Chair of Automatic Control
Department of Mechanical Engineering
Technical University of Munich

Toward Control of Multi-agent Systems: Cooperative Navigation of Autonomous Agents in Unknown Environments

Ertuğ Olcay

Vollständiger Abdruck der von der Fakultät für Maschinenwesen der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzender:	Prof. Dr.-Ing. Veit St. Senner
Prüfer der Dissertation:	1. Prof. Dr.-Ing. habil. Boris Lohmann
	2. Prof. Maruthi R. Akella, Ph.D. The University of Texas at Austin Austin (TX), USA

Die Dissertation wurde am 23.06.2020 bei der Technischen Universität München eingereicht und durch die Fakultät für Maschinenwesen am 03.12.2020 angenommen.

Abstract

Cooperative control of multi-agent systems has been investigated intensively for many different purposes. These include not only technical applications, such as swarm robotics, transportation and logistics but also control of human crowds and even understanding of collective behavior of animal groups. From the control engineering perspective, influencing autonomous, interacting multiple agents in a harmonic way is a challenging task. Since the motivation in usage of multiple agents is mostly to increase efficiency at low operational costs, multi-robot systems usually consist of simple, low cost robots. Engineering applications of mobile multi-agent systems vary from exploration and reconnaissance missions to surveillance systems.

This thesis proposes cooperative navigation methods, which are particularly suitable for agents with basic equipment, such as simple range sensors and communication devices. Especially agents, which are operated indoors, may not use a global positioning system. Hence, they have to localize themselves in a map using sensors and odometry. However, in real-world applications, the map of the environment usually does not contain all information, such as obstacles and restricted areas. Hereby, we focus on different specific problem sets and propose novel solutions for them. The considered scenarios cover collective motion planning toward a global target position, trajectory tracking, and area coverage with multiple agents. Beyond collective navigation by preventing inter-agent collisions, obstacle avoidance is one of the primary objectives in each considered task. Unknown obstacles in the workspace of agents usually bring about decision problems in collective motion planning. Particularly, complex obstacle geometries are difficult to handle in cooperative fashion. Since the communication ability of agents is limited, preserving connectivity, in other words, cohesive behavior, is desired in most tasks except area coverage. Through spatial proximity, agents can exchange more information and thus, localize themselves better in the unknown environment using aggregated information from other agents within their communication range.

The main tools in our control mechanisms are potential field approach, geometry-based algorithms and grid-based methods. Potential field-based interactions among the agents, along with virtual forces for obstacle avoidance, yield a common problem called local minima. For this issue, we propose cooperative control strategies with geometry-based algorithms to determine waypoints to escape from local minima and employ grid-based methods to virtually build the unknown workspace. Hereby, an elaborate exchange of information plays a significant role. A further method considered for collective navigation in this work employs optimal control. The presented strategies in this thesis are suitable for different scenarios. The abilities, advantages and drawbacks of the proposed navigation schemes are discussed and demonstrated with numerical experiments.

Acknowledgments

This dissertation is the result of my doctoral research done at the Chair of Automatic Control at the Technical University of Munich between September 2016 and March 2020. The present work would not have been possible without the support of several people I would hereby like to acknowledge.

I owe special thanks to my supervisor Prof. Dr.-Ing. habil. Boris Lohmann for giving me the opportunity to do research in the field of multi-agent systems. His support, trust and freedom he gave me during my time spent at the Chair of Automatic Control greatly impacted my work and the results of this thesis. Thanks to him, I have learned so much in these past years, made new friends at conferences and especially expanded my horizon. Secondly, I would like to thank Prof. Maruthi R. Akella for giving me the chance to visit him at the University of Texas at Austin. He has inspired and encouraged my research. I have greatly benefited from our fruitful discussions on multi-agent coordination problems and enjoyed the time in Austin. I would also like to thank Prof. Dr.-Ing. Veit Senner for serving as chair of the examination committee. In addition, I would like to deeply thank all of my colleagues at the Chair of Automatic Control for their friendship, especially Christian Dengler. It was fun and also motivating to work together and share the office with him.

I would like to thank the German Research Foundation (DFG) for funding the research project, which is part of the Collaborative Research Centre 768 (Sonderforschungsbereich 768) Managing cycles in innovation processes – Integrated development of product service systems based on technical products. This thesis would not have been possible without the financial support.

I also would like to express my gratitude to Dr.-Ing. Uğur Karban for mentoring me and for giving helpful advises during this work.

Finally, I would like to deeply thank all students I have supervised. The success of my dissertation and the research project was largely possible due to their motivation and efforts.

Last but not least, my special thanks go to my family: to my parents and my brother, who have supported and encouraged me unconditionally, and to Carina, who was always at my side with great patience.

Munich, June 18, 2020 *Ertuğ Olcay*

Contents

1 Introduction

> *To return to the difficulty which has been stated with respect both to definitions and to numbers, what is the cause of their unity? In the case of all things which have several parts and in which the totality is not, as it were, a mere heap, but the whole is something beside the parts, there is a cause; for even in bodies contact is the cause of unity in some cases, and in others viscosity or some other such quality.*
>
> — Aristotle, *Complete Works* - Book VIII[1]

Rephrasing Aristotle, a system is something beside, and not the same, as its subsystems. Local interactions among the subsystems and their individual behaviors may yield global behaviors. Also many complex tasks can be divided into smaller tasks and these are allocated to the individual entities, the so-called *agents*, which are directly or indirectly interconnected with each other. In this way, a global complex problem can be solved by the agents in a collaborative fashion. Modeling, control, and predictions using multi-agent systems (MASs) have been intensively studied in different disciplines, including computer science, biology, statistical physics and engineering. The agents in a multi-agent system represent dynamical subsystems that can refer to autonomous robots, unmanned aerial vehicles (UAVs), sensors or humans depending on the problem.

Due to their efficiency, low cost, flexibility, and reliability, MASs have a wide range of applications in computer networks, robotics, modeling, and smart grids [43]. Computer networks are complex systems due to the high number of interconnected devices. For example, MAS-based approaches were proposed for control, monitoring and security of computer networks [116, 63]. In addition, MAS-based modeling techniques help to simulate human social behavior, decision-making and opinion-forming, where the humans are agents [61, 72], [153, 156][2].

Robots are continuously developed to take over the *dull, dirty* and *dangerous* tasks from humans, also known as the 3-Ds. Most of these tasks can be performed more efficiently and even sometimes with lower operational costs by multiple robots than by a single robot. In order to achieve a common goal, multiple robots require cooperative control strategies. Based on the characteristics defined in [118, p. 1], a cooperative MAS should have the following features:

[1] Translated by W. D. Ross.

[2] The supervised student thesis [159] is also an example for the MAS-based modeling of a social system.

- Trajectories of all the agents evolve collaboratively toward accomplishing a common objective. The common objective can be interpreted as the equilibrium point of the system.
- Agents can usually exchange some information via a communication network.
- Agents can generally interact with a dynamically changing physical environment, which might have an influence on the system's dynamics and communication network.
- A state change of each agent may be subject to both constraints in kinematics and dynamics.

For further theory about cooperative control, the reader is referred to [118] and the study [69] gives an overview on the collective control of MASs. In addition, the book [82] summarizes the methods and challenges in the control of dynamic networked systems.

Besides cooperative control systems, there are natural systems that do not require external instructions for a specific task. Such systems are referred to as *self-organized systems*, which are a subcategory of MAS. Internal influences through interactions among the agents yield some form of order out of an initially disordered system. This natural order is usually a pattern arising mostly in physical and biological systems. Examples of self-organization include crystallization, the growth of snowflakes and galaxies. For further insights into the topic, the reader is referred to the literature [15, 30, 56, 138].

A form of self-organization can also be observed in biological systems with species that exhibit cooperative behavior to achieve a common goal (see Fig. 1.1). An example for this form of organization in nature is flocking behavior. *Flocking* represents a collective motion of a large number of self-driven and interacting individuals in a group. This collective behavior exhibited by entities aggregating together frequently appears in nature.

(a) Bees in the honeycomb storing nectar in its cells.

(b) Fisch school.

(c) Geese flying in formation.

Figure 1.1: Cooperative behaviors in nature[3].

[3] (a): https://pixabay.com/de/photos/honigbienen-bienenstock-honig-326337/
(b): https://pixabay.com/de/photos/fisch-schwarm-unterwasser-tauchen-1190393/
(c): https://pixabay.com/photos/formation-migratory-birds-geese-508038/

There are resemblances between schooling, swarming and herding behavior. The term flocking can also refer to swarm behavior in insects, herding behavior in land animals, and schooling of fishes. However, flocking usually denotes the motion of a population with a common velocity. Besides, *swarming* mainly refers to the phenomenon of cohesive motion, which does not necessarily require a common velocity [37].

1.1 Motivation

The research field of control of multi-robot systems and swarm robotics continues to draw inspiration from the behavior of cooperative animals in nature [46]. The mechanisms of cooperation, local interactions and communication between agents have been under increasing levels of investigation for many different purposes. Since all self-organizing systems do not have a reasonable goal, it is of interest to externally initiate a goal and steer a group toward desired objectives. A simple example for such an external control could be a herding dog that steers the flock (Fig. 1.2). Such external control mechanisms of group behaviors have been investigated for their potential engineering applications, especially for the control of multi-robot systems.

Autonomous multi-robot systems have a wide range of applications such as building the maps of unknown areas, discovery and exploration of certain regions [42]. In addition, monitoring, reconnaissance and rescue with swarms of autonomous mobile robots are other possible tasks for muti-robot systems [132, 84, 131, 96, 77]. Data collection in unknown or partially known environments [144], transportation in a warehouse [71], robot formation[4] and vehicle platooning [122, 48] are some further applications. A detailed description of different problems and approaches for MAS can be found in [115].

Figure 1.2: A herding dog coordinates a group of sheep[5].

In many multi-agent control problems, multiple agents collaboratively work on a common task. One of the major issues in many multi-robot coordination tasks is the difficult operation areas with complex obstacles and restricted domains. Autonomous vehicles

[4] We refer the reader to [135] for an elaborate study on formation control systems.
[5] `https://bit.ly/2wWAEgT`

and robots utilize different kinds of navigation strategies. Motion planning approaches for autonomous robots can be generally divided into two categories.

The first one is the global navigation approach (global motion planning or offline planning), which assumes that the robot receives a map of the environment before path planning. This map can include all static obstacles and boundaries of the considered area. In this case, an optimal path is computed based on the prior knowledge of the environment. Optimal control-based schemes [25, 87] and graph traversal algorithms such as A* algorithm [59] have been investigated for motion planning in this category. The weakness of these approaches is that the robot cannot handle environmental changes during an operation.

The second strategy is the local navigation approach (local motion planning or online planning), which does not require prior information about the environment. The robot can plan its actions autonomously based on sensor data and react to unexpected events in the workspace. Over the past two decades, several methods have been proposed for local navigation of multi-agent systems. The ability of robots to localize themselves on a map and to plan elaborated motions are the basics of many local navigation approaches. There are several Simultaneous Localization And Mapping (SLAM) algorithms that allow robots to build a map of the environment and, at the same time, use this map to compute their location. Since some SLAM algorithms are based on probabilistic methods, a combination of SLAM with other navigation mechanisms may increase efficiency [92]. Further information on robot localization techniques can be found in [32]. Artificial potential field-based methods are widespread approaches, which are based on the generation of artificial attraction, repulsion, and alignment forces [54, 98, 73, 75]. Further similar approaches also exist, such as the Vector Field Histogram (VFH) [24, 137, 80], which exhibits problems in handling local minima[6] (e.g., concave obstacles). Another method is Model Predictive Control (MPC) [5], which is applied to minimize a cost function repeatedly at each time step using a nonlinear dynamic model of the system. It is well-suited for changing environments in motion planning. However, this method is generally computationally intensive, especially in multi-agent settings.

In many cases, the operation area of the agents is usually unknown or partially known. A possible collision or loss of some agents during a task can cause expenses and lead to the failure of the mission. Hence, collision-free path planning of multi-robot systems is one of the keys to a successful mission. Perception of the environment, object recognition, position and velocity control are some of the constituent parts of robot guidance tasks. Many of the existing strategies either consider *idealized*, simple small obstacles or do not enable agents to escape from non-convex obstacles. Camera-based approaches are often used for object identification and obstacle avoidance. However, the acquisition costs for multiple robots can be high depending on the number of agents. Furthermore, high-intense sunlight or smoky environments can reduce their performance. Due to the low acquisition as well as their operational costs and simplicity, sensor-based approaches are popular in cooperative motion planning.

[6]The problem of local minimum will be explained in Chapter 4.

1.2 Statement of Contribution

In this thesis, we mainly focus on the navigation of dynamic multi-agent systems considering various agent dynamics and different problems. Control and navigation of swarming multi-agent systems are inspired by flocking behavior in nature. Three simple rules were defined to imitate flock-like behavior in a multi-agent system, *cohesion, separation, and alignment* [123], which are consistently taken into consideration in our approaches, especially in multi-agent rendezvous problems[7]. The aim of this thesis is to extend the literature on cooperative navigation. On top of this, the main goal of this work is to develop viable approaches using distributed control strategies, which are easy-scalable with respect to the system size (agent number), adaptable to environmental changes and simple to implement.

In **Chapter** 3, we investigate the evolution and properties of the Cucker-Dong model [37], which represents a self-organizing system under specific conditions. The model is quite stable with respect to *separation* without any condition. Thus, preserving this property, we extended the Cucker-Dong model to build a target-tracking and an obstacle-avoiding flock. This extension includes the obstacle avoidance algorithm to avoid convex obstacles and we have added a common group objective to the CD model. Study [152] was published in connection with this chapter.

With the aforementioned motivation, in **Chapter** 4, we deal with a rendezvous problem and propose a heuristic information-driven algorithm for flocking systems to escape from concave obstacles and to prevent local minima. In this setting, the agents receive only the environmental information by using their sensors and via communication with other agents. The proposed method uses cellular decomposition of the workspace. With this algorithm, agents can explore the unknown workspace and create their information maps through local information exchange. In addition, recognition of concave obstacles allows the agents to perform escape maneuvers from possible local minima. Study [155] was published in connection with this chapter.

In **Chapter** 5, we propose a collective navigation framework considering the same problem as in the previous chapter. The presented framework is based on a tangential escape schema and a sophisticated information sharing via communication network. In contrast to the previous chapter, the communication protocol allows multiple robots to efficiently explore an unknown area through exchange of local information about critical points in the workspace along with actions of neighboring agents. Study [158] was submitted in connection with this chapter.

Chapter 6 addresses a cooperative coverage problem with multiple autonomous agents, which can be applied to exploration missions and even to floor cleaning. For this purpose, the exploration area is decomposed into identical cells. In many similar approaches, however, either the robots know the obstacle locations, or they are not capable of identifying their environment completely. In this chapter, we propose a sensor-based framework to cover a given workspace simultaneously with multiple mobile agents in a cooperative fashion without any prior knowledge of the environment. With our ap-

[7] In rendezvous problems, all agents desire to meet at a predefined location.

proach, the agents are capable of avoiding collisions with different shaped obstacles and autonomously constructing a virtual map of the whole area by identifying inaccessible domains. Study [157] was published in connection with this chapter.

In **Chapter** 7, we are concerned with the control of a group of autonomous agents with limited information through guidance of a single controlled agent, which is called the leader. In this setting, the leader is the only agent that knows the predetermined objective of the group and the environment. The agents in the group can only interact with the agents within their communication range. In addition, they cannot perceive the obstacles and interact with them. In order to control the leader, we utilize an optimal control strategy. With this method, the leader applies optimal interventions and steers the entire system toward group objectives. Considered objectives include rendezvous problem, trajectory tracking, and avoiding collisions with obstacles at the same time. Study [154] was published in connection with this chapter.

1.3 Outline of the Thesis

This thesis includes theoretical results in the cooperative and decentralized coordination of multiple autonomous agents for different application objectives. The thesis is organized in eight chapters, each one focusing on a specific problem.

Chapter 1 presents a general introduction to multi-agent systems, some of its applications and research problems investigated in this thesis.

Chapter 2 gives an overview of the theoretical background required to understand the upcoming chapters. It provides certain preliminaries for this dissertation. The basic notions of graph theory are presented because the concepts developed in this work frequently employ graph-theoretic approaches. Furthermore, preliminaries on modeling and control of cooperative agents, which are often applied in this work, are introduced.

Chapter 3 explores the potential of Newton-type particle systems with interacting particles for swarm navigation applications. The agents (particles) considered in this chapter use simple rules to organize themselves without any external intervention. However, the self-organization usually emerges under certain initial conditions. With the motivation of utilizing some properties of self-organized agents, the Cucker-Smale and the Cucker-Dong model are reviewed first. Furthermore, a navigation concept is proposed, which is built upon the Cucker-Dong model. Finally, numerical experiments are conducted to demonstrate the abilities of the concept.

Chapter 4 considers a local path planning problem for coordination of multi-agent systems in environments with complex obstacles without having any prior knowledge. A challenging factor in this chapter is the limited sensor and communication range of the agents. For the cooperative navigation, a potential function-based approach is employed. However, artificial potential field methods are usually plagued by being trapped into a local minimum. In order to overcome this problem, a heuristic escape strategy is proposed, which enables the agents to recognize possible local minima in an

anticipatory fashion and plan an appropriate motion through a multi-criteria decision-making process to escape from the potential trap.

Chapter 5 addresses the same problem as in Chapter 4 and presents an analytical navigation framework that solely exploits the sensing information and shared data among the agents for collision-free motion planning. The decentralized navigation concept employs potential fields to guarantee inter-agent collision avoidance, preserve proximity and increase safety in collision avoidance with obstacles. In addition, a sophisticated algorithm that uses local communication and information exchange is developed to enable the agents to make early decisions and simultaneous, optimal collective maneuvers in complex environments.

Chapter 6 deals with efficient area coverage using a mobile sensor network. The sensors denote agents with limited communication bandwidths and sensing performance. The challenge is that the agents explore a given workspace in a cooperative manner without having any prior knowledge of the environment. For this purpose, an area coverage algorithm for multiple mobile agents is developed. The proposed algorithm includes cooperative motion planning, autonomous map creation and a scheme for sensor-based recognition of inaccessible regions on the constructed map.

Motivated by the hierarchical group dynamics in nature, an optimal control-based concept for the coordination of multi-agent systems with an external leader is presented in Chapter 7. After a brief introduction to optimal control and numerical optimization approaches, the optimal control problem is formulated and solved using a numerical method. The proposed framework is validated in a simulation environment considering various group objectives.

Finally, Chapter 8 summarizes the thesis, provides concluding remarks and an outlook for future research directions.

The figures shown in this Introduction were licensed under a Creative Commons license. The figures in the rest of the work were produced by the author of this thesis.

2 Theoretical Background

This chapter introduces the theoretical basis that will be frequently applied in upcoming chapters. First, the essential preliminaries on graph theory, lattice-type geometry, perception of obstacles by means of sensors, its modeling for simulation purposes and mathematical tools to work with differentiable functions are presented. Finally, the general assumptions used in the thesis are summarized.

2.1 Preliminaries

The mathematical description of the network topology of a flocking or a swarming MAS is mainly based on graph theory. Thus, a basic knowledge is required to understand the geometrical structure of flocks. Moreover, the structural dynamics of flocks are described via artificial collective potentials. Their background is provided in the following sections. Since the dynamics of agents applied in the approaches in Chapter 4 - 7 are mainly based on those in [100], many of the preliminaries are also drawn from this work.

All vectors and matrices will be characterized in bold. *Note that in later chapters, slightly different notations than those used in this chapter may be used. They will be clearly explained and solely used in that particular chapter.*

2.1.1 Basic Notations of Graph Theory

Modeling and control of multi-agent systems bring graph theory and system dynamics together. A simple graph G without multiple edges or self-loops is an ordered or unordered pair $G = (\mathcal{V}, \mathcal{E})$ consisting of:

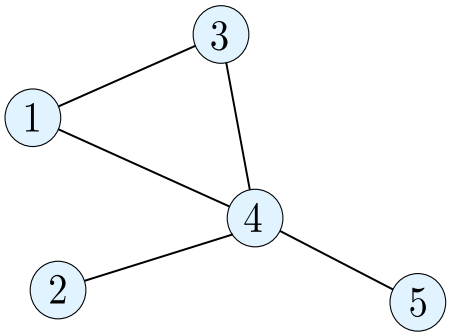
- $\mathcal{V} = \{1, .., n\}$, a set of n vertices (or nodes) and
- $\mathcal{E} \subseteq \{(i, j) : i, j \in \mathcal{V}, j \neq i\}$, a set of edges, also referred to as links or lines.

With $(i, j) \in \mathcal{E} \Leftrightarrow (j, i) \in \mathcal{E}$, the graph is undirected, i.e., the edges between nodes do not have a specific direction and exchanged information is bidirectional. In the context of this thesis, the vertices represent the agents and the edges represent the inter-agent connections. The quantities $|\mathcal{V}|$ and $|\mathcal{E}|$ denote the order and the size of graph.

The *adjacency matrix* of a graph is a square matrix $\mathbf{A} = [a_{ij}]$ representing the connectivity of vertices on a graph by containing non-zero elements in a_{ij} if and only if the nodes i and j are connected by an edge. If the graph does not have self-loops, all diagonal elements a_{ii} are equal to 0. In the case of an undirected graph, the adjacency matrix is symmetric ($\mathbf{A} = \mathbf{A}^\top$). Moreover, the adjacency matrix of a weighted graph

can contain elements of any value from 0 to 1. In an unweighted graph, only 0- and 1-elements exist in the adjacency matrix. The set of neighbors of each node i is defined as

$$\mathcal{N}_i = \{j \in \mathcal{V} : a_{ij} \neq 0\} = \{j \in \mathcal{V} : (i,j) \in \mathcal{E}\}. \tag{2.1}$$



$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure 2.1: Example for an undirected, unweighted simple graph with 5 vertices and the corresponding adjacency matrix.

In a MAS, each node representing an agent i has an interaction range $r > 0$. If the Euclidean distance between the agent i and an agent j with $j \in \mathcal{V}$ is shorter than r, they are considered spatial neighbors.

Let $\mathbf{p} = (\mathbf{p}_1^\top, \ldots, \mathbf{p}_n^\top)^\top \in \mathbb{R}^{mn}$ be the configuration of all n nodes of the graph with all nodes $i \in \mathcal{V}$ and their positions $\mathbf{p}_i \in \mathbb{R}^m$. Then, the pair $(G, \mathbf{p})$ is called a *structure* that consists of a graph and the configuration of all its nodes.

