chung, and Fred Y. Hadaegh. Model predictive control of swarms of spacecraft using sequential convex programming. *Journal of Guidance, Control, and Dynamics*, 37(6):1725–1740, 2014. [Page 30.]
- [89] Daniel Morgan, Giri P. Subramanian, Soon-Jo Chung, and Fred Y. Hadaegh. Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming. *The International Journal of Robotics Research*, 35(10):1261–1285, 2016. [Pages 30, 32, and 128.]
- [90] Sarah Tang, Justin Thomas, and Vijay Kumar. Hold or take optimal plan (hoop): A quadratic programming approach to multi-robot trajectory generation. *The International Journal of Robotics Research*, 37(9):1062–1084, 2018. [Page 31.]

- [91] Yu Wan, Jun Tang, and Songyang Lao. Distributed conflict-detection and resolution algorithms for multiple uavs based on key-node selection and strategy coordination. *IEEE Access*, 7:42846–42858, 2019. [Pages 31 and 32.]
- [92] Rahul Tallamraju, Sujit Rajappa, Michael J. Black, Kamalakar Karlapalem, and Aamir Ahmad. Decentralized mpc based obstacle avoidance for multi-robot target tracking scenarios. In *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–8, Aug 2018. [Pages 31 and 32.]
- [93] Baskın Şenbaşlar, Wolfgang Hönig, and Nora Ayanian. Robust trajectory execution for multi-robot teams using distributed real-time replanning. In Nikolaus Correll, Mac Schwager, and Michael Otte, editors, *Distributed Autonomous Robotic Systems*, pages 167–181. Springer International Publishing, Cham, 2019. [Pages 31 and 32.]
- [94] Dsta@71 innovation call, 2019. URL https://www.dsta.gov.sg/71/InnoCall_ChallengeStatement_190212.pdf. [Page 36.]
- [95] Rafael Yanushevsky. *Guidance of unmanned aerial vehicles*. CRC press, 2011. [Page 37.]
- [96] Satadal Ghosh, Debasish Ghose, and Soumyendu Raha. Capturability of augmented pure proportional navigation guidance against time-varying target maneuvers. *Journal of Guidance, Control, and Dynamics*, 37(5):1446–1461, 2014. [Page 37.]
- [97] Ramsingh G. Raja, Charu Chawla, and Radhakant Padhi. Dynamic inversion-based nonlinear aiming point guidance of unmanned aerial vehicles for reactive obstacle avoidance. *Unmanned Systems*, 1(02):259–275, 2013. [Page 37.]
- [98] Vitaly Shaferman and Tal Shima. Cooperative optimal guidance laws for imposing a relative intercept angle. *Journal of Guidance, Control, and Dynamics*, 38(8):1395–1408, 2015. [Page 37.]

- [99] Xbee 2.4ghz modules. URL <https://www.digi.com/products/xbee-rf-solutions/2-4-ghz-modules>. [Page 43.]
- [100] Abdul Hanif Zaini and Lihua Xie. Triggered communication for reciprocal collision avoidance in multi-agent navigation. In *2018 IEEE 14th International Conference on Control and Automation (ICCA)*, pages 34–41, June 2018. [Pages 55 and 149.]
- [101] Maurice Heemels, Karl Henrik Johansson, and Paulo Tabuada. An introduction to event-triggered and self-triggered control. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 3270–3285. IEEE, 2012. [Page 56.]
- [102] Chai K Toh. *Ad hoc mobile wireless networks: protocols and systems*. Pearson Education, 2001. [Pages 58 and 82.]
- [103] Abdul Hanif Zaini and Lihua Xie. A hybrid collision avoidance algorithm for multi-agent traffic coordination under limited communication. In *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 1879–1884, Nov 2018. [Pages 79 and 149.]
- [104] Abdul Hanif Zaini and Lihua Xie. Distributed drone traffic coordination using triggered communication. *Unmanned Systems*, 08(01):1–20, 2020. [Pages 79 and 149.]
- [105] Matthew Turpin, Nathan Michael, and Vijay Kumar. Capt: Concurrent assignment and planning of trajectories for multiple robots. *The International Journal of Robotics Research*, 33(1):98–112, 2014. [Page 81.]
- [106] Shu-peng Lai, Meng-lu Lan, Ya-xuan Li, and Ben M. Chen. Safe navigation of quadrotors with jerk limited trajectory. *Frontiers of Information Technology & Electronic Engineering*, 20(1):107–119, 2019. [Pages 129, 130, and 143.]
- [107] Ola Dahl. Path-constrained robot control with limited torques-experimental evaluation. *IEEE transactions on robotics and automation*, 10(5):658–669, 1994. [Page 129.]

-
- [108] Hung Pham and Quang-Cuong Pham. Time-optimal path tracking via reachability analysis. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3007–3012, May 2018. [Page [129](#).]
- [109] Ayush Agrawal and Koushil Sreenath. Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation. In *Robotics: Science and Systems XIII Conference*, July 2017. [Pages [131](#) and [132](#).]
- [110] Peter Corke. *Robotics, vision and control: fundamental algorithms In MATLAB® second, completely revised*, volume 118. Springer, 2017. [Pages [138](#) and [139](#).]