For an interaction range $r > 0$, a proximity net $G(\mathbf{p}) = (\mathcal{V}, \mathcal{E}(\mathbf{p}))$ can be defined with the set of edges between vertices, which are spatial neighbors. Together with the configuration of positions of all nodes, $(G(\mathbf{p}), \mathbf{p})$ is called a proximity structure.

The proximity net is undirected if all interaction ranges r are identical. In this work, since the agents are able to communicate with each other within a given identical communication range, all proximity nets are undirected, i.e. the communication is bidirectional. For more details on graph theory and its application in networked control, the reader is referred to [89].

2.1.2 Lattice-Type Geometry among Agents

In order to describe the geometry of a flock by a proximity net, let $\mathcal{V}_\alpha = \{1, 2, \ldots, n\}$ be the set of vertices and $\mathcal{E}_\alpha \subseteq \{(i,j) : i,j \in \mathcal{V}_\alpha, j \neq i\}$ be the set of edges of a graph G_α free of multiple edges or self-loops. The adjacency matrix with $a_{ij} \neq 0 \Leftrightarrow (i,j) \in \mathcal{E}_\alpha$ of this graph is symmetric. The set of all vertices $\mathcal{V}_\alpha$ represents the members of the flock. These physical agents will from now on also be referred to as α-agents.

The distance between two α-agents j and i is defined as the Euclidean norm $\|\cdot\|$ in $\mathbb{R}^m$, i.e., $d_{ij} = \|\mathbf{p}_j - \mathbf{p}_i\|$ with $(i,j) \in \mathcal{V}_\alpha$. Since the α-agents can only communicate within a certain specified interaction range, for the communication of two agents, the distance between them has to be smaller than their interaction range r_c[1]. The agents that are within the interaction range are called neighboring agents or *neighbors*. The *neighborhood* of an agent i is defined as follows:

$$\mathcal{N}_i^\alpha = \{j \in \mathcal{V}_\alpha : \|\mathbf{p}_j - \mathbf{p}_i\| < r_c\}. \tag{2.2}$$

With the set of edges given by

$$\mathcal{E}_\alpha(\mathbf{p}) = \{(i,j) : i \in \mathcal{V}_\alpha,\, j \in \mathcal{N}_i^\alpha\}, \tag{2.3}$$

a proximity net $G_\alpha(\mathbf{p}) = (\mathcal{V}_\alpha, \mathcal{E}_\alpha(\mathbf{p}))$ can be defined.

Flocking agents usually tend to form certain regular spatial structures. In most cases, each α-agent is desired to keep a predefined distance $d, (r_c > d > 0)$ to its neighbors. The mathematical boundary condition for this can be given as

$$\|\mathbf{p}_j - \mathbf{p}_i\| = d, \qquad \forall j \in \mathcal{N}_i^\alpha(\mathbf{p}). \tag{2.4}$$

If condition (2.4) is fulfilled by all agents, they obtain a geometric pattern referred to in [100] as the *α-lattice* (see Fig. 2.2(a)). In reality, flocking agents only approximate the α-lattice configuration. Therefore, [100] additionally refers to a quasi-α-lattice form. Agents in a *quasi-α-lattice* configuration satisfy

$$-\delta \leq \|\mathbf{p}_j - \mathbf{p}_i\| - d \leq \delta \qquad \forall (i,j) \in \mathcal{E}_\alpha(\mathbf{p}),\, \delta \ll d, \tag{2.5}$$

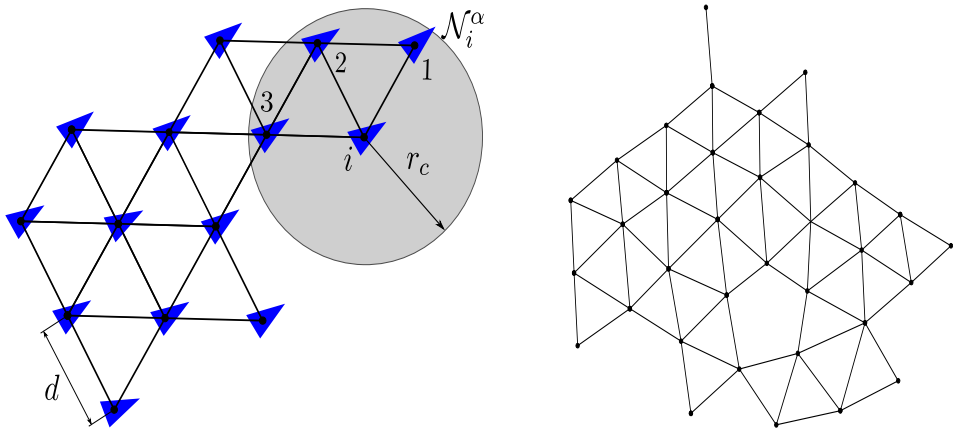
where δ denotes a small tolerance for the distance between the agents. Such a configuration is illustrated in Fig. 2.2(b) as an example.

The accuracy of the lattice-type formation can be analyzed by evaluating the deviation energy given by

$$E(\mathbf{p}) = \frac{1}{(|\mathcal{E}_\alpha(\mathbf{p})| + 1)} \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i^\alpha} \psi(\|\mathbf{p}_j - \mathbf{p}_i\| - d). \tag{2.6}$$

where the function ψ is a pairwise potential, e.g., $\psi(z) = z^2$. A more detailed explanation of potential functions follows in Section 2.2 and Section 4.1. The ideal α-lattice structure is reached at the global minimum of the deviation energy $E(\mathbf{p}) = 0$. However, even the deviation energy values on the order of 10^{-3} yield quasi-α-lattice structures. Hence, for an ideal lattice structure, quite a low energy is required.

[1]The interaction range r_c is also called *communication range* in later sections.



(a) α-lattice structure (b) Quasi-α-lattice structure

Figure 2.2: Illustration of lattice-type geometries.

2.1.3 Perception of Obstacles

In this thesis, we design different sensor-based motion planning methods for multiple agents. Hence, we assume that the α-agents are equipped with sensors to detect the presence of obstacles. The interaction topology between agents and obstacles are described by means of sensor data and thus, the so-called *virtual* β-agents are created. A β-agent is the virtual projection of an α-agent onto the obstacle. In this way, agents can virtually interact with obstacles.

Mathematically, an obstacle k lies within the neighborhood of an α-agent if it is positioned inside a predefined radius $r_s > 0$ describing the obstacle detection range or the *sensing range*. The set of detected obstacles is given by

$$\mathcal{N}_i^\beta = \{k \in \mathcal{V}_\beta : \|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\| < r_s\}, \tag{2.7}$$

where $\mathcal{V}_\beta = \{1, 2, \ldots, \tilde{n}\}$ is the set of all obstacles with $\tilde{n} \in \mathbb{N}$. The virtual position and velocity of each β-agent is denoted by $(\hat{\mathbf{p}}_{i,k}, \hat{\mathbf{v}}_{i,k}) \in \mathbb{R}^m \times \mathbb{R}^m$. An α-agent can sense multiple β-agents at the same time in narrow pathways shown in Fig. 2.3.

With the set of edges described by

$$\mathcal{E}_\beta(\mathbf{p}) = \{(i,k) : i \in \mathcal{V}_\alpha,\ k \in N_i^\beta\}, \tag{2.8}$$

a directed, bipartite proximity net $G_\beta(\mathbf{p}) = (\mathcal{V}_\beta, \mathcal{E}_\beta(\mathbf{p}))$ can be defined.

In order to ensure that the α-agents keep a certain safe distance d_s to the obstacles $(r_s > d_s > 0)$,

$$\|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\| = d_s \qquad \forall k \in \mathcal{N}_i^\beta(\mathbf{p}), \tag{2.9}$$

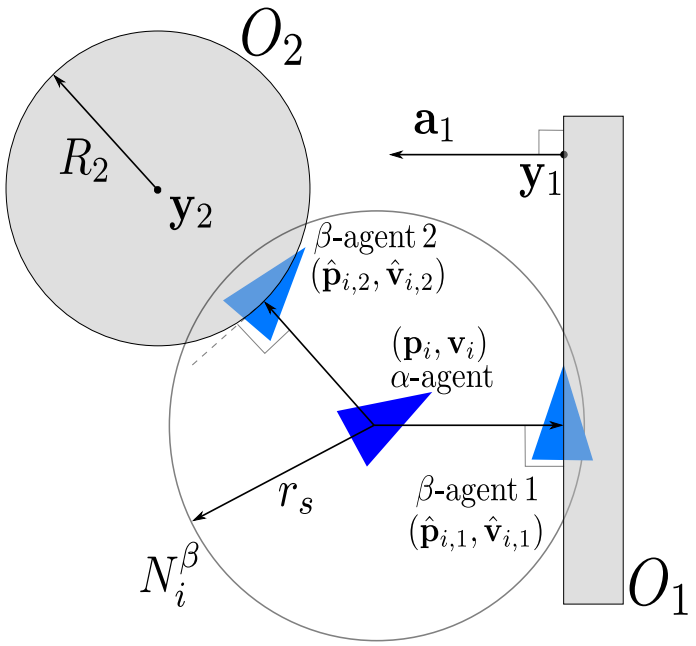


Figure 2.3: Different types of obstacles in the sensing region of an α-agent and their representations as β-agents. O_1 represents an ostacle with a hyperplane boundary and O_2 represents a spherical obstacle.

we will utilize later artificial repulsive forces. Satisfying condition (2.9) enables the agents to have a safe distance from the obstacles and thus, to avoid colliding with them.

In order to create a β-agent on an obstacle O_k, we have to determine its virtual position and velocity by using the sensor data. The virtual state $(\hat{\mathbf{p}}_{i,k}, \hat{\mathbf{v}}_{i,k})$ is basically the projection of the physical state onto the edge of the detected obstacle. The computation of these in a simulation environment is performed using the following calculation schemes based on obstacle geometry.

1. **Obstacles with a Hyperplane Boundary**

$$\hat{\mathbf{p}}_{i,k} = \mathbf{P}\mathbf{p}_i + (\mathbf{I}_m - \mathbf{P})\mathbf{y}_k, \qquad \hat{\mathbf{v}}_{i,k} = \mathbf{P}\mathbf{v}_i, \tag{2.10}$$

   where $\mathbf{P}$ is the projection matrix defined as

$$\mathbf{P} = \mathbf{I}_m - \mathbf{a}_k\mathbf{a}_k^\top, \tag{2.11}$$

   and $\mathbf{a}_k$ is the unit normal passing through a point $\mathbf{y}_k$ on the obstacle. Hereby, $\mathbf{I}_m \in \mathbb{R}^{m\times m}$ denotes an m-dimensional unit matrix.

2. **Spherical Obstacles**
   For a spherical obstacle with radius R_k and center point $\mathbf{y}_k$, the virtual state of β-agents is computed as follows:

$$\hat{\mathbf{p}}_{i,k} = \mu\mathbf{p}_i + (1-\mu)\mathbf{y}_k, \qquad \hat{\mathbf{v}}_{i,k} = \mu\mathbf{P}\mathbf{v}_i \tag{2.12}$$

with the projection matrix $\mathbf{P} = \mathbf{I}_m - \mathbf{a}_k\mathbf{a}_k^\top$. In this case, $\mathbf{a}_k = \frac{(\mathbf{p}_i - \mathbf{y}_k)}{\|\mathbf{p}_i - \mathbf{y}_k\|}$ represents the unit vector from the center point $\mathbf{y}_k$ to the position of α-agent. $\mu = \frac{R_k}{\|\mathbf{p}_i - \mathbf{y}_k\|}$ defines a distance ratio of radius to real distance from the center point and has a value in the range $(0, 1]$.

Remark 2.1. The equations for spherical obstacles in (2.12) can be simply reformulated as

$$\hat{\mathbf{p}}_{i,k} = \mathbf{p}_i + (1-\mu)(\mathbf{y}_k - \mathbf{p}_i), \quad \hat{\mathbf{v}}_{i,k} = \mu\mathbf{v}_i - \mu\mathbf{v}_{i\perp} = \mu(\mathbf{v}_i - \mathbf{v}_{i\perp}).$$

The appropriate portion of the vector from α-agent to the center point $(\mathbf{y}_k - \mathbf{p}_i)$ is added to the agent's current position $\mathbf{p}_i$. This portion corresponds to $(1-\mu)$. In this way, $\hat{\mathbf{p}}_{i,k}$ describes a point on the obstacle's surface. Additionally, the virtual velocity $\hat{\mathbf{v}}_{i,k}$ is computed by subtracting the component of the agent's velocity $\mathbf{v}_i$ in the direction of the obstacle center point $\mathbf{v}_{i\perp}$ from the agent's current velocity $\mathbf{v}_i$. Then, this is weighted with the distance ratio μ. △

Remark 2.2. These computation procedures are useful for the simulation. In real systems, $\hat{\mathbf{p}}_{i,k}$ can be perceived by means of sensors and $\hat{\mathbf{v}}_{i,k}$ is calculated using sensor measurements. △

2.1.4 Gradient Systems

The interactions among the agents in a MAS can be modeled using artificial potential fields, which generate artificial reactive forces (attractive or repulsive). In this thesis, some terms to control agent interactions are designed as *gradient systems*.

Mathematically denoted, a function $V : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ defined from the state space of n dimensions to the real scalar value is called a *potential function*, which may refer to the energy of a system. Assuming that $V(\mathbf{x})$ is twice differentiable, then the gradient of $V(\mathbf{x})$ is defined by

$$\nabla V(\mathbf{x}) = (\partial V/x_1, \ldots, \partial V/x_n). \tag{2.13}$$

Any point $\mathbf{x}^*$ satisfying $\nabla V(\mathbf{x}^*) = 0$ is a critical point. Checking the second derivative, namely the Hessian matrix $\mathbf{H}$ of the energy function, gives hints about this point. A positive definite $\mathbf{H}$ indicates a local maximum and a negative definite $\mathbf{H}$ indicates a local minimum [34].

With the aforementioned definitions, a gradient system is a dynamic system of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) = -\nabla V(\mathbf{x}), \tag{2.14}$$

where $V(\mathbf{x})$ is a potential function. In dynamics of a multi-agent system, $\mathbf{f}(\mathbf{x})$ can represent a force acting on an agent at the position $\mathbf{x}$. The following properties regarding the gradient systems are important.

- $\dot{V}(\mathbf{x}) \leq 0$ and $\dot{V}(\mathbf{x}) = 0$ if $\mathbf{x}$ is an equilibrium point of (2.14).
- At any point, where $\mathbf{f}(\mathbf{x}) \neq \mathbf{0}$, the trajectories of (2.14) are orthogonal to the level sets of $V(\mathbf{x})$. The critical point $\mathbf{x}^*$ with $\mathbf{f}(\mathbf{x}^*)$ can be an isolated minima of $V(\mathbf{x})$ when a neighborhood of $\mathbf{x}^*$, which does not contain any other minima, exists. Following this, $\mathbf{x}^*$ is an asymptotically stable equilibrium point of (2.14) [136].

For a further introduction to gradient systems, the reader is referred to [29, 136, 145].

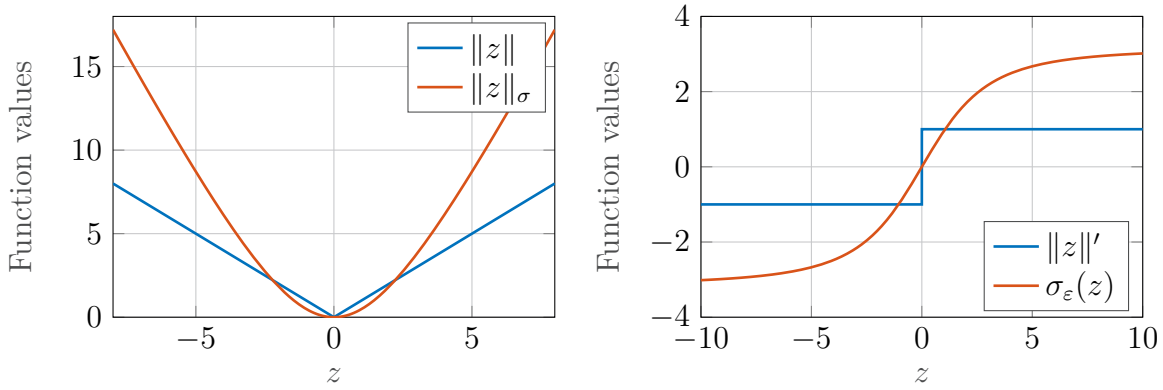
2.1.5 σ-Norm

Since the Euclidean norm is not differentiable at $z = 0$, the so-called σ-norm is introduced, which ensures differentiability of potential functions at any point. The σ-norm is not an actual norm but the mapping of a vector. Using a fixed parameter $\varepsilon > 0$, the map $\mathbb{R}^m \to \mathbb{R}_{\geq 0}$ is defined as

$$\|z\|_\sigma = \frac{1}{\varepsilon}\left[\sqrt{1 + \varepsilon\|z\|^2} - 1\right] \tag{2.15}$$

with the gradient of the σ-norm $\sigma_\varepsilon(z) = \nabla\|z\|_\sigma$ given by

$$\sigma_\varepsilon(z) = \frac{z}{\sqrt{1 + \varepsilon\|z\|^2}} = \frac{z}{1 + \varepsilon\|z\|_\sigma}. \tag{2.16}$$



(a) Plots of the σ-norm and the Euclidean norm.

(b) Comparing the gradients of σ-norm and the Euclidean norm.

Figure 2.4: Plots using $\varepsilon = 0.1$.

2.1.6 Smoothing of Functions

In order to smooth potential functions and generate weighted adjacency matrices, a scalar bump function $\rho_h(z)$ is helpful. In this thesis, we use the following bump function

$$\rho_h(z) = \begin{cases} 1, & z \in [0, h) \\ \frac{1}{2}\left[1 + \cos(\pi \frac{(z-h)}{(1-h)})\right], & z \in [h, 1] \\ 0, & \text{otherwise} \end{cases} \tag{2.17}$$

where $h \in (0, 1)$ (cf. [100, 105]).

This function is especially beneficial in our case because it only gives nonzero values in an interval defined by h. Thus, it can be used to generate the forces acting only in the specified interaction ranges. Also, the influence of the virtual force decreases with values of $z \geq h$, which represents the influence of forces decreasing at a greater distance.

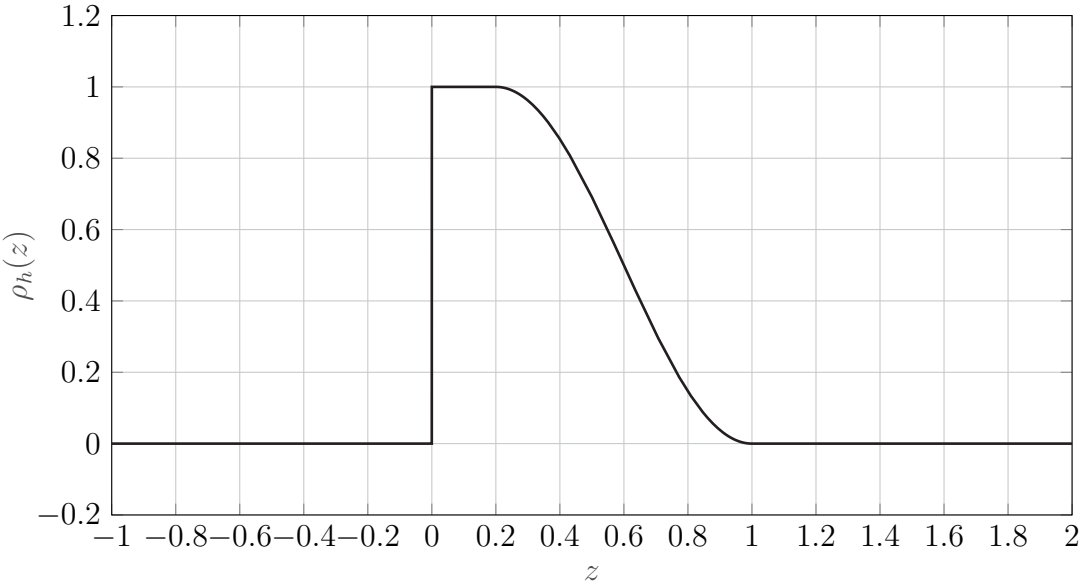


Figure 2.5: Plot of the bump function (2.17) using $h = 0.2$. This function will be later used to calculate the forces between interacting agents.

By using the aforementioned bump function, a smoothed spatial adjacency matrix can be defined element-wise with $r_\alpha = \|r_c\|_\sigma$ and $a_{ii} = 0 \; \forall (i, \mathbf{p})$ by:

$$a_{ij}(\mathbf{p}) = \rho_{h_\alpha}\left(\frac{\|\mathbf{p}_j - \mathbf{p}_i\|_\sigma}{r_\alpha}\right), \qquad j \neq i. \tag{2.18}$$

The adjacency matrix then contains zero elements for α-agent pairs, whose respective distances to each other are greater than r_α.

Analogous to that, the adjacency matrix for the interaction between β- and α-agents can be defined by

$$b_{ik}(\mathbf{p}) = \rho_{h_\beta}\left(\frac{\|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\|_\sigma}{d_\beta}\right), \qquad k \neq i, \tag{2.19}$$

where $d_\beta < r_\beta$ with $d_\beta = \|d_s\|_\sigma$, $r_\beta = \|r_s\|_\sigma$.

2.2 Modeling of Collective Potential Functions

A flock is supposed to form an α-lattice structure as described in Section 2.1.2. Therefore, it is appropriate to design a collective potential function by considering the deviation energy. The solutions satisfying the constraints (2.5) should be local minima of such a function $V(\mathbf{p}) : \mathbb{R}^{mn} \to \mathbb{R}_{\geq 0}$ and vice versa. This would guarantee that all the agents struggle to maintain a certain distance d to their neighbors. In order to have differentiable potential functions, we reformulate the constraints using σ-norm:

$$\|\mathbf{p}_j - \mathbf{p}_i\|_\sigma = d_\alpha, \qquad \forall j \in \mathcal{N}_i^\alpha(\mathbf{p}) \tag{2.20}$$

where $d_\alpha = \|d\|_\sigma$. This induces a smooth collective potential function given by

$$V(\mathbf{p}) = \frac{1}{2}\sum_i \sum_{j \neq i} \psi_\alpha(\|\mathbf{p}_j - \mathbf{p}_i\|_\sigma), \tag{2.21}$$

where $\psi_\alpha(z)$ is a smooth pairwise potential function, which generates repulsive or attractive forces between two individual agents dependent of the distance. Multiplying the term with $\frac{1}{2}$ eliminates the double-count of the term $\psi_\alpha(z)$ due the summation. The potential $\psi_\alpha(z)$ should have a finite cut-off at $r_\alpha = \|r_c\|_\sigma$ and the global minimum lies at $z = d_\alpha$ (see Fig. 2.6(b)). In order to construct $\psi_\alpha(z)$ with a finite cut-off, an action function $\varphi_\alpha(z)$ is defined as follows:

$$\varphi_\alpha(z) = \rho_{h_\alpha}(z/r_\alpha)\varphi(z - d_\alpha) \tag{2.22}$$

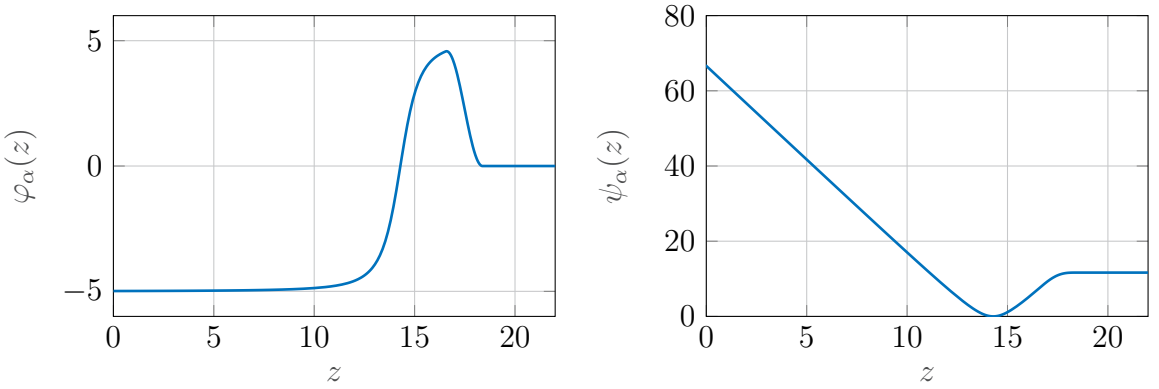
with $\varphi(z)$ defined as

$$\varphi(z) = \frac{1}{2}\left[(a+b)\sigma_1(z+c) + (a-b)\right], \tag{2.23}$$

where $\sigma_1 = \frac{z}{\sqrt{1+z^2}}$ and $\varphi(z)$ is an uneven sigmoid function (Fig. 2.6(a)) with parameters: $0 < a \leq b$, $c = \frac{|a-b|}{\sqrt{4ab}}$, which guarantee $\varphi(0) = 0$. Using the bump function results in (2.22) yields $\varphi_\alpha(z) = 0$ for all $z \geq r_\alpha$ (and also for $z < 0$). This means that the potential disappears if the distance between two agents is larger than their communication range.

The pairwise potential can then be defined by the integral over a corresponding action function $\varphi_\alpha(z)$ as follows:

$$\psi_\alpha(z) = \int_{d_\alpha}^{z} \varphi_\alpha(s)\mathrm{ds}. \tag{2.24}$$



(a) Plot of the action function $\varphi_\alpha(z)$.

(b) Smooth pairwise potential with finite cut-off in the global minimum at d_α.

Figure 2.6: Plots of action function and potential function with parameters: $d = 7$, $d_\alpha = 14.3$, $r_c = 1.2d$, $r_\alpha = 18.38$, $\varepsilon = 0.1$, $a = 5$, $b = 5$, $h_\alpha = 0.9$.

2.3 Multi-Agent Systems with Double-Integrator Dynamics

Many flocking, formation and consensus algorithms have been widely developed based on double-integrator dynamics. The motion of a wide range of vehicle classes, such as UAVs, holonomic ground vehicles or robots, can be simply modeled by using double-integrators.

The motion of an agent in a group usually emerges not only based on the virtual interaction forces through perception of other agents and obstacles, but also based on a group objective. A double-integrator multi-agent system can be described in the following form:

$$\dot{\mathbf{p}}_i = \mathbf{v}_i,$$
$$\dot{\mathbf{v}}_i = \mathbf{u}_i,$$

where $\mathbf{u}_i \in \mathbb{R}^m$ is the control input of agent i (e.g., $m = 2, 3$). In order to model and analyze the collective dynamics, a Laplacian matrix $\mathbf{L}$ is defined, which is a tool frequently employed in graph theory. The Laplacian matrix describes the relations between nodes and edges of a graph. In the context of the present work, it represents the relations among the agents in the MAS and it is used for the stability analysis. The

Laplacian matrix of the graph G consists of a diagonal matrix $\mathbf{D}$ and the adjacency matrix $\mathbf{A}$:

$$\mathbf{L} = \mathbf{D} - \mathbf{A}, \quad \in \mathbb{R}^{n \times n} \tag{2.25}$$

where the entries of $\mathbf{D}$ are the row sums $\sum_{j=1}^{n} a_{ij}$ of $\mathbf{A}$. This is also called the *degree matrix* of the graph. The following properties of $\mathbf{L}$ are of interest for this work:

- $\mathbf{L}$ has nonnegative eigenvalues

  $$\mathrm{Re}\{\lambda_i\} \geq 0, \quad i = 1, 2, \ldots n$$

  and satisfies the following for an undirected graph:

  $$\mathbf{z}^\top \mathbf{L} \mathbf{z} = \frac{1}{2} \sum_{(i,j) \in \mathcal{E}} a_{ij} (\mathbf{z}_i - \mathbf{z}_j)^2. \qquad \mathbf{z} \in \mathbb{R}^n \tag{2.26}$$

- The rank of $\mathbf{L}$ indicates the number of connected components[2] of the graph:
  - $c \geq 1$ connected components, if $\mathrm{rank}(\mathbf{L}) = n - c$
  - The graph is connected, if $\mathrm{rank}(\mathbf{L}) = n - 1$

The multi-agent system can be described in a more compact way by means of the Kronecker product, which is defined by

$$\mathbf{A} \otimes \mathbf{B} = [a_{ij}\mathbf{B}].$$

With this, each ij-th block of the Kronecker product of two matrices $\mathbf{A} \otimes \mathbf{B}$ equals $a_{ij}\mathbf{B}$, which yields an nm-dimensional Laplace matrix:

$$\underbrace{\hat{\mathbf{L}}}_{nm \times nm} = \underbrace{\mathbf{L}}_{n \times n} \otimes \underbrace{\mathbf{I}_m}_{m \times m}. \tag{2.27}$$

This allows a reformulation of the first property (2.26):

$$\mathbf{z}^\top \hat{\mathbf{L}} \mathbf{z} = \frac{1}{2} \sum_{(i,j) \in \mathcal{E}} a_{ij} (\mathbf{z}_i - \mathbf{z}_j)^2, \qquad \mathbf{z} \in \mathbb{R}^{nm} \tag{2.28}$$

where $\mathbf{z}^\top = (\mathbf{z}_1^\top, \mathbf{z}_2^\top, \ldots, \mathbf{z}_n^\top)$ is a single column vector and $\mathbf{z}_i \in \mathbb{R}^m$. The vectors used here can represent the position or velocity of agents.

[2]The connected component in the graph denotes a set of vertices that are linked to each other by edges.

LaSalle's Invariance Principle

Since the stability of flocking systems is analyzed by means of the LaSalle's invariance principle in later chapters, we recall [68, Theorem 4.4] and explain it in a few words. We consider an autonomous system

$$\dot{\mathbf{x}} = f(\mathbf{x}), \tag{2.29}$$

where $f : \mathcal{D} \to \mathbb{R}^n$ with the domain $\mathcal{D} \subseteq \mathbb{R}^n$ and the equilibrium point $\mathbf{x}_R$ satisfying $f(\mathbf{x}_R) = \mathbf{0}$. LaSalle's approach is basically an extension of the Lyapunov's stability theorem. The Lyapunov's theorem continues with assumption of the equilibrium position at the origin $\mathbf{x}_R = \mathbf{0}$. The system is considered asymptotically stable if a continuously differentiable function $V(\mathbf{x})$ exists such that $V(\mathbf{x}) : \mathcal{D} \to \mathbb{R}$ with $\mathcal{D} \subseteq \mathbb{R}^n$, $V(\mathbf{x})$ is positive definite on $\mathcal{D}$, and $\dot{V}(\mathbf{x})$ is negative definite on $\mathcal{D}$. The function $V(x)$ is also called the Lyapunov function. The Lyapunov function fails to satisfy the asymptotic stability condition if $\dot{V}(x)$ is only negative semidefinite. In this case, further investigations can be done based on the LaSalle's invariance principle which says:

$\mathcal{X}$ is a compact, positively invariant set with respect to (2.29) and $V : \mathcal{X} \to \mathbb{R}$ is a continuously differentiable function, which satisfies the condition $\dot{V}(\boldsymbol{x}) \leq 0$ in $\mathcal{X}$. The set $\mathcal{Y}$ is a set of all points in $\mathcal{X}$ for which $\mathcal{Y} = \{\boldsymbol{x} \in \mathcal{X} \,|\dot{V}(\boldsymbol{x}) = 0\}$. If $\mathcal{M}$ is the largest positive invariant set in $\mathcal{Y}$, then every solution starting in $\mathcal{X}$ approaches to $\mathcal{M}$ for $t \to \infty$.

Let $\mathbf{x}_R = \mathbf{0}$ be an equilibrium point of the system (2.29). Let a function $V(\mathbf{x}) : \mathcal{D} \to \mathbb{R}$ be a continuously differentiable one so that $V(\mathbf{x})$ on $\mathcal{D}$ is positive semidefinite and $\dot{V}(\mathbf{x})$ on $\mathcal{D}$ is negative semidefinite. The equilibrium point $\mathbf{x}_R = \mathbf{0}$ is asymptotically stable if the largest positive invariant subset $\mathcal{M}$ of $\mathcal{Y} = \{\mathbf{x} \in \mathcal{D} | \dot{V}(\mathbf{x}) = 0\}$ equals $\mathcal{M} = \{\mathbf{0}\}$.

In this work, the energy of flocking systems is described by the Hamiltonian function $H(\mathbf{p}, \mathbf{v})$, which is comparable to $V(\mathbf{x})$ in LaSalle's invariance principle. In this way, we can follow Lyapunov's theorem regarding stability. In addition, the cases with $\dot{H}(\mathbf{p}, \mathbf{v}) = 0$ are explained with LaSalle's invariance principle.

2.4 General Assumptions

Since the main focus of this work is laid on the theoretical development of motion planners, we have some general assumptions to reduce complexity in development and design. These are summarized in the following.

- In the considered scenarios, measurements are without delay. For dealing with an unknown, varying delay of measurements in a networked control system, we refer to [114].
- Sensor uncertainties are neglected so that the measurements are without noise.

  For dealing with noisy measurements, we refer to the distributed Kalman Filter (DKF) for an improved collective state prediction [102, 74, 103, 104, 143, 78].

Using DKF, agents fuse multiple measurements and covariance matrices in order to improve their estimations in a distributed way.

- The systems considered in this work do not have unmodeled dynamics. Study [76] can help if the agent dynamics are subject to uncertainties.
- Communication between agents occurs without package loss and time delay.

3 Control of a Self Driven Particle System

The inevitable fate of large groups is to perish because of lack of unity.
— Napoleon Bonaparte, *Aphorisms and Thoughts*

Parts of the following chapter have been published in [152].

Napoleon Bonaparte mentions in his book *Aphorisms and Thoughts* that large groups tend to dissolve as a result of missing consensus. Similarly, particles in a particle system require also some conditions or cooperative mechanisms for a collective behavior.

In 1987, Reynolds proposed three simple rules for computer animation of flocks of birds [123]. The first rule is *alignment*, which is a behavior that causes velocity match with agents close by. The second rule is *cohesion*, which makes agents stay close to each other. The last rule is *separation* that avoids collision with nearby flock mates.

A pioneering study on flocking is the Vicsek model, which aimed to show the collective motion of self-propelled particles [139]. The work of Vicsek *et al.* inspired many researchers and it was extended in study [33]. Based on this work, Cucker and Smale introduced a particle model in [39], the so-called Cucker-Smale (CS) model, describing the evolution of a flock and analyzed its velocity consensus behavior. Moreover, this model had also great impact on further studies later [130, 112, 55].

Many control strategies for flocking of multi-agent systems have been studied in recent years. In [41], the coordinated control of a group of autonomous mobile robots using sliding-mode controllers was investigated. Another attempt was model predictive flocking control to improve formation of agents [150]. Furthermore, Bongini *et al.* proposed sparse control, which is a control strategy to ensure flocking by actuating only a few agents instead of actuating the entire group at once [21, 22].

An intensively studied strategy to control flocks is the leader-follower approach. In the leader-based approach, the flock is controlled by a virtual leader [75] or by a physical leader, which is viewed by all follower agents [133, 134, 79, 117]. Another strategy is the optimal control technique investigated in the frame of leader-follower approach [65].

The chapter is structured as follows: After a general introduction to alignment models in Section 3.1, we begin with the Cucker-Smale model in Section 3.2. In Section 3.3,

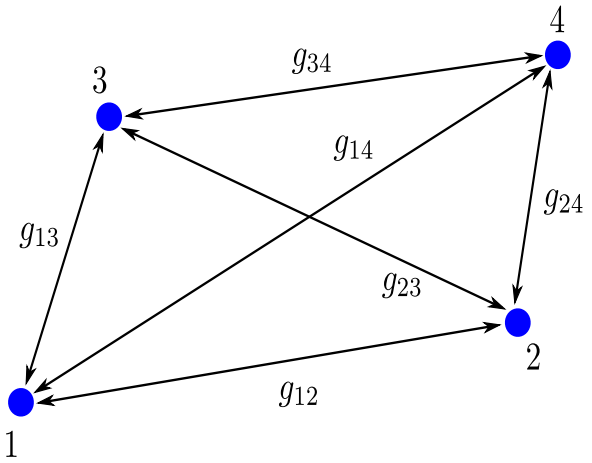


Figure 3.1: Information exchange between four agents.

the Cucker-Dong (CD) model and its main properties are reviewed in detail. Section 3.4 focuses on the leader-following strategy and its application to the CD model to navigate a group. In addition to the leader-based navigation, the reactive control law based on potential functions is applied to avoid circular and polygon-shaped obstacles. This section is based on [152], a contribution by the author of this thesis. In Section 3.5, simulation results are shown.

3.1 Introduction to Alignment Models

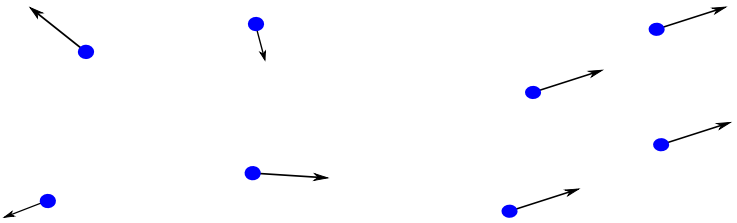
3.1.1 Dynamics and the Consensus State

Before we analyze the Cucker-Smale model and the Cucker-Dong model, we should define the properties of the so-called alignment models. Imitation is the mechanism that mainly determines the dynamics of the alignment systems, so that after a certain time, several states of the agents match. According to [21], the dynamics of a system with N agents are described by the following differential equations in a general form:

$$\begin{aligned} \dot{\mathbf{p}}_i(t) &= \mathbf{v}_i(t), \\ \dot{\mathbf{v}}_i(t) &= \sum_{j=1}^{N} g_{ij}(t)(\mathbf{v}_j(t) - \mathbf{v}_i(t)), \end{aligned} \tag{3.1}$$

for $i = 1, \ldots, N$ with $N \in \mathbb{N}$. The state of each agent is defined by $(\mathbf{p}_i, \mathbf{v}_i) \in \mathbb{R}^m \times \mathbb{R}^m$, which represent the position and velocity of the i-th agent in an m-dimensional space, respectively. The term $(\mathbf{v}_j(t) - \mathbf{v}_i(t))$ expresses the effort of the agent i to adapt its velocity to that of the agent j. The exchange of information about the individual states required for the matching is defined by the function $g_{ij}(t)$. If this matching occurs without external interventions, we call the process *self-organization*. Hereby, the

predominant state after the alignment of velocities is called *consensus* and is of central importance for the analysis of the system.



(a) Velocities before the alignment process.

(b) Velocities after the alignment process.

Figure 3.2: The alignment process.

The mean velocity of the agents is given by

$$\bar{\mathbf{v}}(t) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{v}_i(t). \tag{3.2}$$

A system converges to the state of consensus if the following holds for each agent $i = 1, \ldots, N$ (cf. [21], p. 8):

$$\lim_{t \to +\infty} \|\mathbf{v}_i(t) - \bar{\mathbf{v}}(t)\| = 0. \tag{3.3}$$

The velocity of each agent $\mathbf{v}_i$ includes the mean velocity $\bar{\mathbf{v}}$ and the undesired deviation $\mathbf{v}_i^{\perp}$ (Fig. 3.3). This can be written as:

$$\mathbf{v}_i(t) = \bar{\mathbf{v}}(t) + \mathbf{v}_i^{\perp}(t). \tag{3.4}$$

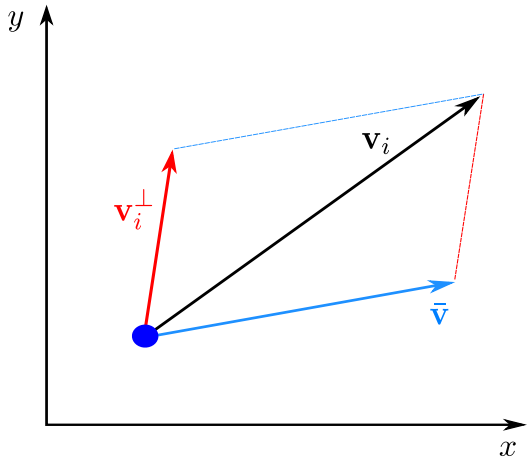


Figure 3.3: Partition of $\mathbf{v}_i$ in $\bar{\mathbf{v}}$ and $\mathbf{v}_i^{\perp}$.

Equation (3.3) is thus identical to the following expression (cf. [21], p. 15).

$$\lim_{t\to+\infty} \|\mathbf{v}_i^{\perp}\| = 0. \tag{3.5}$$

In this case, the velocity of each agent does not deviate from the mean velocity and thus, the alignment process is completed. The evolution of the mean velocity can be formulated as

$$\begin{aligned}
\dot{\bar{\mathbf{v}}}(t) &= \frac{1}{N}\sum_{i=1}^{N} \dot{\mathbf{v}}_i(t) \\
&= \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N} g_{ij}(t)(\mathbf{v}_j(t) - \mathbf{v}_i(t)) \\
&= \frac{1}{N}\left(\sum_{i=1}^{N}\sum_{j=1}^{N} g_{ij}(t)\mathbf{v}_j(t) - \sum_{i=1}^{N}\sum_{j=1}^{N} g_{ij}(t)\mathbf{v}_i(t)\right) \\
&= \frac{1}{N}\left(\sum_{i=1}^{N}\sum_{j=1}^{N} g_{ij}(t)\mathbf{v}_j(t) - \sum_{i=1}^{N}\sum_{j=1}^{N} g_{ji}(t)\mathbf{v}_j(t)\right) \\
&= \frac{1}{N}\sum_{j=1}^{N}\left(\sum_{i=1}^{N} g_{ij}(t) - \sum_{i=1}^{N} g_{ji}(t)\right)\mathbf{v}_j(t).
\end{aligned} \tag{3.6}$$

Since we assume that the information exchange is undirected, the ratio of information transfer from agent i to agent j is identical to that from agent j to agent i. This implies that

$$g_{ij}(t) = g_{ji}(t), \tag{3.7}$$

holds for $t \geq 0$. Using (3.7) in (3.6) yields

$$\dot{\bar{\mathbf{v}}}(t) = \mathbf{0} \tag{3.8}$$

for $t \geq 0$. Thus, the average velocity $\bar{\mathbf{v}}$ has a constant value and it is independent of time. We can write the consensus state $\mathbf{v}_\infty = \lim_{t\to+\infty} \bar{\mathbf{v}}(t)$ as follows:

$$\mathbf{v}_\infty = \frac{1}{N}\sum_{1=1}^{N} \mathbf{v}_i(0). \tag{3.9}$$

This indicates that the state of consensus depends only on the initial values of the system $\mathbf{p}_i(t=0)$ and $\mathbf{v}_i(t=0)$ (see [21], p. 9).

3.1.2 Dispersion and Dissent

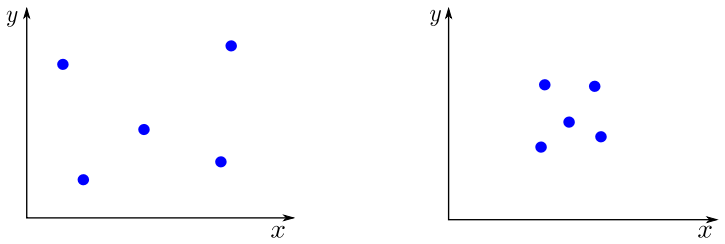
It is guaranteed that the system can converge to the consensus state for all arbitrary initial values $\mathbf{p}_i(0)$ and $\mathbf{v}_i(0)$. Hence, a detailed analysis of the system is required to evaluate its convergence behavior.

For a global analysis, i.e., in order to evaluate the system at the macro level, it is not appropriate to consider the state variables $\mathbf{p}_i$ and $\mathbf{v}_i$ of each agent separately. For this purpose, the terms *dispersion* and *dissent* are introduced. These terms define the state of the system in a global fashion, but they do not allow to make any statements about the individual states of the agents at the micro level.

The dispersion $\Gamma(t)$ is defined as

$$\Gamma(t) = \frac{1}{2N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|^2. \tag{3.10}$$

This is the mean square distance between the agents. Thus, the dispersion can be seen as a measure for the distribution of the agents in space (Fig. 3.4).



(a) High dispersion (b) Low dispersion

Figure 3.4: Illustration of the dispersion.

The dissent $\Lambda(t)$ is given as

$$\Lambda(t) = \frac{1}{2N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \|\mathbf{v}_i(t) - \mathbf{v}_j(t)\|^2, \tag{3.11}$$

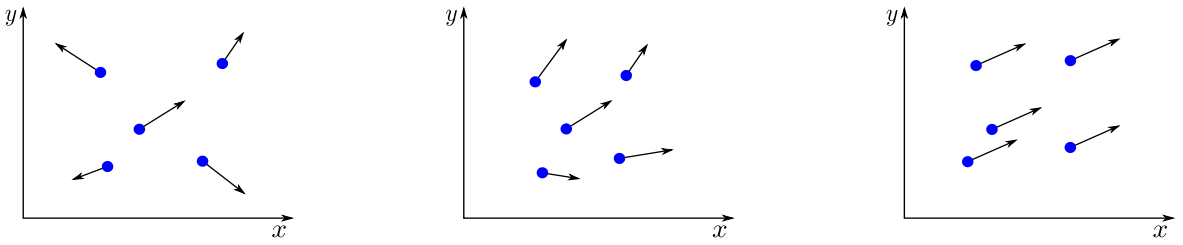
and it describes the mean square velocity difference between the agents. The dissent is a measure of the deviation of the system from the consensus state.

By utilizing

$$\sum_{j=1}^{N} \|\mathbf{v}_i(t) - \mathbf{v}_j(t)\|^2 = N\|\mathbf{v}_i(t) - \bar{\mathbf{v}}(t)\|^2 \tag{3.12}$$

and (3.4), we can rewrite (3.11) as

$$\Lambda(t) = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{v}_i^{\perp}(t)\|^2. \tag{3.13}$$



(a) High dissent (b) Low dissent (c) The consensus state

Figure 3.5: Illustration of the dissent.

This is a more convenient definition of the dissent that will be used from now on. Furthermore, with (3.5), another definition of convergence to consensus can be given as

$$\lim_{t\to+\infty} \Lambda(t) = 0. \tag{3.14}$$

3.2 The Cucker-Smale Model

The *Cucker-Smale* (CS) model introduced in [39] is an example for the alignment model. It describes the evolution of a flock and its dynamics are given by the following differential equations (cf. [39], p. 853):

$$\begin{aligned} \dot{\mathbf{p}}_i(t) &= \mathbf{v}_i(t), \\ \dot{\mathbf{v}}_i(t) &= \frac{1}{n(r)} \sum_{j=1}^{N} a(\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|)(\mathbf{v}_j(t) - \mathbf{v}_i(t)). \end{aligned} \tag{3.15}$$

The CS model has an identical form to (3.1). The interaction between the agents is described by the function $a : \mathbb{R}_{0,+} \to [0, H]$, which is defined as

$$a(r) = \frac{H}{(1 + r^2)^{\beta}}. \tag{3.16}$$

This function is shown in Fig. 3.6 and depends on the following parameters:

- r denotes the distance between two agents.
- H defines the strength of agent i to adapt its velocity to that of agent j.
- β determines the rate of decrease in the information exchange between two agents due to their distance from each other.

The normalizing term $\frac{1}{n(r)}$ in (3.15) describes the limited attention span of an agent or its limited ability to process information. If an agent i has only a few interaction partners in its environment, it interacts more strongly with these. This has a positive impact on the rate of its velocity match. On the other hand, if there are many agents

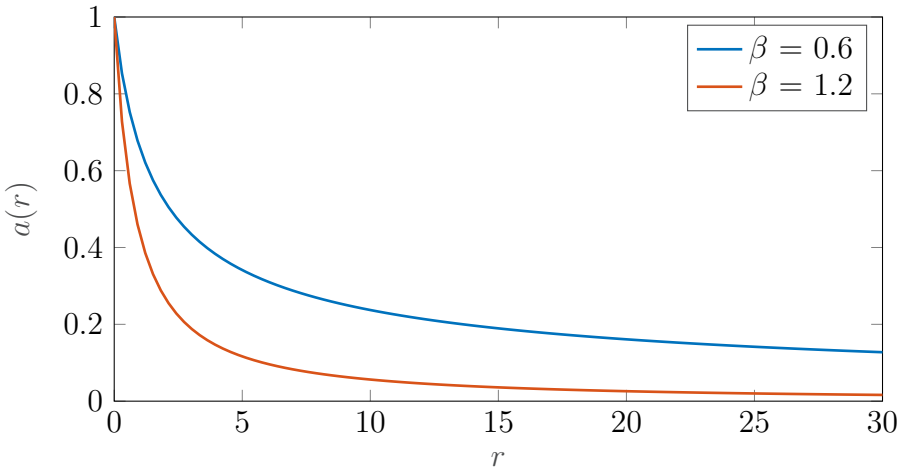


Figure 3.6: Interaction function $a(r)$ with $H = 1$.

in the environment of agent i, it has to allocate its attention more, which reduces the ability to match its velocity. In [21], a normalizing term is proposed as

$$n(t) = \frac{1}{\sum_{j=1}^{N} \Phi(\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|)} \tag{3.17}$$

with

$$\Phi(r) = \frac{1}{(1 + r^2)^{\beta}}. \tag{3.18}$$

Thereby, the information that agent i exchanges with agent j at time t is divided by the total information acquired by agent i at time t. However, this results in a directed interaction:

$$\frac{a(\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|)}{\sum_{k=1}^{N} \Phi(\|\mathbf{p}_i(t) - \mathbf{p}_k(t)\|)} \neq \frac{a(\|\mathbf{p}_j(t) - \mathbf{p}_i(t)\|)}{\sum_{k=1}^{N} \Phi(\|\mathbf{p}_j(t) - \mathbf{p}_k(t)\|)}. \tag{3.19}$$

This means that the information flow from agent i to agent j is not identical to that from agent j to agent i. However, this contradicts the assumption (3.7), which says $g_{ij}(t) = g_{ji}(t)$. Therefore, the term $1/N$ is simply used for the normalization, which is an approximation of $1/\Phi(r)$ and ensures an undirected information exchange.

In this way, the dynamics of the Cucker-Smale model can be rewritten as (cf. [38])

$$\begin{aligned} \dot{\mathbf{p}}_i(t) &= \mathbf{v}_i(t), \\ \dot{\mathbf{v}}_i(t) &= \frac{1}{N} \sum_{j=1}^{N} a(\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|)(\mathbf{v}_j(t) - \mathbf{v}_i(t)). \end{aligned} \tag{3.20}$$

3.2.1 Evolution of the Consensus through Self-Organization

The behavior of convergence to the consensus state, which denotes the decrease of dissent, is of particular interest in this section. Thus, we consider the evolution of consensus in the CS model. Since the system is not a closed-loop using output feedback, the consensus error $(\mathbf{v}_i - \bar{\mathbf{v}})$ cannot be reduced automatically. At first glance, the differential equations in (3.20) do not have any terms that describe repelling forces, which may yield an increase in dissent. Hereby, the only alignment term is $a(\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|)(\mathbf{v}_j(t) - \mathbf{v}_i(t))$. One can propose that even without a controller, the dissent may decrease and thus, the system converges to the consensus state. In the following, we analyze this proposition. The time derivative of the dissent is given by

$$\frac{\mathrm{d}}{\mathrm{dt}}\Lambda(t) = \frac{\mathrm{d}}{\mathrm{dt}}\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{v}_i^{\perp}(t)\|^2 = \frac{2}{N}\sum_{i=1}^{N}\dot{\mathbf{v}}_i^{\perp}(t)\cdot\mathbf{v}_i^{\perp}(t). \tag{3.21}$$

As formulated in (3.4), the velocity $\mathbf{v}_i$ of an agent is composed of the average velocity $\bar{\mathbf{v}}$ and the undesired velocity deviation (error) $\mathbf{v}_i^{\perp}$. The error can be defined as

$$\mathbf{v}_i^{\perp} = \mathbf{v}_i - \bar{\mathbf{v}} \tag{3.22}$$

and its time derivative is as follows:

$$\dot{\mathbf{v}}_i^{\perp} = \dot{\mathbf{v}}_i - \dot{\bar{\mathbf{v}}}. \tag{3.23}$$

Inserting (3.23) into (3.21) yields

$$\frac{\mathrm{d}}{\mathrm{dt}}\Lambda(t) = \frac{2}{N}\sum_{i=1}^{N}\dot{\mathbf{v}}_i(t)\cdot\mathbf{v}_i^{\perp}(t) - \frac{2}{N}\sum_{i=1}^{N}\dot{\bar{\mathbf{v}}}(t)\cdot\mathbf{v}_i^{\perp}(t). \tag{3.24}$$

Using the statement $\dot{\bar{\mathbf{v}}}(t) = \mathbf{0}$ from (3.8) results in

$$\frac{\mathrm{d}}{\mathrm{dt}}\Lambda(t) = \frac{2}{N}\sum_{i=1}^{N}\dot{\mathbf{v}}_i(t)\cdot\mathbf{v}_i^{\perp}(t). \tag{3.25}$$

By inserting $\dot{\mathbf{v}}_i(t)$ from 3.20 into (3.25), we finally obtain the following:

$$\frac{\mathrm{d}}{\mathrm{dt}}\Lambda(t) = \frac{2}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}a(\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|)(\mathbf{v}_j(t) - \mathbf{v}_i(t))\cdot\mathbf{v}_i^{\perp}(t) \tag{3.26}$$

for the time derivative of the dissent. For (3.16), it holds $a(\|\mathbf{p}_i(t) - \mathbf{p}_j(t)|) \leq H$ with $H \geq 0$. Thus, we can define an upper boundary as

$$\frac{\mathrm{d}}{\mathrm{dt}}\Lambda(t) \leq \frac{2}{N^2}H\sum_{i=1}^{N}\sum_{j=1}^{N}(\mathbf{v}_j(t) - \mathbf{v}_i(t))\cdot\mathbf{v}_i^{\perp}. \tag{3.27}$$

The following inequality always holds,

$$\sum_{i=1}^{N}\sum_{j=1}^{N}(\mathbf{v}_j(t)-\mathbf{v}_i(t))\cdot\mathbf{v}_i^{\perp}(t)\leq 0 \tag{3.28}$$

and it yields

$$\frac{\mathrm{d}}{\mathrm{dt}}\Lambda(t)\leq 0. \tag{3.29}$$

This shows that the dissent decreases without an external intervention.

3.2.2 Conditions for the Guaranteed Convergence

The expression (3.29) implies that the system converges to the consensus state. However, there is no guarantee for the absolute convergence $\Lambda(t)=0$. There might be a certain time instant t^*, at which $\frac{\mathrm{d}}{\mathrm{dt}}\Lambda(t^*)=0$ holds. This would mean that the dissent decreases no longer, but rather converges to a constant value $\Lambda(t^*)>0$. As (3.9) denotes, the consensus state depends only on the initial values. Therefore, we briefly analyze the initial conditions required for the absolute convergence to the consensus state.

As mentioned at the beginning of the chapter, it is inconvenient to consider the states of the agents separately. Therefore, the terms *dispersion* and *dissent* were introduced. In the previous section, we have formulated the time derivative of dissent (3.26) and noted that it decreases over time. The right-hand side of the equation is dependent of $\mathbf{p}_i(t)$ and $\mathbf{v}_i(t)$. However, we seek another formulation for the derivative of dissent in relation to dispersion. In [21], the following inequality is formulated.

$$\frac{2}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}a(\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|)(\mathbf{v}_j(t)-\mathbf{v}_i(t))\cdot\mathbf{v}_i^{\perp}(t)\leq -2a\left(\sqrt{2N\Gamma(t)}\right)\Lambda(t). \tag{3.30}$$

With this, the following holds

$$\frac{\mathrm{d}}{\mathrm{dt}}\Lambda(t)\leq -2a\left(\sqrt{2N\Gamma(t)}\right)\Lambda(t). \tag{3.31}$$

Assume that there exists $T>0$, for which $\Lambda(t)>0$ and $\Lambda(0)>0$. We integrate the inequality (3.31) using the Gronwall-Bellman lemma [106, Theorem 1.2.2][1] and we obtain

$$\Lambda(t)\leq\Lambda(0)\exp\Big(-2\int_0^t a\left(\sqrt{2N\Gamma(s)}\right)\mathrm{ds}\Big),\quad t\in[0,T]. \tag{3.32}$$

[1]The lemma allows to derive explicit boundaries from the implicit information of a certain integral inequality.

Obviously, the dissent exponentially decreases since the function $a(r)$ is strictly positive. The proposition in [31] defines a relation between $\Lambda(0)$ and $\Gamma(0)$ for the convergence of $\Lambda(t)$ to 0 for $t \to \infty$.

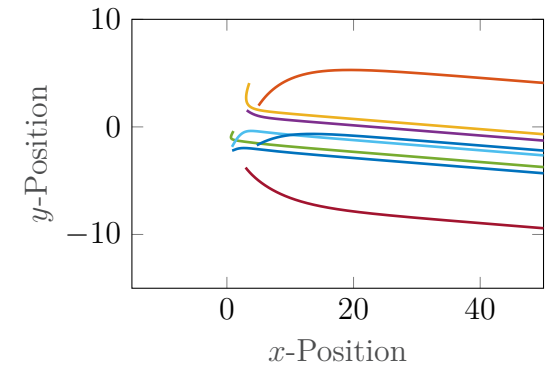
Proposition 3.1 (Conditional consensus emergence)**.** *(cf. [31, Proposition 1]) Let the state of each agent* $(\mathbf{p}_i, \mathbf{v}_i)$ *be such that* $\Gamma(0)$ *and* $\Lambda(0)$ *satisfy*

$$\int_{\sqrt{\Gamma(0)}}^{\infty} a\left(\sqrt{2N}r\right) dr \geq \sqrt{\Lambda(0)}. \tag{3.33}$$

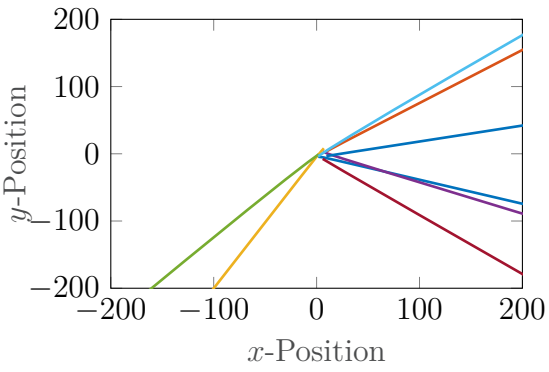
Then, the solution of (3.20) with initial state $(\mathbf{p}_i(0), \mathbf{v}_i(0))$ *tends to consensus.*

It should be noted that this is a sufficient but not necessary condition. This means that consensus will certainly emerge in the system if condition (3.33) is satisfied. However, it is highly strict because there are cases in which the system converges to consensus, despite $\sqrt{\Lambda(0)} > \int_{\sqrt{\Gamma(0)}}^{\infty} a\left(\sqrt{2N}r\right) \mathrm{dr}$.

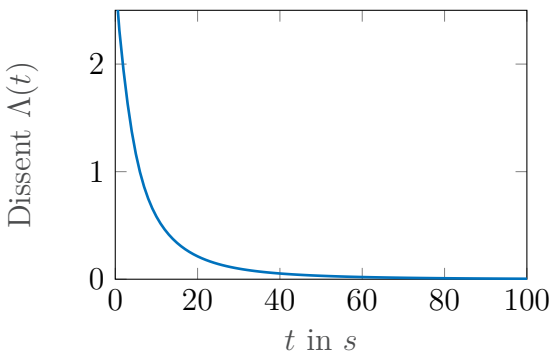
Fig. 3.7 shows simulations with different initial values using the parameters $N = 8$, $H = 1$, $\beta = 0.8$ verify the presented condition for the consensus emergence. It can be seen that consensus emergence depends on the initial states.



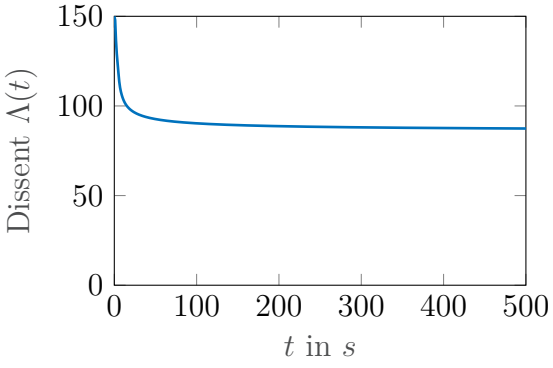
(a) Trajectories of agents fullfilling (3.33).



(b) Trajectories of agents not fullfilling (3.33).



(c) Dissent converges to $\Lambda = 0$.



(d) Dissent does not converge to $\Lambda = 0$.

Figure 3.7: Consensus emergence of the CS model with different initial states.

3.3 The Cucker-Dong Model

This section introduces a particle model, on which the next section is strongly based, and its analysis. In the previous section, we have investigated an alignment model, in which certain states of the agents match after the alignment process is completed. In other words, we extensively considered the agents' velocities in the CS model, which determine the consensus criterion. Hereby, the local confinement and collision freedom of the agents are in the foreground. Local confinement means the restriction of distances of the agents from each other. This property will also be referred to cohesion in this work.

Since the CS model does not guarantee inter-agent collision avoidance, it has been updated by Cucker and Dong [37] with a repulsive term. The CD model also describes the dynamics of $N \in \mathbb{N}$ agents [37, 36].

The dynamics of the CD model are given by the following system of differential equations (cf. [37], p. 1011):

$$\begin{aligned}
\dot{\mathbf{p}}_i(t) &= \mathbf{v}_i(t), \\
\dot{\mathbf{v}}_i(t) &= \sum_{j=1}^{N} a\left(\|\mathbf{p}_j(t) - \mathbf{p}_i(t)\|^2\right)(\mathbf{p}_j(t) - \mathbf{p}_i(t)) + \sum_{j=1}^{N} f\left(\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|^2\right)(\mathbf{p}_i(t) - \mathbf{p}_j(t)),
\end{aligned} \tag{3.34}$$

for $i = 1, \ldots, N$ without damping term. The dynamics consist of the attraction and repulsion terms. The attraction force is defined as

$$a(r) = \frac{H}{(1+r)^{\beta}}, \quad H > 0, \quad \beta \geq 0, \tag{3.35}$$

where H and β are constant parameters and the maximum of the function $a(r)$ is at $a(0)$. For the collision avoidance, a repulsion term of the form

$$f(r) = r^{-s}, \quad s > 1, \tag{3.36}$$

is introduced (see Fig. 3.8). The position differences between agents determine the direction of distance-dependent forces. In addition, the interaction between two agents becomes weaker with increasing distance. The fundamental difference between the CS model and the CD model is that the presence of forces in the CD model depends on the distance between the agents. However, in the CS model, the distance between the agents is only relevant for the interaction strength. Furthermore, the agents in the CD system exchange only information about their positions, but not about their velocities. Therefore, it is reasonable to suppose that the CD system behaves more *chaotically* than the CS system due to the limited information items exchanged among the agents.

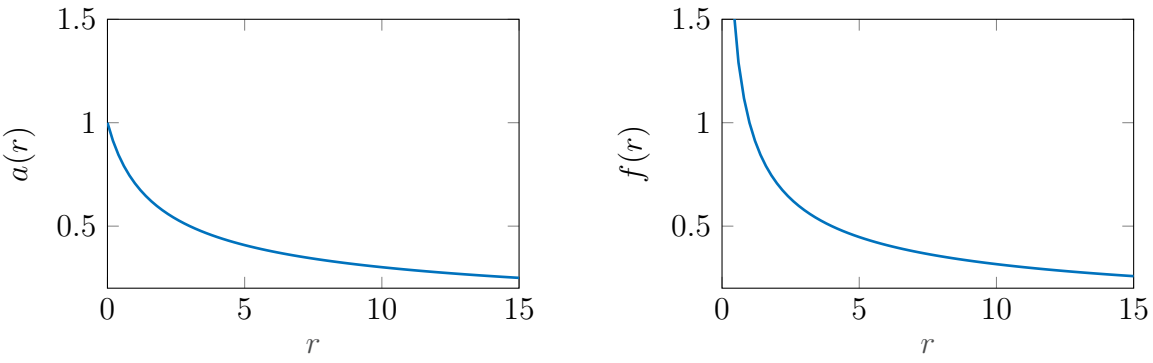


Figure 3.8: Functions of attraction and repulsion terms ($H = 1$, $\beta = 0.5$, $s = 0.5$).

In the CS model, the velocities of the agents are identical after a successful alignment process. Aligned velocities result in a constant distance between the agents. However, the demand for cohesive behavior in the CD system allows variable distances between the agents.

In order to analyze the stability of the CD system, it is beneficial, as in the CS model, to define a property that describes the system on its macro level. For this purpose, we consider the total energy $E(t)$ of the system. This consists of the kinetic energy $E_{kin}(t)$, the potential energy $E_{pot,a}(t)$ due to the attraction forces, and the potential energy $E_{pot,f}(t)$ due to the repulsion forces. Since the integral of the force over the distance range gives the energy, it follows

$$E_{pot,a}(t) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\int_{0}^{\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2} a(r)\mathrm{d}r \tag{3.37}$$

and

$$E_{pot,f}(t) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\int_{\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2}^{\infty} f(r)\mathrm{d}r. \tag{3.38}$$

Furthermore, the kinetic energy of the system can be given by

$$E_{kin}(t) = \sum_{i=1}^{N}\|\mathbf{v}_i(t)\|^2. \tag{3.39}$$

Thus, the total energy of the system (see [37], p. 1011) is defined as follows:

$$E(t) = \underbrace{\sum_{i=1}^{N}\|\mathbf{v}_i(t)\|^2}_{E_{kin}} + \underbrace{\frac{1}{2}\sum_{i,j=1}^{N}\int_{0}^{\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2} a(r)\mathrm{d}r}_{E_{pot,a}} + \underbrace{\frac{1}{2}\sum_{i,j=1}^{N}\int_{\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2}^{\infty} f(r)\mathrm{d}r}_{E_{pot,f}}. \tag{3.40}$$

An important property of (3.34) is that the total energy of the system is constant. We can prove this by considering the derivative of the total energy as

$$\frac{\mathrm{d}}{\mathrm{dt}}E(t) = \frac{\mathrm{d}}{\mathrm{dt}}\sum_{i=1}^{N}\|\mathbf{v}_i(t)\|^2 + \frac{\mathrm{d}}{\mathrm{dt}}\frac{1}{2}\sum_{i,j=1}^{N}\int_0^{\|\mathbf{p}_i-\mathbf{p}_j\|^2} a(r)\mathrm{dr} + \frac{\mathrm{d}}{\mathrm{dt}}\frac{1}{2}\sum_{i,j=1}^{N}\int_{\|\mathbf{p}_i-\mathbf{p}_j\|^2}^{\infty} f(r)\mathrm{dr}. \quad (3.41)$$

For clarity, the terms are derived separately with $r_{ij}^2 = \|\mathbf{p}_i - \mathbf{p}_j\|^2$.

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{dt}}\sum_{i=1}^{N}\|\mathbf{v}_i(t)\|^2 =& 2\sum_{i=1}^{N}\dot{\mathbf{v}}_i(t)\cdot\mathbf{v}_i(t)\\
=& 2\sum_{i,j=1}^{N} a(r_{ij}(t)^2)(\mathbf{p}_j(t)-\mathbf{p}_i(t))\cdot\mathbf{v}_i(t)\\
&+ 2\sum_{i,j=1}^{N} f(r_{ij}(t)^2)(\mathbf{p}_i(t)-\mathbf{p}_j(t))\cdot v_i(t)\\
=& \underbrace{-\sum_{i,j=1}^{N} a(r_{ij}(t)^2)(\mathbf{p}_j(t)-\mathbf{p}_i(t))\cdot(\mathbf{v}_i(t)-\mathbf{v}_j(t))}_{A}\\
&+ \underbrace{\sum_{i,j=1}^{N} f(r_{ij}(t)^2)(\mathbf{p}_i(t)-\mathbf{p}_j(t))\cdot(\mathbf{v}_i(t)-\mathbf{v}_j(t))}_{B}.
\end{aligned} \quad (3.42)$$

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{dt}}E_{pot,a}\left(r(t)\right) &= \frac{\mathrm{d}}{\mathrm{dr}}E_{pot,a}(r)\cdot\frac{\mathrm{d}}{\mathrm{dt}}r(t)\\
&= \frac{1}{2}\sum_{i,j=1}^{N} a(r_{ij}^2)\cdot 2\cdot(\mathbf{p}_j(t)-\mathbf{p}_i(t))\cdot(\mathbf{v}_i(t)-\mathbf{v}_j(t)\\
&= \underbrace{\sum_{i,j=1}^{N} a(r_{ij}^2)(\mathbf{p}_j(t)-\mathbf{p}_i(t))\cdot(\mathbf{v}_i(t)-\mathbf{v}_j(t))}_{C}.
\end{aligned} \quad (3.43)$$

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{dt}}E_{pot,f}\left(r(t)\right) &= \frac{\mathrm{d}}{\mathrm{dr}}E_{pot,f}(r)\cdot\frac{\mathrm{d}}{\mathrm{dt}}r(t)\\
&= \frac{1}{2}\sum_{i,j=1}^{N} f(r_{ij}^2)\cdot 2\cdot(\mathbf{p}_j(t)-\mathbf{p}_i(t))\cdot(\mathbf{v}_i(t)-\mathbf{v}_j(t))\\
&= \underbrace{-\sum_{i,j=1}^{N} f(r_{ij}^2)(\mathbf{p}_i(t)-\mathbf{p}_j(t)\cdot(\mathbf{v}_i(t)-\mathbf{v}_j(t))}_{D}.
\end{aligned} \quad (3.44)$$

If the terms are added together, term A is canceled by term C and term B is canceled by term D. Therefore, the total energy of the system remains constant for $t \geq 0$ and $\frac{\mathrm{d}}{\mathrm{dt}}E(t) = 0$. Thus, $E(t) = E(0)$ holds. Since the stability of the system is considered as a function of energy, it depends only on the initial state of the system.

Fig. 3.9 and Fig. 3.10 show the trajectory of the agents and the corresponding energy function, respectively. The parameters $N = 8$, $H = 1$, $\beta = 0.8$, $s = 2$ are used for the simulation. The initial values were generated with random values; $\mathbf{p}_i(0) \in [-1, 1]$ and $\mathbf{v}_i(0) \in [-1, 1]$.

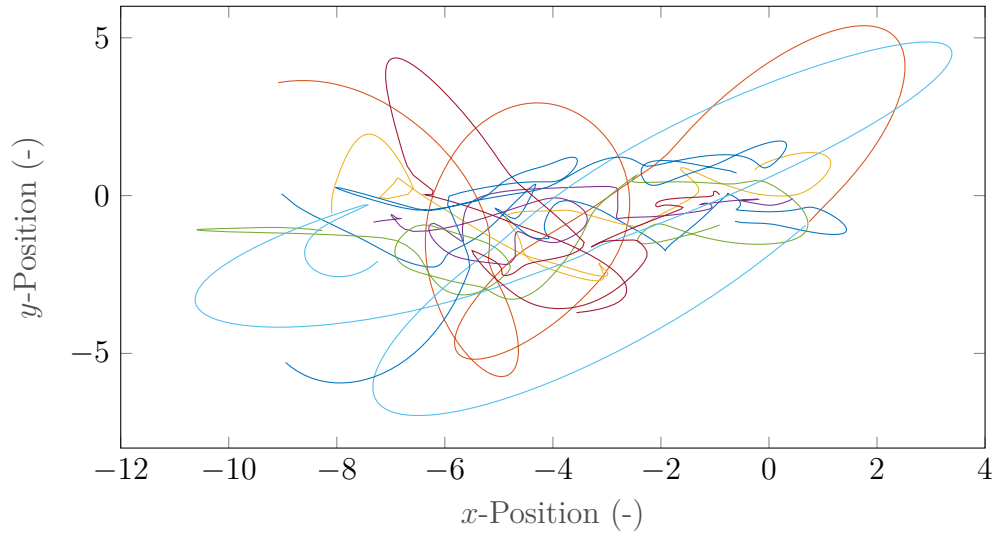


Figure 3.9: Trajectories of the agents.

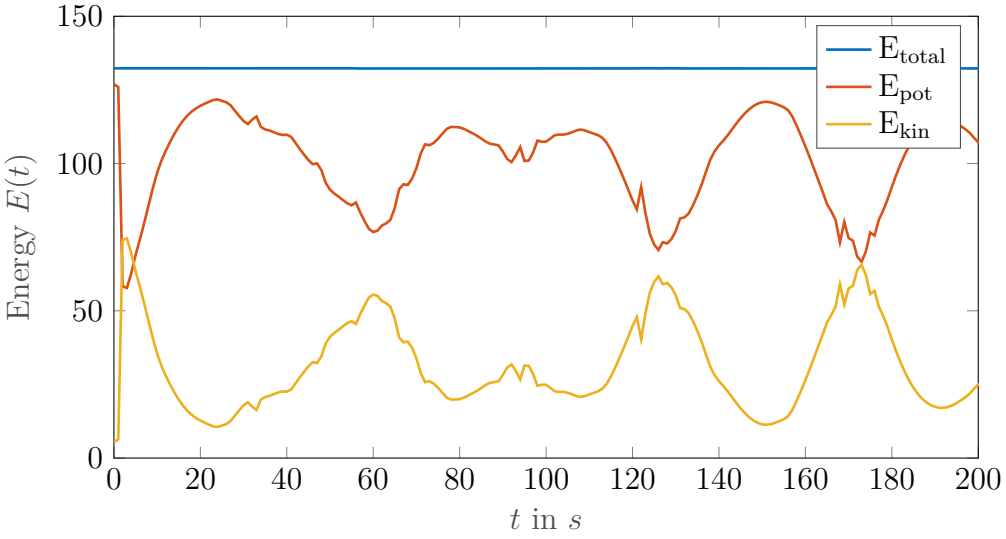


Figure 3.10: Evolution of the total energy, kinetic energy and potential energy.

Obviously, the kinetic energy E_{kin} and the potential energy E_{pot} are always complementary so that the total energy E_{total} remains constant.

3.3.1 Cohesion

The CD model mainly deals with cohesion in a collision avoiding manner. The basic requirement for cohesion is that $\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\| < \infty$ for all pairs $(i,j) \in \{1,\dots,N\}$ and for $0 \leq t < T$. Under specific conditions, cohesion can be guaranteed. The following theorem gives these conditions based on the initial energy of the system (3.34).

Theorem 3.1. *(cf. [23, Theorem 1])*

For any system with N agents defined by (3.34) satisfying $\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|^2 > 0$ for all $i \neq j$ and initial energy

$$E(0) < \frac{1}{2}\int_0^\infty f(r)\,dr,$$

there exists a unique solution[2] $(\mathbf{p}(t), \mathbf{v}(t))$ of the system (3.34) with initial state $(\mathbf{p}(0), \mathbf{v}(0))$. In addition, assume that one of the two following hypotheses holds.

1) $\beta \leq 1$

2) $\beta > 1$ and $E(0) < (N-1)\int_0^\infty a(r)\,dr$

Then, the group is cohesive and collision-avoiding.

Remark 3.1. For more details about Theorem 3.1, the reader is referred to [37, Theorem 2.1]. △

In order to determine under which conditions the system has cohesive behavior, we consider the total energy $E(t)$. With the definition of $a(r)$ in (3.35), $f(r)$ in (3.36) and with the property $E(t) = E(0)$, the following

$$\begin{aligned} E(t) = E(0) &= \frac{1}{2}\sum_{i,j=1}^{N}\int_0^{\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2}\frac{H}{(1+r)^\beta}\mathrm{dr} + \frac{1}{2}\sum_{i,j=1}^{N}\int_{\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2}^{\infty} r^{-s}\mathrm{dr} + \sum_{i=1}^{N}\|v_i(t)\|^2 \\ &\geq \frac{1}{2}\sum_{i,j=1}^{N}\int_0^{\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2}\frac{H}{(1+r)^\beta}\mathrm{dr} \end{aligned} \tag{3.45}$$

holds (see [37], p. 1015). In order to prove these conditions, we assume that there exists a positive constant B_0 such that, for all $t \geq 0$, $\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| < B_0$ with $i \neq j$. In addition, there is an agent q, which has the same distance $\|\mathbf{p}_q(t) - \mathbf{p}_i(t)\| = B_0$ from all other agents $i \in \{1,\dots,N\} \setminus \{q\}$. With (3.45), the total energy can be given as

$$\begin{aligned} E(0) &\geq \sum_{i=1, i\neq q}^{N}\int_0^{\|\mathbf{p}_q(t)-\mathbf{p}_i(t)\|^2}\frac{H}{(1+r)^\beta}\mathrm{dr} \\ &\geq (N-1)\int_0^{B_0^2}\frac{H}{(1+r)^\beta}\mathrm{dr}. \end{aligned} \tag{3.46}$$

[2] $\mathbf{p} = (\mathbf{p}_1^\top,\dots,\mathbf{p}_N^\top)^\top \in \mathbb{R}^{mN}$, $\mathbf{v} = (\mathbf{v}_1^\top,\dots,\mathbf{v}_N^\top)^\top \in \mathbb{R}^{mN}$

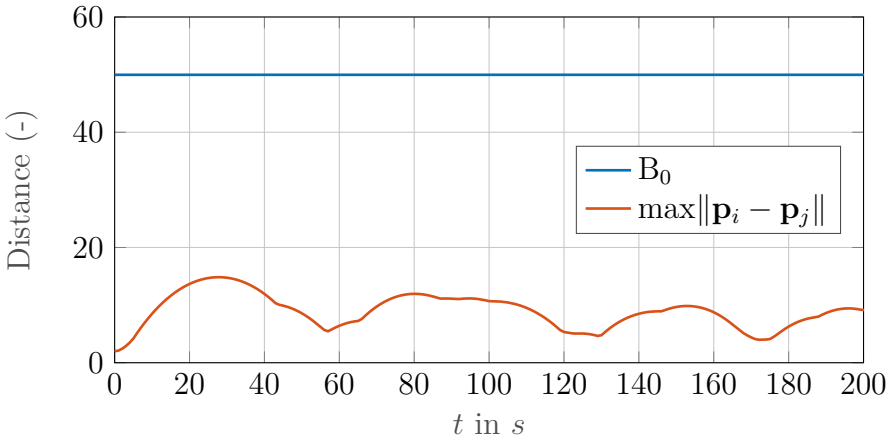


Figure 3.11: Evolution of the maximum distance between two agents for $\beta < 1$.

In order to compute the latter integral, the following cases dependent on β have to be differentiated

$$E(0) \geq \begin{cases} \frac{H(N-1)}{(1-\beta)} \left((1 + B_0^2)^{1-\beta} - 1 \right) & \text{for } \beta \neq 1 \\ H(N-1) log(1 + B_0^2) & \text{for } \beta = 1. \end{cases} \tag{3.47}$$

It follows from (3.47) that the upper bound B_0 for the possible distance between two agents $\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|$ can be defined as

$$B_0 \leq \begin{cases} \left(\left(\frac{(1-\beta)}{H(N-1)} E(0) + 1 \right)^{\frac{1}{1-\beta}} - 1 \right)^{\frac{1}{2}} & \text{for } \beta \neq 1, \\ \left(exp\left(\frac{1}{H(N-1)} E(0) \right) - 1 \right)^{\frac{1}{2}} & \text{for } \beta = 1. \end{cases} \tag{3.48}$$

A further case differentiation is made to corroborate the conditions in Theorem 3.1.

1) $\beta \leq 1$:

   Assume that $\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|$ is restricted. It should hold that $B_0 < \infty$. Looking at the right side of the inequalities in (3.48), obviously, $B_0 = \infty$ is only possible if $E(0) = \infty$. In order to guarantee cohesion for all pairs $(i, j) \in \{1, \ldots, N\}$ for $0 \leq t < T$, the energy of the system has to satisfy $E(0) < \infty$ because $\int_0^\infty f(r)\mathrm{d}r < \infty$.

   For example, using the parameters $N = 8$, $H = 1$, $\beta = 0.8$, $s = 2$ in (3.48) results in an upper boundary of $B_0 = 49.9$ for the distance between two agents. The simulation verifies that this value is never exceeded (Fig. 3.11). The trajectories of the agents are the same as in Fig. 3.9 due to the identical parameters and initial states.

2) $\beta > 1$:

In order to ensure cohesion for the case $\beta > 1$, we consider the inequality (3.48) and rewrite it as

$$B_0^2 \leq \left(\frac{(1-\beta)}{H(N-1)} E(0) + 1 \right)^{\frac{1}{2(1-\beta)}} - 1 \tag{3.49}$$

$$\leq \left(1 - \frac{\beta - 1}{H(N-1)} E(0) \right)^{-\frac{1}{2(\beta-1)}} - 1. \tag{3.50}$$

Since the exponent of the first term in (3.50) is always negative, the base is not allowed to be zero. Otherwise, it would hold $B_0^2 = \infty$. Thus, the following inequality

$$1 - \frac{\beta - 1}{H(N-1)} E(0) > 0$$

has to be true. This results in the following condition for cohesion:

$$E(0) < \frac{H(N-1)}{\beta - 1} =: \vartheta, \tag{3.51}$$

where ϑ is called the critical energy threshold (cf. [37, p. 1015], [23, p. 197]). The simulation with the parameters $N = 8$, $H = 1$, $\beta = 1.2$, $s = 2$ demonstrates that the maximum distance between two agents continuously increases and is therefore unbounded (Fig. 3.12). The noncohesive behavior is also remarkable in the trajectories of agents (Fig. 3.13).

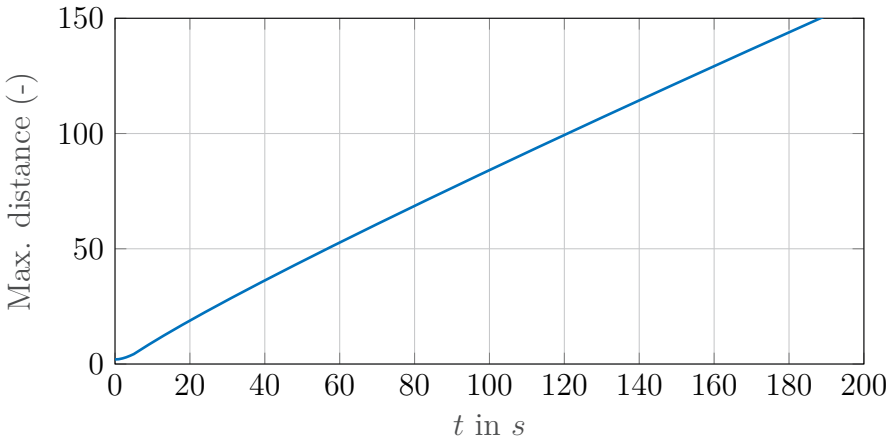


Figure 3.12: Evolution of the maximum distance between two agents for $\beta > 1$ and $E(0) > \vartheta$.

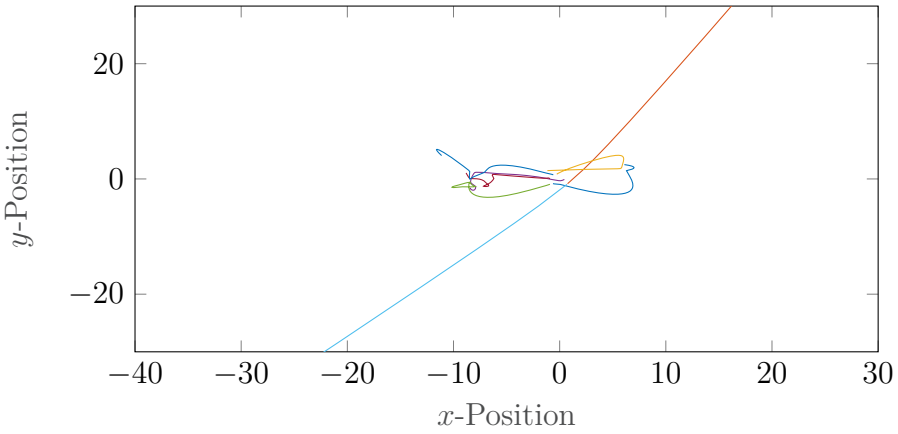


Figure 3.13: Trajectories of the agents for $\beta > 1$ and $E(0) > \vartheta$.

3.3.2 Collision Avoidance

Another property of the CD model is collision avoidance. A system defined by (3.34) is collision-free if $\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\| > 0$ for all pairs $(i, j) \in \{1, \dots, N\}$ for $0 \leq t < T$. In order to investigate a condition for this, we first assume that two agents z and q collide and it holds $\|\mathbf{p}_z(t) - \mathbf{p}_q(t)\|^2 = 0$. All other agents $(i, j) \in \{1, \dots, N\} \setminus \{z, q\}$ fulfill $\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|^2 > 0$. The initial energy of the system in this case can be given by

$$\begin{aligned} E(0) = &\frac{1}{2}\sum_{i,j=1}^{N}\int_0^{\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2} a(r)\mathrm{d}r + \frac{1}{2}\sum_{i,j=1}^{N}\int_{\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2}^{\infty} f(r)\mathrm{d}r + \sum_{i=1}^{N}\|\mathbf{v}_i(t)\|^2 \\ \geq &\frac{1}{2}\int_0^{\|\mathbf{p}_z(t)-\mathbf{p}_q(t)\|^2} a(r)\mathrm{d}r + \frac{1}{2}\int_0^{\|\mathbf{p}_q(t)-\mathbf{p}_z(t)\|^2} a(r)\mathrm{d}r \\ &+ \frac{1}{2}\int_{\|\mathbf{p}_z(t)-\mathbf{p}_q(t)\|^2}^{\infty} f(r)\mathrm{d}r + \frac{1}{2}\int_{\|\mathbf{p}_q(t)-\mathbf{p}_z(t)\|^2}^{\infty} f(r)\mathrm{d}r \\ \geq &\int_0^{\infty} f(r)\mathrm{d}r \\ = &\infty. \end{aligned} \tag{3.52}$$

This proves that a collision between two agents can only occur if the initial energy of the system is infinite, which contradicts the assumption of energy boundedness.

In order to determine the possible minimum distance between two agents $\|\mathbf{p}_z(t) - \mathbf{p}_q(t)\| = d_0$, we formulate the following inequality:

$$\begin{aligned} E(0) &\leq \frac{1}{2}\int_{\|\mathbf{p}_z-\mathbf{p}_q\|^2}^{\infty} f(r)\mathrm{d}r + \frac{1}{2}\int_{\|\mathbf{p}_q-\mathbf{p}_z\|^2}^{\infty} f(r)\mathrm{d}r \\ &\leq \int_{d_0^2}^{\infty} f(r)\mathrm{d}r. \end{aligned} \tag{3.53}$$

Solving the integral yields

$$d_0 \geq ((s-1)E(0))^{\frac{1}{2(1-s)}} . \tag{3.54}$$

For the parameters $N = 8$, $H = 1$, $\beta = 0.8$ or $\beta = 1.2$, $s = 2$, the minimum possible distance between two agents is $d_0 \approx 0.0894$, which is independent of the value of the parameter β (Fig. 3.14).

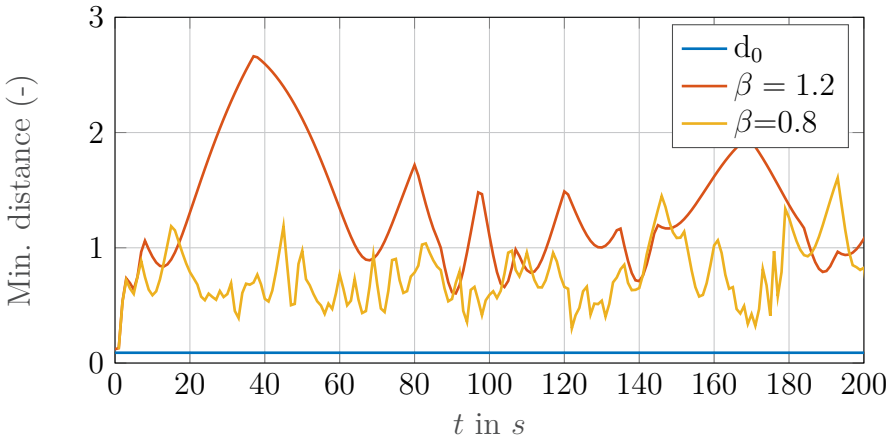


Figure 3.14: Minimum distance between two agents for $\beta = 0.8$ and $\beta = 1.2$.

3.4 Leader-Following Cucker-Dong Model

In this section, we introduce a virtual leader to the CD model to enable the agents trajectory tracking skills. The virtual leader is not a physical entity, but the state expression of an imaginary object moving along a desired trajectory viewed by all agents. The state of the virtual leader is $(\mathbf{p}_r, \mathbf{v}_r) \in \mathbb{R}^m \times \mathbb{R}^m$ and its dynamics are modeled as follows:

$$\begin{aligned} \dot{\mathbf{p}}_r(t) &= \mathbf{v}_r(t), \\ \dot{\mathbf{v}}_r(t) &= f_r(\mathbf{p}_r, \mathbf{v}_r), \end{aligned} \tag{3.55}$$

with $(\mathbf{p}_r(0), \mathbf{v}_r(0)) = (\mathbf{p}_0, \mathbf{v}_0)$. For the *rendezvous* problem, we aim to utilize the cohesion and collision avoidance properties of the CD model. As mentioned in the previous section, the CD model can guarantee cohesion only under certain initial conditions. If these are not satisfied, a feedback controller is required to ensure cohesion. However, if a virtual leader exists, cohesion is naturally ensured for $0 \leq t < T$ because all agents are

attracted from a reference point. In the extended CD system, each agent is actuated and has the following dynamics:

$$\begin{aligned}\dot{\mathbf{p}}_i &= \mathbf{v}_i,\\ \dot{\mathbf{v}}_i &= \sum_{j=1}^{N} a\left(\|\mathbf{p}_j(t)-\mathbf{p}_i(t)\|^2\right)(\mathbf{p}_j(t)-\mathbf{p}_i(t)) + \sum_{j=1}^{N} f\left(\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2\right)(\mathbf{p}_i(t)-\mathbf{p}_j(t))\\ &\quad - b\,\mathbf{v}_i + \mathbf{u}_i^{\beta} + \mathbf{u}_i^{\gamma},\end{aligned} \tag{3.56}$$

where $b\,\mathbf{v}_i$ is a damping term, $\mathbf{u}_i^{\beta}$ and $\mathbf{u}_i^{\gamma}$ are control inputs for obstacle avoidance and tracking term, respectively.

3.4.1 Collision Avoidance in the Leader-Following CD Model

One distinctive property of the CD model is the guaranteed inter-agent collision avoidance, which is mentioned in Section 3.3. In order to examine if collision avoidance is guaranteed also in the existence of the aforementioned new control inputs, we assume that two agents z and q collide and $\|\mathbf{p}_z(t)-\mathbf{p}_q(t)\|^2 = 0$. For all other pairs $(i,j) \in \{1,\dots,N\}\setminus\{z,q\}$, $\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2 > 0$ holds. Note that the control inputs are arbitrary and bounded. The total energy of the system can be explicitly described similarly to (3.40) as follows:

$$\begin{aligned}E(t) = &\ \frac{1}{2}\sum_{i,j=1}^{N}\int_0^{\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2} a(r)\mathrm{d}r\\ &+ \frac{1}{2}\sum_{i,j=1}^{N}\int_{\|\mathbf{p}_i(t)-\mathbf{p}_j(t)\|^2}^{\infty} f(r)\mathrm{d}r + \sum_{i=1}^{N}\|\mathbf{v}_i(t)\|^2\\ &+ E_\beta(t) + E_\gamma(t)\\ &\geq \frac{1}{2}\int_0^{\|\mathbf{p}_z(t)-\mathbf{p}_q(t)\|^2} a(r)\mathrm{d}r + \frac{1}{2}\int_0^{\|\mathbf{p}_q(t)-\mathbf{p}_z(t)\|^2} a(r)\mathrm{d}r\\ &+ \frac{1}{2}\int_{\|\mathbf{p}_z(t)-\mathbf{p}_q(t)\|^2}^{\infty} f(r)\mathrm{d}r + \frac{1}{2}\int_{\|\mathbf{p}_q(t)-\mathbf{p}_z(t)\|^2}^{\infty} f(r)\mathrm{d}r\\ &+ E_\beta^{zq}(t) + E_\gamma^{zq}(t)\\ &\geq \underbrace{\int_0^{\infty} f(r)\mathrm{d}r}_{\infty} + E_\beta^{zq}(t) + E_\gamma^{zq}(t) = \infty,\end{aligned} \tag{3.57}$$

where E_β and E_γ are energies explicitly through $\mathbf{u}_i^{\beta}$ and $\mathbf{u}_i^{\gamma}$, respectively. E_β^{zq} and E_γ^{zq} are partial energies through the agent pair (z,q).

Based on (3.57), the collision of two agents (z, q) can only occur if the energy of the system is infinite. Therefore, $E(0) < \infty$ is the only requirement for collision-free navigation. Since the control inputs $\mathbf{u}_i^\beta$ and $\mathbf{u}_i^\gamma$ are assumed to be bounded, $E(0) < \infty$ is satisfied in any case. As it is shown, additional control inputs do not influence the inter-agent collision avoidance property of the CD model.

3.4.2 Reactive Control for Obstacle Avoidance

For the reactive control, we assume that every agent can measure the relative position between the closest point on an obstacle and itself. In this section, only two kinds of obstacles are considered, circular and convex polygon-shaped obstacles (see Fig. 3.15). The main idea is based on [100] explained in Section 2.1.3 and the high-level workflow is briefly itemized in the following:

1. Determining if an obstacle is within the sensing range r_s of an agent.
2. Creating a virtual β-agent that is a projection of the agent onto the edge of the obstacle.
3. Adding a control term $\mathbf{u}_i^\beta$ such that the agent does not collide with the β-agent.

In the simulation environment, β-agents on polygon edges are calculated using (2.10) and β-agents on circular obstacles are created by means of (2.12). If a vertex point is detected, it is treated as a circular obstacle but with a small imaginary radius (e.g., $R_k \leq 0.01$). In a later chapter in this thesis, we will revisit the vertex point detection (endpoint detection) for practical implementation.

For the collision avoidance with obstacles, the control term $\mathbf{u}_i^\beta$ is defined similarly to the repulsive component of the CD dynamics and it is given by

$$u_i^\beta = \sum_{j=1}^{N} f_\beta \left(\|\mathbf{p}_i(t) - \hat{\mathbf{p}}_{i,k}(t)\|^2 \right) \left(\mathbf{p}_i(t) - \hat{\mathbf{p}}_{i,k}(t) \right), \tag{3.58}$$

with

$$f_\beta(r) = (r - \delta)^{-s_\beta}, \quad s_\beta > 1.$$

The repulsive force generated by $f_\beta(r)$ helps agents to keep a certain distance δ to the detected obstacle.

3.4.3 Navigational Feedback

In order to define a collective group objective for the flock, we introduce a virtual leader called the γ-agent. The γ-agent is not considered by the agents to be a physical agent.

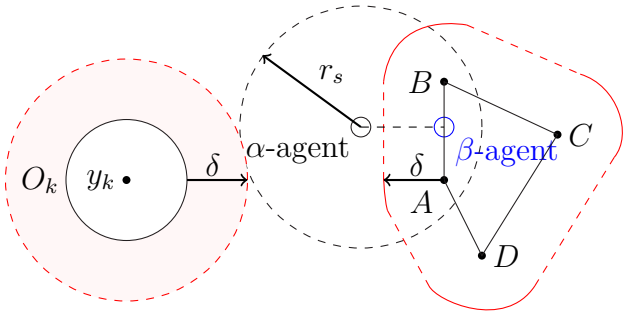


Figure 3.15: Illustration of a circular and a polygon-shaped obstacle with the virtual β-agent.

However, it is possible to include a physical γ-agent into the flock. The dynamics of the virtual agent are given by

$$\dot{\mathbf{p}}_r(t) = \mathbf{v}_r(t),$$
$$\dot{\mathbf{v}}_r(t) = f_r(\mathbf{p}_r, \mathbf{v}_r),$$

where $(\mathbf{p}_r, \mathbf{v}_r) \in \mathbb{R}^m \times \mathbb{R}^m$ represent the position and velocity of the virtual target, respectively. Moreover, the virtual leader can also be a static target point ($\mathbf{v}_r = \mathbf{0}$). The control input for tracking is defined as follows

$$u_i^\gamma = -c_1(\mathbf{p}_i(t) - \mathbf{p}_r(t)) - c_2(\mathbf{v}_i(t) - \mathbf{v}_r(t)), \quad c_1, c_2 > 0. \tag{3.59}$$

This term guarantees cohesive behavior by attracting the agents to a dynamic rendezvous point. The constant parameters c_1 and c_2 in (3.59) are weighting factors to influence the strength of the tracking feedback.

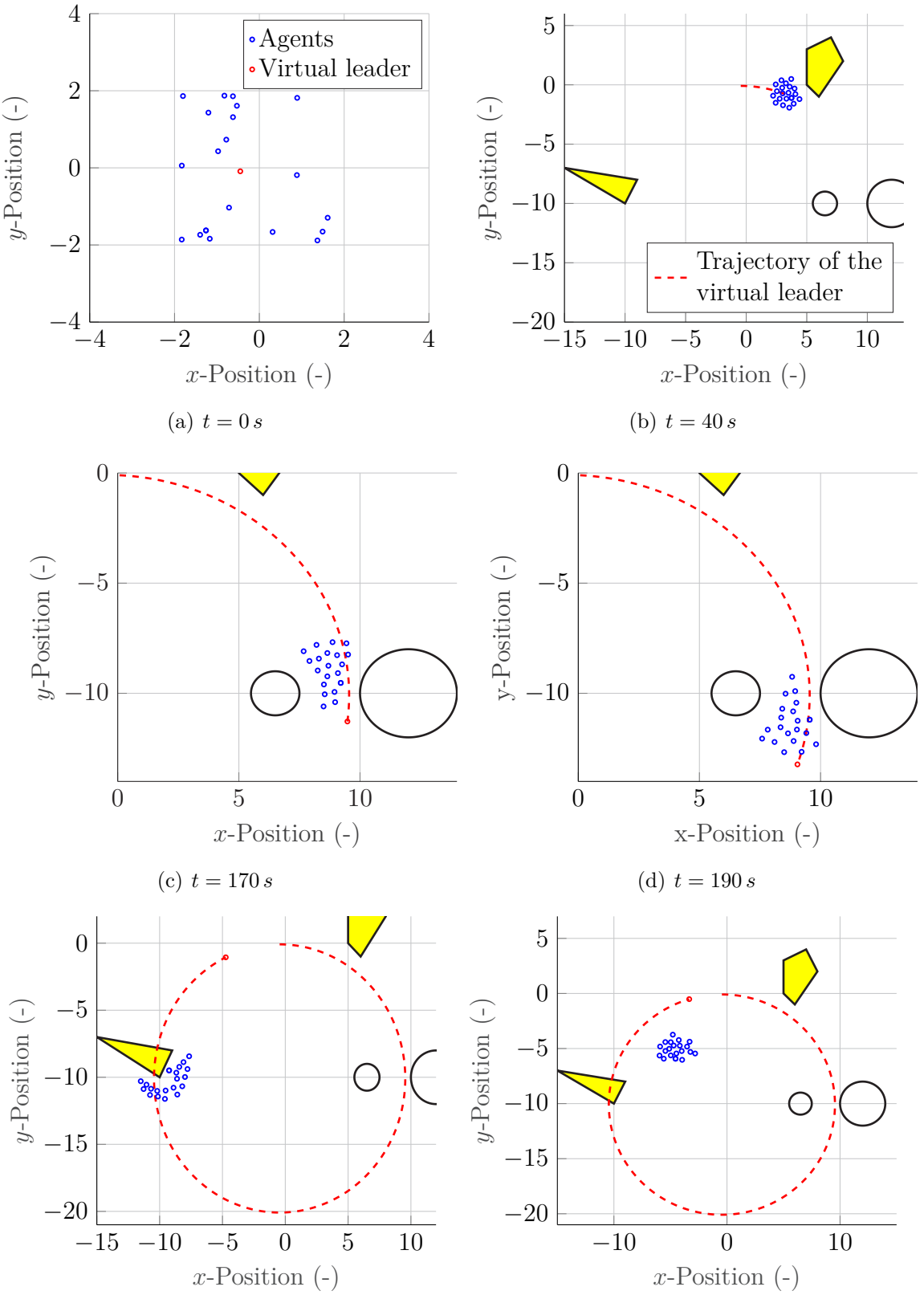
3.5 Numerical Examples

In this section, we present results of numerical tests with the proposed control scheme. In the first simulation, the virtual leader moves along a circular trajectory with a constant speed of $\|\mathbf{v}_r\| = 1$. We consider a system of $N = 20$ agents in the two-dimensional plane $m = 2$. The agents are randomly placed in the box $[-2, 2] \times [-2, 2]$ and initial velocities $\mathbf{v}_i(0) \in \mathbb{R}^2$ are randomly selected from $[-1, 1] \times [-1, 1]$, as shown in Fig. 3.16(a). The initial position of the virtual leader is the center point of the flock. The parameters for the simulation are: $H = 1$, $s = 2$, $\beta = 0.5$, $b = 0.05$, $c_1 = c_2 = 1$, $s_\beta = 1$, $\delta = 1$, $r_s = 1.5$. Fig. 3.16(b) shows that the group tracks the virtual leader. In addition, two closely located circular obstacles can be avoided by preserving inter-agent collision avoidance (Fig. 3.16(c)-Fig. 3.16(d)).

In Fig. 3.16(e), it can be seen that almost all agents maintain a specific minimum distance to the polygon-shaped obstacle. However, one agent is closer to the obstacle.

The reason might be that the force through navigation and the interaction forces among other agents are higher than the repulsion force of the β-agent at that time instant. Fig. 3.16(e) demonstrates that the group can avoid the triangle obstacle by preserving their cohesion. Moreover, leaving the obstacle behind, the group quickly returns to the virtual leader (Fig. 3.16(f)).

In the second simulation, the initial configuration of agents is the same as in the first simulation and the virtual leader moves along a sinusoidal trajectory with a constant speed of $\|\mathbf{v}_r\| = 1$. Fig. 3.17(a) shows again that the group can avoid the triangle obstacle by preserving their cohesion. Due to the all-to-all connection of agents in the CD system, the group has a high tendency to cohesion (Fig. 3.17(b)).

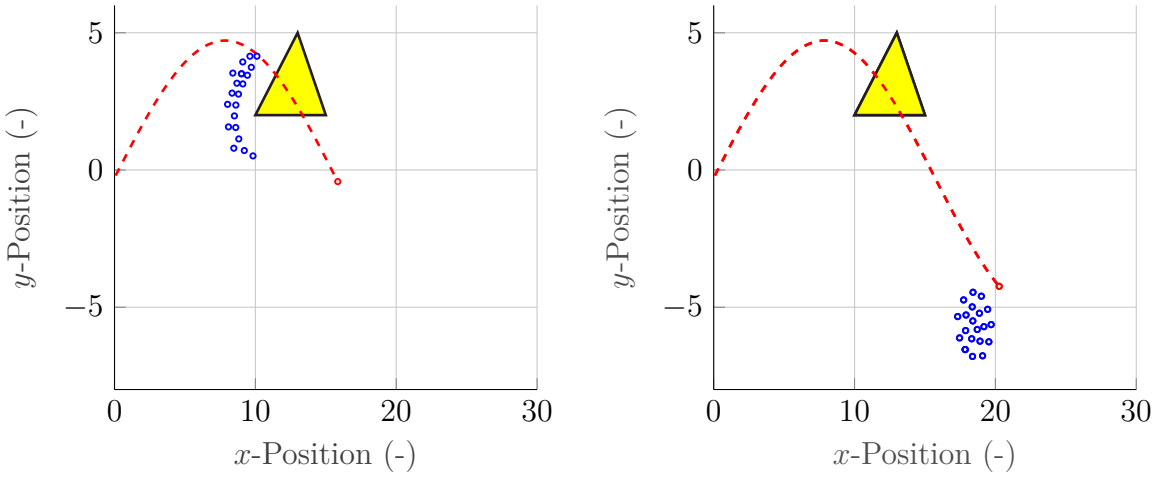


(a) $t = 0\,s$ (b) $t = 40\,s$

(c) $t = 170\,s$ (d) $t = 190\,s$

(e) $t = 585\,s$ (f) $t = 600\,s$

Figure 3.16: CD flocking of $N = 20$ agents - Tracking a circular trajectory.



(a) $t = 150\,s$ (b) $t = 190\,s$

Figure 3.17: CD flocking of $N = 20$ agents - Tracking a sinusoidal trajectory.

3.6 Chapter Highlights

In this chapter, we extended the Cucker-Dong model for a particle system to build a target-tracking and an obstacle-avoiding flock. This extension includes a common group objective and an obstacle avoidance algorithm to work with convex obstacles. In order to do this, we introduced two control inputs. The leader itself is called the γ-agent and moves along a predefined trajectory. The obstacle-avoiding reactive control term is designed similarly to the repulsive term in CD dynamics. This repelling force is activated once an obstacle is detected by an agent.

A property of the CD flock is that it usually does not allow fragmentation due to a high tendency to cohesion. Moreover, with increasing distance between two agents, the interaction forces converge to zero. Thus, if an agent has a long distance to the rest of the group due to fragmentation during obstacle avoidance, it might be hard to catch up with the rest of the group because the interaction strength between the agent and the rest of the group decreases.

Since we considered only convex obstacles, collisions can still occur under some obstacle configurations e.g., if most agents perpendicularly move to the edge of a polygon-shaped obstacle, which yields balance among virtual forces. Such complex cases are investigated in later chapters. According to our experience, a group with more agents is beneficial because of high cohesive energy.

The supervised student thesis [165] has contributed to the development of this chapter's results.

4 Potential Function-Based Control Scheme

Parts of the following chapter have been published in [155].

Cooperative, multiple agents are employed for many different tasks to increase the efficiency and success of a mission. The navigation of autonomous, mobile multi-robot systems in changing environments is a challenging problem that has been investigated in recent years. However, many of the existing collective path planning approaches do not guarantee a reliable escape in environments with complex, non-convex obstacles without any prior knowledge. Local motion planning approaches are usually suitable for these kinds of problems. The robots can plan their actions autonomously using sensor measurements and react to their environment. The proposed approach in this chapter falls into this category.

The outline of this chapter is structured as follows: In Section 4.1, the potential fields are briefly introduced. In Section 4.2, we dive into the details of the modeling of flocking behavior using artificial potential fields. This section includes the modeling of interactions among agents and navigation. In addition, the section provides analysis of collective dynamics, which will be helpful in the next chapter. Section 4.3 introduces the obstacle avoidance mechanism, which is also employed in Chapter 5 and Chapter 6. Section 4.4 describes the issue of local minima. In Section 4.5, we propose a heuristic algorithm to preview local minima and escape from them. This section is based on [155], a contribution by the author of this thesis. Section 4.6 illustrates the effectiveness of the presented approach through simulation scenarios.

4.1 Potential Fields

Potential field methods have been widely applied for the cooperative control of multiple robots, especially for collision avoidance due to its simplicity. Interactions between the agents and reactions of agents to obstacles can be modeled by means of the artificial potential field method. This approach defines the potential energy as a function of distance $U : \mathbb{R}^n \to \mathbb{R}_{\leq 0}$ to obstacles, targets or to another agent. The agents receive artificial attractive-repulsive forces by following the negative gradient of such potential functions $(-\nabla U)$, so that the energy is minimized.

In local path planning approaches with limited perception capacity of agents, the artificial forces are activated only if an obstacle or another agent is within the the sensing range. Two agents would attract each other until they have the minimum distance d. However, if the distance falls below d, a virtual repulsion force should ensure that the agents do not collide with each other and instead return to their α-lattice configuration. In the case of obstacle detection, the agent should only receive artificial repulsive forces from the detected object.

Such requirements can be fulfilled by means of potential field functions. The best known example for potential fields is the Earth's gravitational field, which has the gravitational potential as its potential function. The artificial potential field approaches usually suffer from the problem of local minima, which will be explained later in Section 4.4.

4.2 Flocking using Artificial Potential Fields

In this section, the dynamics of the agents[1] are formulated. For path planning, the agents are considered as point masses and their dynamics are modeled based on potential functions. The robots receive virtual attractive and repulsive forces by following the gradient of a potential field. For path planning, we consider only the position and velocity of each robot i in an m-dimensional space, $(\mathbf{p}_i, \mathbf{v}_i) \in \mathbb{R}^m \times \mathbb{R}^m$. The dynamics of each mobile agent i are described using the following system of differential equations:

$$\dot{\mathbf{p}}_i = \mathbf{v}_i, \tag{4.1}$$

$$\dot{\mathbf{v}}_i = \mathbf{u}_i, \tag{4.2}$$

where $\mathbf{u}_i \in \mathbb{R}^m$ is the control input. For the time-discrete implementation, however, the dynamics of the system can be given as

$$\begin{aligned} \mathbf{p}(t_k + 1) &= \mathbf{p}_i(t_k) + \mathbf{v}_i \Delta t, \\ \mathbf{v}(t_k + 1) &= \mathbf{v}_i(t_k) + \mathbf{u}_i \Delta t, \end{aligned} \tag{4.3}$$

where Δt is a finite step size and t_k is the k-th time step. Since the agents have a limited communication radius r_c, the neighborhood of an agent is defined as

$$\mathcal{N}_i^{\alpha} = \{j \in \mathcal{V} : \|\mathbf{p}_j - \mathbf{p}_i\| < r_c\}, \tag{4.4}$$

where $\mathcal{V} = \{1, \ldots, n\}, n \in \mathbb{N}$, is a set of all agents. $\mathcal{N}_i^{\alpha}$ is an undirected, time-varying network of neighboring agents. Thus, the communication among robots in this set is bidirectional.

[1] The agents can be seen as holonomic vehicles.

In order to establish a flock-like behavior, a deviation energy is generated as a function depending on the distance between two agents. For the differentiability of this function at $z = 0$, a mapping, the so-called σ-norm, is used as follows [100]:

$$\|z\|_\sigma = \frac{1}{\varepsilon}[\sqrt{1 + \varepsilon\|z\|^2} - 1], \tag{4.5}$$

with a fixed parameter $\varepsilon > 0$. For a smooth transition between 0 and 1 in the definition of *spatially-weighted* neighborhood matrix and for the interaction forces, we use the bump function defined as:

$$\rho_h(z) = \begin{cases} 1, & z \in [0, h) \\ \frac{1}{2}[1 + \cos(\pi\frac{(z-h)}{(1-h)})], & z \in [h, 1) \\ 0, & \text{otherwise} \end{cases} \tag{4.6}$$

where $h \in (0, 1)$. With the help of (4.6), we can define the adjacency matrix $\mathbf{A} = [a_{ij}]$ of the network $\mathcal{N}_i^\alpha$ as follows:

$$a_{ij}(\mathbf{p}) = \rho_h\left(\frac{\|\mathbf{p}_j - \mathbf{p}_i\|_\sigma}{r_\alpha}\right) \in [0, 1], \qquad j \neq i$$

with $r_\alpha = \|r_c\|_\sigma$. In this thesis, we aim to utilize the benefits of flocking behavior (cohesion, separation and alignment properties). Due to the ease of analysis and robustness in terms of collective behavior, the dynamics of the multi-agent system are described similarly to those in the flocking algorithm in [100]. The control input is given by the following dynamics:

$$\mathbf{u}_i = \mathbf{u}_i^\alpha + \mathbf{u}_i^\beta + \mathbf{u}_i^\gamma, \tag{4.7}$$

where $\mathbf{u}_i^\alpha$ is the control input for *flock-like* behavior, $\mathbf{u}_i^\beta$ is for the *obstacle avoidance* and $\mathbf{u}_i^\gamma$ is for the *navigation*. These control inputs will be explained in detail later on.

4.2.1 (α, α)-Interaction

The collective behavior without navigation and obstacle avoidance was described by [100] and it was called *free-flocking*. The main objective of this was to keep α-agents together in a flock, avoid inter-agent collisions, and form an α-lattice structure.

The control input $\mathbf{u}_i^\alpha$ of each agent is given by

$$\mathbf{u}_i^\alpha = \underbrace{c_1^\alpha \sum_{j \in \mathcal{N}_i^\alpha} \varphi_\alpha(\|\mathbf{p}_j - \mathbf{p}_i\|_\sigma)\mathbf{n}_{ij}}_{\text{Gradient-based term}} + \underbrace{c_2^\alpha \sum_{j \in \mathcal{N}_i^\alpha} a_{ij}(\mathbf{p})(\mathbf{v}_j - \mathbf{v}_i)}_{\text{Consensus term}}, \tag{4.8}$$

where $\mathbf{n}_{ij} = \sigma_\varepsilon(\mathbf{p}_j - \mathbf{p}_i) = \dfrac{\mathbf{p}_j - \mathbf{p}_i}{\sqrt{1+\varepsilon\|\mathbf{p}_j - \mathbf{p}_i\|^2}}$ a vector from $\mathbf{p}_j$ to $\mathbf{p}_i$ and $\varepsilon \in (0,1)$ a constant of the σ-norm. The parameters c_1^α, c_2^α are positive constants.

The control input describes the interaction among the α-agents. The first term in (4.8), the so-called *gradient-based term*, is responsible for the position control of the agent i. The second term represents the *consensus term* that may act as artificial damping force between the agents to align their velocities. The necessary potential function for the gradient-based term has been formulated in Section 2.2.

Remark 4.1. The control input $\mathbf{u}_i^\alpha$ fulfills all Reynolds rules. However, the flocking behavior strongly depends on the initial states of the agents. Solely using this control input is known as *(α,α)-protocol of flocking* [100]. Since this includes no group objective, the protocol fails to work for most initial states and fragmentation occurs. △

4.2.2 Navigation

Introducing a group objective, such as a common target position, the so-called γ-agent, makes all agents move toward the same position. In this way, they come closer and this increases the chance that they get into communication range of each other. Thus, the group can build an α-lattice structure more easily. The control term for the *navigation* is defined similarly to that in Section 3.4.3 as follows

$$\begin{aligned}\mathbf{u}_i^\gamma &:= \mathbf{f}_i^\gamma(\mathbf{p}_i, \mathbf{v}_i, \mathbf{p}_d, \mathbf{v}_d) \\ &= -c_1^\gamma \sigma_1(\mathbf{p}_i - \mathbf{p}_d) - c_2^\gamma(\mathbf{v}_i - \mathbf{v}_d), \qquad c_1, c_2 > 0\end{aligned} \tag{4.9}$$

where $(\mathbf{p}_d, \mathbf{v}_d) \in \mathbb{R}^2 \times \mathbb{R}^2$ represents the *desired* goal position and the *desired* velocity, respectively.

Remark 4.2. In order to track a target point moving sinusoidally with high speed and a large amplitude, high values for c_1^γ, c_2^γ are necessary, but this can be at the expense of the α-lattice form. The reason for this is that the navigation input can be stronger than the (α,α)-interaction and exceed its priority, although navigation is usually a secondary goal in the navigation of flocking systems. △

4.2.3 Collective Dynamics

With the help of the mathematical tools presented in Section 2.3, the collective dynamics of the flock can be formulated as follows:

$$\dot{\mathbf{p}} = \mathbf{v} \tag{4.10}$$

$$\dot{\mathbf{v}} = -\nabla V(\mathbf{p}) - \hat{\mathbf{L}}(\mathbf{p})\mathbf{v} + \mathbf{f}_\gamma(\mathbf{p}, \mathbf{v}, \mathbf{p}_d, \mathbf{v}_d) \tag{4.11}$$

where $V(\mathbf{p})$ is a smooth collective potential function given in Eq. (2.21), which depends on the positional configuration of the agents. The gradient term in (4.11) represents

the collective gradient-based term (cf. (4.8)). The second term with $\hat{\mathbf{L}}(\mathbf{p})$ is the nm-dimensional Laplacian matrix of the graph $G(\mathbf{p})$, which is constructed according to Eq. (2.27). Together with the velocity configuration $\mathbf{v}$ of the flocking system, the Laplacian term allows to describe the collective consensus term and the kinetic energy of the system. The third term $\mathbf{f}_\gamma(\mathbf{p}, \mathbf{v}, \mathbf{p}_r, \mathbf{v}_r)$ corresponds to the collective navigational feedback, which will be described in detail later. With $\mathbf{f}_\gamma \equiv \mathbf{0}$, the Hamiltonian function describing the energy of the system ((4.10)-(4.11)) is given by

$$H(\mathbf{p}, \mathbf{v}) = V(\mathbf{p}) + \sum_{i=1}^{n} \|\mathbf{v}_i\|^2, \tag{4.12}$$

where the first term and the second term represent the potential and the kinetic energy of the system, respectively. Based on the property of positive semi-definiteness introduced in Eq. (2.26) and the square sum formulation (2.28) of the nm-dimensional Laplacian matrix, the change in energy

$$\dot{H} = -\mathbf{v}^\mathsf{T} \hat{\mathbf{L}}(\mathbf{p}) \mathbf{v} \leq 0 \tag{4.13}$$

is negative and thus, the system must be dissipative.[2]

The present states $(\mathbf{p}, \mathbf{v})$ do not guarantee the boundedness of the solution, especially if the flock disperses. For the stability analysis of the system described by (4.10) and (4.11), defining new suitable states is the key to success so that we can employ LaSalle's invariance principle.

For this purpose, study [100] proposes to split the dynamics of the system into translational and structural dynamics. These are described using *moving* reference states defined as

$$\begin{aligned} \tilde{\mathbf{p}}_i &= \mathbf{p}_i - \mathbf{p}_c \\ \tilde{\mathbf{v}}_i &= \mathbf{v}_i - \mathbf{v}_c \end{aligned} \tag{4.14}$$

where $\mathbf{p}_c = \frac{1}{N} \sum_{i=1}^{N} \mathbf{p}_i$ is the average position and $\mathbf{v}_c = \frac{1}{N} \sum_{i=1}^{N} \mathbf{v}_i$ is the average velocity of the system.

Despite the new formulation of the states, the relative positions and velocities of agents to one another remain identical so that $\tilde{\mathbf{p}}_i - \tilde{\mathbf{p}}_j = \mathbf{p}_i - \mathbf{p}_j$ and $\tilde{\mathbf{v}}_i - \tilde{\mathbf{v}}_j = \mathbf{v}_i - \mathbf{v}_j$. It follows that

$$V(\mathbf{p}) = V(\tilde{\mathbf{p}}), \qquad \nabla V(\mathbf{p}) = \nabla V(\tilde{\mathbf{p}})$$

with $\tilde{\mathbf{p}} = (\tilde{\mathbf{p}}_1^\top, \ldots, \tilde{\mathbf{p}}_n^\top)^\top \in \mathbb{R}^{mn}$. Analogously, the description of the (α, α)-interaction does not change. According to [100, Lemma 2, p. 406], the collective navigation term can be given as

$$\mathbf{f}_\gamma(\mathbf{p}, \mathbf{v}, \mathbf{p}_d, \mathbf{v}_d) = \mathbf{g}(\tilde{\mathbf{p}}, \tilde{\mathbf{v}}) + \mathbf{1}_n \otimes \mathbf{h}(\mathbf{p}_c, \mathbf{v}_c, \mathbf{p}_d, \mathbf{v}_d) \tag{4.15}$$

[2] $\dot{H}$ is only a time derivative (cf. [99, Appendix A.2]).

with[3]

$$\mathbf{g}(\tilde{\mathbf{p}}, \tilde{\mathbf{v}}) = -c_1\tilde{\mathbf{p}} - c_2\tilde{\mathbf{v}}, \tag{4.16}$$

$$\mathbf{h}(\mathbf{p}_c, \mathbf{v}_c, \mathbf{p}_d, \mathbf{v}_d) = -c_1(\mathbf{p}_c - \mathbf{p}_d) - c_2(\mathbf{v}_c - \mathbf{v}_d), \tag{4.17}$$

where $\tilde{\mathbf{v}} = (\tilde{\mathbf{v}}_1^\top, \ldots, \tilde{\mathbf{v}}_n^\top)^\top \in \mathbb{R}^{mn}$. With these, we can reformulate the collective dynamics of the flocking system (Eqs. (4.10) and (4.11)) by decomposing them into

$$\text{structural dynamics} \begin{cases} \dot{\tilde{\mathbf{p}}} = \tilde{\mathbf{v}} \\ \dot{\tilde{\mathbf{v}}} = -\nabla V(\tilde{\mathbf{p}}) - \hat{\mathbf{L}}(\tilde{\mathbf{p}})\tilde{\mathbf{v}} + \mathbf{g}(\tilde{\mathbf{p}}, \tilde{\mathbf{v}}) \end{cases} \tag{4.18}$$

and

$$\text{translational dynamics} \begin{cases} \dot{\mathbf{p}}_c = \mathbf{v}_c \\ \dot{\mathbf{v}}_c = \mathbf{h}(\mathbf{p}_c, \mathbf{v}_c, \mathbf{p}_d, \mathbf{v}_d). \end{cases} \tag{4.19}$$

Apparently, the structural dynamics describes the motion of the α-lattice in the moving frame and the translational dynamics depicts the total displacement of the flock center. With these reformulations, the stability analysis of the flocking system is possible by using LaSalle's invariance principle.

4.2.4 Stability Analysis of the Flocking System

For later analysis in this thesis, it is worth understanding the stability concept of the flocking system applying (4.8) and (4.9). The stability of the flocking system can be defined based on the following properties:

1. The stability of a desired equilibrium state of translational dynamics.
2. The stability of certain equilibrium states of structural dynamics.

Since the translational dynamics depend only on linear terms, the first property is easy to analyze. The velocity match and the convergence of positions to a predefined state depends only on time and the parameters c_1^γ and c_2^γ. However, the second property is more difficult to analyze, because considerably more factors play a role, such as the potential energy, the kinetic energy and the navigation term. Since a target position or a tracking task is common in technical applications of flocking systems, we introduce the stability considering the navigation term $\mathbf{f}_\gamma \neq \mathbf{0}$.

The structural dynamics of the agents can be formulated as follows:

$$\begin{aligned} \dot{\tilde{\mathbf{p}}} &= \tilde{\mathbf{v}}, \\ \dot{\tilde{\mathbf{v}}} &= -\nabla U(\tilde{\mathbf{p}}) - \mathbf{D}(\tilde{\mathbf{p}})\tilde{\mathbf{v}}, \end{aligned} \tag{4.20}$$

[3] $\mathbf{1}_n$ represents an n-dimensional vector whose elements are all equal to one.

where $U(\tilde{\mathbf{p}})$ and $\mathbf{D}$ are the so-called aggregate potential function and the damping matrix, respectively.

$$U(\tilde{\mathbf{p}}) = V(\tilde{\mathbf{p}}) + c_1 J(\tilde{\mathbf{p}}) \tag{4.21}$$

$$\mathbf{D}(\tilde{\mathbf{p}}) = c_2 \mathbf{I}_m + \hat{\mathbf{L}}(\tilde{\mathbf{p}}) \tag{4.22}$$

with the moment of inertia $J(\tilde{\mathbf{p}}) = \frac{1}{2}\sum_{i=1}^{n} \|\tilde{\mathbf{p}}_i\|^2$. The Hamiltonian function representing the structural energy of the system is defined as

$$H(\tilde{\mathbf{p}}, \tilde{\mathbf{v}}) = U(\tilde{\mathbf{p}}) + K(\tilde{\mathbf{v}}), \tag{4.23}$$

where $K(\tilde{\mathbf{p}})$ is the kinetic energy. For a group of agents with the structural dynamics (4.20) and with the assumption that $K(\tilde{\mathbf{p}}(0))$ and $J(\tilde{\mathbf{p}}(0))$ are finite, the following statements hold (cf. [100, Theorem 2]):

1. The group of agents remains cohesive for all times $t \geq 0$.
2. Nearly every solution of the structural dynamics converges to the equilibrium state $(\tilde{\mathbf{p}}^*, \mathbf{0})$, where $\tilde{\mathbf{p}}^*$ is a local minimum of $U(\tilde{\mathbf{p}})$.
3. All agents asymptotically move with the same velocity.
4. Assuming that the initial structural energy of the flocking system is less than $(k+1)c^*$ with $c^* = \psi_\alpha(0)$ and $k \in \mathbb{Z}_+$. Then, at most, k different α-agent pairs can collide. This means that in the case of $k = 0$ a collision-free navigation can be guaranteed.

For proofs of the statements, the reader is referred to [100, p. 408-409]. Since the proof of the third statement is relevant for the proof of Proposition 5.1 in Chapter 5, we briefly mention it in the following.

Proof. The derivative of the Hamiltonian with respect to time is given as

$$\dot{H}(\tilde{\mathbf{p}}, \tilde{\mathbf{v}}) = -\tilde{\mathbf{v}}^\top (c_2 \mathbf{I}_m + \hat{\mathbf{L}}(\tilde{\mathbf{p}}))\tilde{\mathbf{v}} = -c_2(\tilde{\mathbf{v}}^\top \tilde{\mathbf{v}}) - \tilde{\mathbf{v}}^\top \hat{\mathbf{L}} \tilde{\mathbf{v}} < 0. \tag{4.24}$$

This indicates that the system is strictly dissipative for $\tilde{\mathbf{v}} \neq \mathbf{0}$, because the energy $H(\tilde{\mathbf{p}}, \tilde{\mathbf{v}})$ falls monotonically for $H(\tilde{\mathbf{p}}(0), \tilde{\mathbf{v}}(0)) < \infty$.

According to LaSalle's invariance principle, as described in Section 2.3, all solutions of (4.20), which lie in an invariant set $\mathcal{X}$, converge to the largest invariant set $E = \{\tilde{\mathbf{p}}, \tilde{\mathbf{v}} \in \mathcal{X} : \dot{H} = 0\}$. Note that we consider a dynamic system for all $t \geq 0$ and $G(\mathbf{p}(t))$ to be a connected graph. The condition $\dot{H} = 0$ for (4.24) is only fulfilled if all relative velocities represented by means of $\mathbf{v}_c$ match: $\tilde{\mathbf{v}}_1 = \tilde{\mathbf{v}}_2 = \ldots = \tilde{\mathbf{v}}_n$ and $\tilde{\mathbf{v}} = \mathbf{0}$. With this, it follows that $\tilde{\mathbf{v}}_i = 0,\ \forall i$ and the velocities match $(\mathbf{v}_1 = \mathbf{v}_2 = \ldots = \mathbf{v}_n)$.

It can be also concluded that almost all configurations $(\tilde{\mathbf{p}}, \tilde{\mathbf{v}})$ asymptotically converge to an equilibrium state $(\tilde{\mathbf{p}}^*, \mathbf{0})$, which is a local minimum of $U(\tilde{\mathbf{p}})$.

■

4.3 Obstacle Avoidance

For the obstacle avoidance, we assume that every agent can measure the relative position of the closest point on the edge of an obstacle. An obstacle is detected if the Euclidean distance between the agent and obstacle is less than the sensing range as described in Section 2.1.3. A virtual β-agent is created, which is a projection of α-agent i on the edge of the object. In this way, each agent can identify a set of detected obstacle points at a given time instant,

$$\mathcal{N}_i^\beta = \{\hat{\mathbf{p}}_{i,k} \Big| \ \|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\| < r_s\}, \tag{4.25}$$

where r_s is the sensor range for obstacle detection and $\hat{\mathbf{p}}_{i,k}$ is the detected point within the sensors, which has the shortest distance to the agent's current position. In order to ensure obstacle avoidance, the potential for interaction between α- and β-agents is defined as follows:

$$V_\beta(\mathbf{p}) = \sum_{i \in \mathcal{V}_\alpha} \sum_{k \in \mathcal{N}_i^\beta} \psi_\beta(\|\hat{\mathbf{p}}_{\mathbf{i},\mathbf{k}} - \mathbf{p}_i\|)_\sigma. \tag{4.26}$$

The action function to keep a safe distance d_s from an obstacle can then be defined as

$$\varphi_\beta(z) = \rho_{h_\beta}(z/d_\beta)(\sigma_1(z - d_\beta) - 1), \tag{4.27}$$

where $d_\beta < r_\beta$ with $d_\beta = \|d_s\|_\sigma$ and $r_\beta = \|r_s\|_\sigma$. The function (4.27) is purely repulsive and only takes negative values between $0 < z < d_\beta$, and with all other values, it remains at zero (see Fig. 4.1(a)). Using the action function, a repulsive pairwise potential can be given by

$$\psi_\beta(z) = \int_{d_\beta}^{z} \varphi_\beta(s)ds \geq 0. \tag{4.28}$$

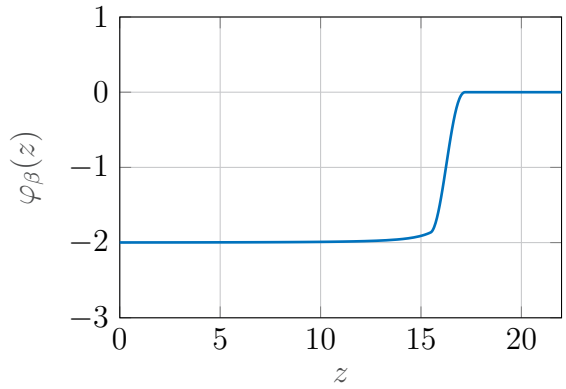
The control input for keeping a safe distance from an obstacle is defined as

$$\mathbf{u}_i^\beta = c_1^\beta \sum_{k \in \mathcal{N}_i^\beta} \varphi_\beta(\|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\|_\sigma)\hat{\mathbf{n}}_{i,k} + c_2^\beta \sum_{k \in \mathcal{N}_i^\beta} b_{i,k}(\mathbf{p})(\hat{\mathbf{v}}_{i,k} - \mathbf{v}_i), \tag{4.29}$$

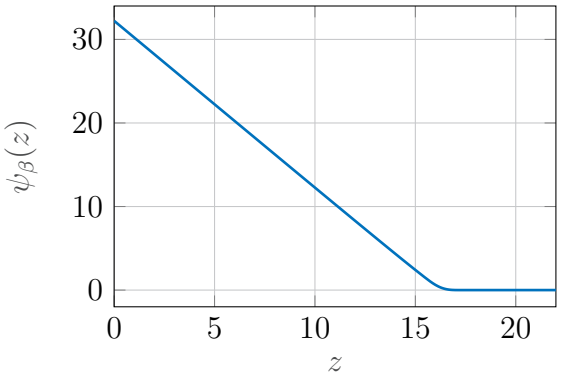
with

$$\hat{\mathbf{n}}_{i,k} = \frac{\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i}{\sqrt{1 + \varepsilon\|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\|^2}}, \ b_{i,k}(\mathbf{p}) = \rho_{h_\beta}\left(\frac{\|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\|_\sigma}{d_\beta}\right),$$

where $\hat{\mathbf{v}}_{i,k}$ is the projection of the $\mathbf{v}_i$ onto the edge of the obstacle k and c_1^β, c_2^β are the positive constant parameters. Moreover, $b_{i,k}$ defines the element of the weighted *adjacency* matrix related to agent i and obstacle k.



(a) Plot of the action function $\varphi_\beta(z)$.



(b) Smooth pairwise potential with finite cut-off in the global minimum at d_β.

Figure 4.1: Parameters for plots: $d_s = 8$, $d_\beta = 17.2$, $\varepsilon = 0.1$, $h_\beta = 0.9$.

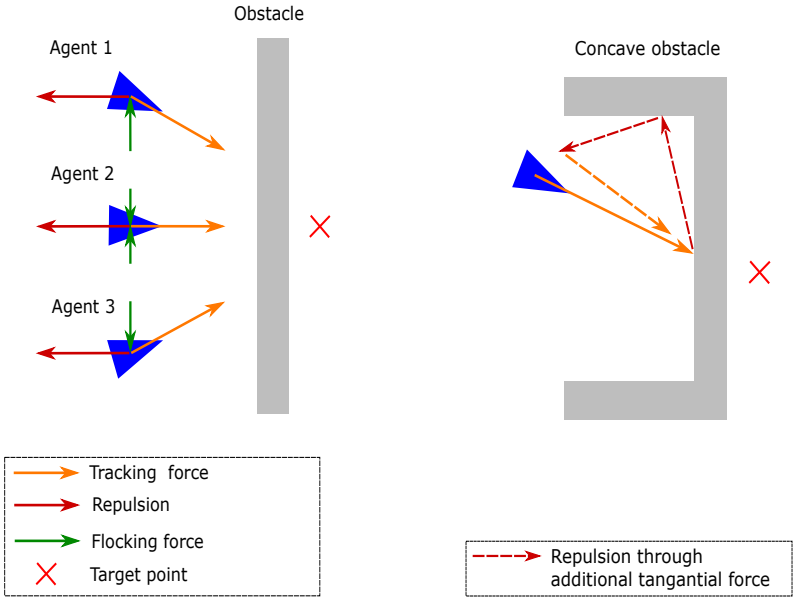
Remark 4.3. The parameters c_η^ν are positive constants for all $\eta = 1, 2$ and $\nu = \alpha, \beta, \gamma$. The priorities of the control objectives *flocking-like behavior*, *obstacle avoidance* and *navigation* are defined by these weighting factors. △

4.4 The Local Minimum Problem

In the following, we describe the problems in artificial potential field-based schemes for local path planning. There are many flocking algorithms using the potential function approach. A potential function depends on the distance among agents or between agents and obstacles. However, one common issue in potential function approaches is that agents can be trapped at a local minimum, where repulsive and attractive forces are balanced (Fig. 4.2(a)). As a result, agents stay stuck behind an obstacle because the gradient vector is zero at a local minimum [98]. Thus, the agents cannot move toward their prescribed objective and the mission might not be successfully accomplished.

Various strategies have been investigated to overcome these problems, especially for single-robot navigation. Sensor-based approaches for collective motion planning include elimination of a local minimum by reconstructing potential functions through additional forces such as the artificial rotational forces proposed in [40]. In this way, the agents can escape from local minima so that the balance between artificial forces are broken. An issue arising in many elimination-based methods is that the success of the escape heavily depends on particular obstacle configurations, especially its concaveness. In addition, escape-based approaches do not always guarantee success in intricate passageways. However, they have a potential to accomplish complex navigation tasks in swarms through communication, evaluation of environmental information and collective intelligence.

An interesting fact in using rotational force fields in combination with the algorithm of Olfati-Saber [100] is that the escape of agents from some concave obstacles heavily



(a) Problem of the local minimum.

(b) Limitation with using a rotational force field as suggested in [40].

Figure 4.2: Conceptual illustration of issues due to the local minimum problem.

depends also on the position of the target point and the concaveness of the obstacle. A typical phenomenon is that an agent first moves through the artificial rotational force away from an obstacle. Then, it loses the perception of the obstacle edge and is driven by the tracking force to its previous position as illustrated in Fig. 4.2(b).

In this chapter, each robot has a range sensor with a limited sensing range r_s and without having prior information about the workspace, it is challenging to navigate the group in an optimal way because the obstacles or restrictions in the operation area cannot be fully perceived with relative short sensing ranges. This requires an elaborated communication and with this, efficient motion planning. Furthermore, each agent can only receive the relative position of its neighbors and exchange information only with other robots within its limited communication bandwidth r_c.

4.5 An Information-Driven Algorithm for an Improved Obstacle Avoidance

Multiple agents are used for cooperative tasks in complex areas. Possible collisions or the loss of some agents during a task may involve costs and lead to failure of the operation. In many studies, control and obstacle avoidance in flocking systems are achieved through potential function-based schemes. However, escape from non-convex obstacles has been barely considered. Due to local minima, concave obstacles are important

issues in the navigation of flocking systems. In study [40], a rotational force field in combination with the repulsive force was used to escape from local minima.

The present method described in this section both derives from and builds upon the flocking algorithm in [100], in which artificial potential functions are applied to avoid collisions and convex obstacles. For this concept, we introduce certain carefully designed rotational forces to the flocking algorithm presented in [100] that share certain similarities with the work [40]. In addition, we enable the agents' ability to recognize possible non-convex obstacles in an anticipatory fashion and replace their target points temporarily through a multi-criteria decision-making process to avoid potential traps. This is based on local information exchange among agents. Furthermore, agents can perform early maneuvers by utilizing local information in case of an upcoming obstacle and local minimum.

4.5.1 Proposed Algorithm

In the proposed concept, the tracking term (4.9) is reconstructed so that possible local minima can be eliminated by setting a virtual temporary target point for each agent based on local information sharing and processing. The concept consists of two heuristic algorithms: The decision for an escape maneuver and the planing of the escape maneuver itself. For this purpose, we utilize a rotational force field, which is constructed by extending the control concept for flocking systems presented in Sections 4.2 and 4.3.

4.5.1.1 Artificial Rotational Forces

Various approaches have been studied to overcome the local minimum problem for flocking multi-agent systems. In the study [40], a rotational force field was proposed as a solution for it. The idea is based on an additional tangential force parallel to the edge of an obstacle, which breaks the balance between attractive and repulsive forces and generates a global rotational motion. Similarly to the method presented in [40], we extend the existing obstacle avoidance term by a new tangential force as follows:

$$\mathbf{u}_i^{\beta} = c_1^{\beta} \sum_{k \in \mathcal{N}_i^{\beta}} \varphi_{\beta}(\|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\|_{\sigma}) \hat{\mathbf{n}}_{i,k} + c_2^{\beta} \sum_{k \in \mathcal{N}_i^{\beta}} b_{i,k}(\mathbf{p})(\hat{\mathbf{v}}_{i,k} - \mathbf{v}_i) + \sum_{k \in \mathcal{N}_i^{\beta}} \mathbf{f}_{i,k}^{\beta r} \tag{4.30}$$

with

$$\mathbf{f}_{i,k}^{\beta r} = w_i^{\beta} \mathbf{n}_{i,k}^{\beta} \tag{4.31}$$

and

$$\mathbf{n}_{i,k}^{\beta} = \frac{c_i^{\beta r}}{\|\mathbf{p}_i - \hat{\mathbf{p}}_{i,k}\|} \begin{pmatrix} y_i - \hat{y}_{i,k} \\ -(x_i - \hat{x}_{i,k}) \end{pmatrix}, \tag{4.32}$$

where $\mathbf{p}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix}$ and $\hat{\mathbf{p}}_{i,k} = \begin{pmatrix} \hat{x}_{i,k} \\ \hat{y}_{i,k} \end{pmatrix}$ are the position of α-agent i and the generated virtual β-agent, respectively. Unit normal vector $\mathbf{n}_{i,k}^{\beta}$ is thus always perpendicular to the vector connecting an agent to the obstacle and it holds: $\langle(\mathbf{p}_i - \hat{\mathbf{p}}_{i,k}), \mathbf{n}_{i,k}^{\beta}\rangle = 0$. The direction of the rotational force $c_i^{\beta r}$ can be defined as follows:

$$c_i^{\beta r} = \begin{cases} +1, & \text{Clockwise rotation} \\ -1, & \text{Counter-clockwise rotation} \end{cases} \tag{4.33}$$

In Eq. (4.31), w_i^{β} is a critical parameter, which represents a positive gain to adjust the escape strength of agent i from concave obstacles and it is given by

$$w_i^{\beta} = (1 + c)(\|\mathbf{f}_i^{\alpha\beta}\| + \lambda_0), \tag{4.34}$$

where λ_0 is a positive factor and independent of the other terms. $\mathbf{f}_i^{\alpha\beta}$ is defined as the sum of the repulsive force generated virtually through the obstacle and the first component of the tracking term (4.9). This is given as

$$\mathbf{f}_i^{\alpha\beta} = c_1^{\beta} \sum_{k \in \mathcal{N}_i^{\beta}} \varphi_{\beta}(\|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\|_{\sigma}) \hat{\mathbf{n}}_{i,k} + c_1^{\gamma} \sigma_1(\mathbf{p}_i - \mathbf{p}_r). \tag{4.35}$$

The constant parameter c in (4.34) strengthens or weakens the rotation. This is related to the angle between $\mathbf{f}_i^{\alpha\beta}$ and $\mathbf{n}_{i,k}^{\beta}$. If this is smaller than $\frac{\pi}{2}$ then $c = c_1$, otherwise $c = c_2$. These constants can be chosen as $-1 < c_1$, $0 < c_2$ and $c_1 < c_2$. This can be explained as follows: If the angle between $\mathbf{f}_i^{\alpha\beta}$ and $\mathbf{n}_{i,k}^{\beta}$ is less than 90°, the agent already receives a small amount of force along the obstacle. In this case, a strong rotational force is not necessary. However, if the angle is greater than 90°, a component of $\mathbf{f}_i^{\alpha\beta}$ counteracts $\mathbf{f}_{i,k}^{\beta r}$ and thus, a greater gain w_i^{β} is required to tangentially escape from the obstacle. This is illustrated in Fig. 4.3. Note that the unit vector of $\mathbf{f}_{i,k}^{\beta r}$ corresponds to $\mathbf{n}_{i,k}^{\beta}$.

Remark 4.4. The algorithms in this section are designed for the *clockwise* rotation direction, $c_i^{\beta r} = +1$. △

4.5.1.2 Information Map and Information Exchange

The proposed algorithm requires communication of acquired information. For this purpose, the given workspace is virtually modeled by the agents as a grid with a finite number of cells. During the operation, each agent generates its own information map that includes the sensed obstacle points and thus defines the occupied cells in the virtual cellular map of the workspace. Similarly to [147, 13, 125], the workspace is decomposed into equal-sized cells. Hereby, $\mathbf{M}_i$ is the information map of agent i . In addition, each agent is capable of memorizing the unit normal vector $\mathbf{n}_{i,k}^{\beta}$ required for the rotational force and creates its own history of tangential vectors $\mathbf{N}_i$. Agents can

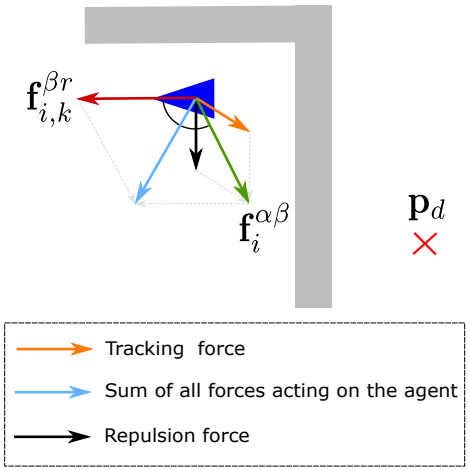


Figure 4.3: Illustration of the forces used for the rotational force field.

share this information with their neighboring agents via local communication. For each agent i, whose sensing range is r_s, its information map is updated in the following way:

- In the beginning of a mission ($t = 0$), the information map $\mathbf{M}_i$ and the history of tangential vectors $\mathbf{N}_i$ are empty. This means that the agent i assumes that the workspace is obstacle-free.
- Once an obstacle is detected by the agent i, the coordinates of the nearest point on the edge of the obstacle are stored in the information map $\mathbf{M}_i$. In addition, the unit normal vector $\mathbf{n}_{i,k}^{\beta}$ used to generate the additional force (4.31), which is tangential to the edge of the detected obstacle, is saved in the $\mathbf{N}_i$.
- If an agent j is in the communication range of the agent i,

  $$\|\mathbf{p}_j - \mathbf{p}_i\| < r_c,$$

  they can exchange their information maps and the history of tangential vectors,

  $$\mathbf{M}_i = \mathbf{M}_j, \quad \mathbf{N}_i = \mathbf{N}_j.$$

4.5.1.3 Decision for the Strategy

Using Algorithm 4.1, agents can recognize potential concave obstacles and make decisions for their escape strategy based on the gathered information. Each agent uses its sensor activity. In order to identify the current detection status, each agent uses a flag s_i defined by

$$s_i(t) = \begin{cases} 1, & \text{if an obstacle within the sensing range } r_s \\ 0, & \text{otherwise} \end{cases} \tag{4.36}$$

Firstly, we check if any rotation direction and any information about an obstacle are available. The tangential forces for the rotation are indicated with orange-colored arrows in Fig. 4.4. If the agent i has performed more than two different rotations and if it currently senses an obstacle, control *Strategy 1* is activated (line 7, cf. Fig. 4.4(c)). More than two different rotation directions denote that the agent might attempt to escape from a concave obstacle. In addition, we check if the tracking direction and tangential force have the identical x-direction (line 11). A movement in the opposite direction or perpendicular to the tracking direction usually indicates that the agent is trapped in a concave obstacle (Fig. 4.4(b)). In this case, *Strategy 2* should be applied.

Each agent defines its individual *safe area* based on its information map $\mathbf{M}_i$. An agent, which uses a clockwise rotational force field, is in its *safe area* if its current y-coordinate y_i is greater than the y-coordinates of all detected points.

This means a point in $\mathbf{M}_i$ with the greatest y-coordinate. If the agent arrives at its safe area (the green-colored area in Fig. 4.4), the control is switched to its initial strategy (line 14 - *Strategy 3*).

The last strategy is designed for recognizing the end position of an obstacle. If an agent realizes that it is at the end of an obstacle, it switches to another control strategy (line 4). If none of the defined conditions is met, the initial strategy (*Strategy 3*) is activated.

4.5.1.4 Escape Strategies from Non-Convex Obstacles

Agents initially have a global target position ($\mathbf{p}_d$, $\mathbf{v}_d = 0$). By using Algorithm 4.2, each agent calculates an individual virtual goal ($\mathbf{p}_i^v$, $\mathbf{v}_i^v = 0$), which is based on sensing information and on the previously determined escape strategy. In obstacle-free areas and for *Strategy 3*, the tracking input is applied in a regular way according to Eq. (4.9). Otherwise, the control input for tracking (4.9) is modified by replacing $\mathbf{p}_d$ temporarily by $\mathbf{p}_i^v$ as follows:

$$\mathbf{u}_i^{\gamma,temp} = -c_{1v}^{\gamma}\sigma_1(\mathbf{p}_i - \mathbf{p}_i^v) - c_{2v}^{\gamma}(\mathbf{v}_i - \mathbf{v}_i^v). \tag{4.37}$$

When Strategy 1 is active, the target point $\mathbf{p}_d$ is replaced temporarily by a new virtual target position $\mathbf{p}_i^v$. Considering the last detected point on the obstacle $\hat{\mathbf{p}}_i^{last}$ (light blue-colored cross in Fig. 4.4(c)) and the unit vector $\mathbf{n}_i^{last}$ lastly applied to generate the rotation, the new virtual target point is given as

$$\mathbf{p}_i^v = \hat{\mathbf{p}}_i^{last} + \bar{a} \cdot \mathbf{n}_i^{last} + \bar{\mathbf{b}}. \tag{4.38}$$

For *Strategy 2*, first, the occupied and unoccupied cells in the workspace are defined according to the information map $\mathbf{M}_i$. Occupied cells are colored with light blue in Fig. 4.4. Then, the point in the information map with the greatest y-coordinate is determined (dark green-colored cross in Fig. 4.4(b)). If there are many points with the identical y-coordinate, the one with the minimum x value is chosen (i.e., (x_{min}, y_{max})). Following this, the agent calculates the center points of the nearest unoccupied cells

Algorithm 4.1 : Decision for the escape strategy

Data : $\mathbf{p}_i$, $\mathbf{p}_d$, $\hat{\mathbf{p}}_i^{last}$, $\mathbf{N}_i$, $\mathbf{M}_i$, s_i

Result : target_control

$\mathbf{N}_i = \{\mathbf{n}_i^1, \mathbf{n}_i^2 \dots \mathbf{n}_i^n\}$, $\mathbf{M}_i = \{\mathbf{m}_i^1, \mathbf{m}_i^2 \dots \mathbf{m}_i^q\}$

$\mathbf{p}_i = [x_i, y_i]$, $\hat{\mathbf{p}}_i^{last} = \left[\hat{x}_i^{last}, \hat{y}_i^{last}\right]$

if $\mathbf{N}_i$ *is not empty* ***and*** $\mathbf{M}_i$ *is not empty* **then**

  if $\hat{x}_i^{last} \leq x$*-obstacle end position* ***and*** $s_i = 1$ **then**

    Apply *Strategy 4*;

    return

  if $size(\mathbf{N}_i) > 2$ ***and*** $s_i = 1$ **then**

    Apply *Strategy 1*;

    return

  for $j = 1 : size(\mathbf{N}_i)$ **do**

    ▷ $n_{i,x}$: x-component of $\mathbf{n}_i^z$, V_x: x-component of $(\mathbf{p}_i - \mathbf{p}_d)$

    if $\frac{n_{i,x}}{\|n_{i,x}\|} \leq \frac{V_x}{\|V_x\|}$ ***and*** $\mathbf{M}_i$ *is not empty* **then**

      Apply *Strategy 2*;

      return

    else if $y_i > max(\mathcal{Y}_{map})$ ▷ $\mathcal{Y}_{map}$ represents the set of y-coordinates of all detected points.

    then

      Apply *Strategy 3*;

    else

      Apply *Strategy 3*;

else

  Apply *Strategy 3*;

in the workspace corresponding to (x_{min}, y_{max}). The new temporary target point is defined based on the center point of one of these areas with the greatest y-coordinate and the minimum x-coordinate ($= \hat{\mathbf{p}}_i^{target}$) in the following way:

$$\mathbf{p}_i^v = \hat{\mathbf{p}}_i^{target} + \bar{\mathbf{c}}. \tag{4.39}$$

$\hat{\mathbf{p}}_i^{target}$ is illustrated with the purple-colored star symbol in Fig. 4.4(b).

Strategy 3 is the initial control strategy, where the target point remains unchanged. Moreover, with *Strategy 4* (Fig. 4.4(d)), the temporary target point is placed using $\hat{\mathbf{p}}_i^{last}$ after detection of the endpoint of an obstacle and the $\mathbf{p}_i^v$ is given by

$$\mathbf{p}_i^v = \hat{\mathbf{p}}_i^{last} + \bar{\mathbf{d}}. \tag{4.40}$$

The parameter $\bar{a}$ is a constant and $\bar{\mathbf{b}}$, $\bar{\mathbf{c}}$ and $\bar{\mathbf{d}}$ represent constant vectors.

Algorithm 4.2 : Execution of the escape maneuvers

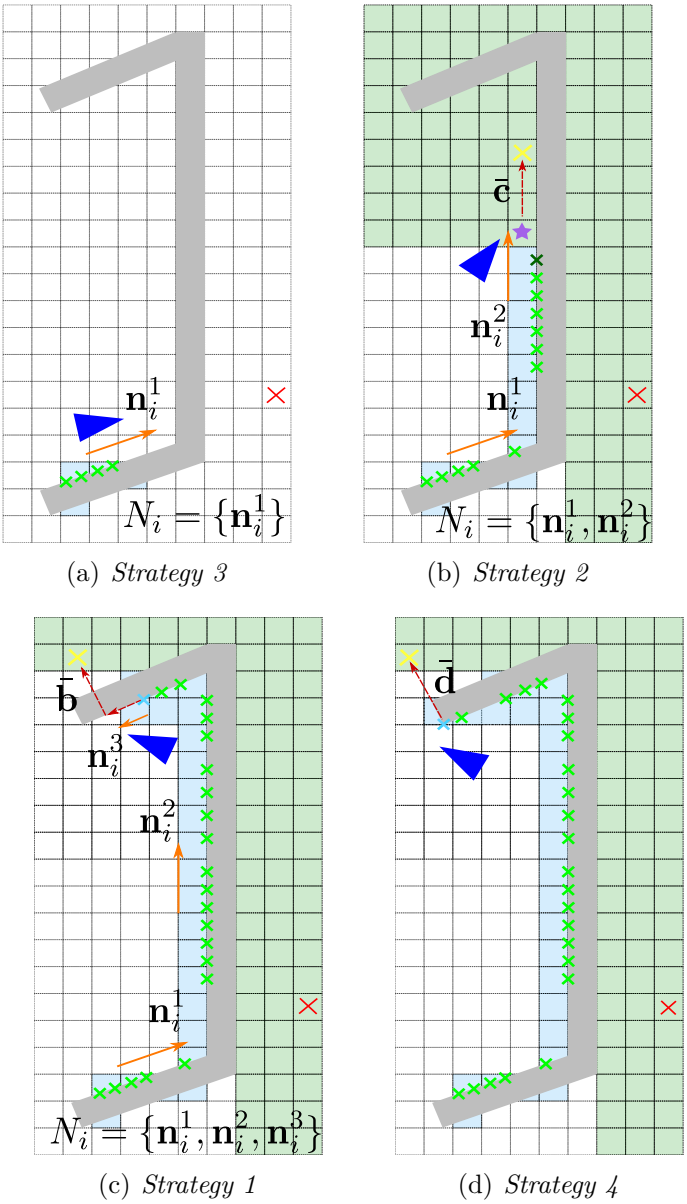
Data : $\mathbf{p}_i$, $\mathbf{v}_i$, $\mathbf{p}_d$, $\mathbf{n}_i^{last}$, $\hat{\mathbf{p}}_i^{last}$, $\mathbf{N}_i$, $\mathbf{M}_i$, s_i

Result : $\mathbf{u}_i^{\gamma,temp}$, $\mathbf{u}_\gamma$

```
if Strategy 1 is active then
    Define a new temporary target point using Eq. (4.38);
else if Strategy 2 is active then
    Define occupied and unoccupied cells using the information map M_i;
    Determine the position of a detected point with (x_min, y_max) in the information
      map;
    Calculate the center points of the nearest unoccupied cells;
    Set a new temporary target point using Eq. (4.39);
else if Strategy 3 is active then
    The target point remains unchanged;
else if Strategy 4 is active then
    Set a new temporary target point based on p̂_i^last using (4.40);
else
    The target point remains unchanged;
```

4.6 Simulation Studies

In this section, we present results of the simulations to demonstrate the effectiveness of the proposed algorithm. For the simulations, we performed numerical experiments with a concave semi-circular obstacle and a concave inclined frame. Both have concave characteristics with respect to the agents' direction of motion.



(a) *Strategy 3* (b) *Strategy 2*

(c) *Strategy 1* (d) *Strategy 4*

Figure 4.4: Illustration of the proposed escape strategies. The yellow-colored cross denotes the temporary target point and the red-colored one represents the global target point.

We consider a system of $N = 5$ agents in the two-dimensional plane $m = 2$. The agents are randomly placed in an area of $[-5, 5] \times [11, 21]$ and their initial velocities $\mathbf{v}_i(0) \in \mathbb{R}^2$ are $[0, 0]$. In all tasks, the target position is defined at $\mathbf{p}_d = (80, 25)^\top$. The workspace is defined in an area of $[-6, 100] \times [0, 80]$ with a cell size of $[2 \times 2]$. The values of the flocking parameters, the weighting factors of the applied control terms and the parameters for the rotational force are given in Table 4.1. Since safety is generally an important criterion in navigation, the *obstacle avoidance* ($\mathbf{u}_\beta$) has the highest priority. The second important task is the *target tracking* ($\mathbf{u}_\gamma$) and then the

flocking ($\mathbf{u}_\alpha$). According to these task priorities, the constant weighting factors are chosen as: $c_1^\beta > c_1^\gamma > c_1^\alpha$.

Fig. 4.5 shows the positions of the agents for the important instants $t = [1\,s, 20\,s, 28\,s, 50\,s]$ and their trajectories up to these time instants. The blue triangles represent the positions and the directions of motion of the agents, and the red-colored cross corresponds to the global target point $\mathbf{p}_d$. It can be seen that the group can avoid the concave semi-circular obstacle. In addition, the exchange of information maps as well as the rotation information enables some agents to make an early decision for an escape maneuver without detecting the obstacle.

In the second simulation, the initial configuration of agents is the same as in the first simulation. Fig. 4.6 shows the positions of the agents for the time instants $t = [2\,s, 20\,s, 41\,s, 80\,s]$ and their trajectories. Here, it can be observed that all agents follow a similar path to the target point ($t = 41\,s$). The reason for the joint path is that the agents set their virtual temporary target point to a point above the sensed position with the greatest y-coordinate (Algorithm 4.1, line 11). However, in the case of an inclined frame, agents have to move along the obstacle's edge to find an exit to the *safe area*. Thus, they have to sense the obstacle (Algorithm 4.1, line 7). After leaving the obstacle behind, the group quickly focuses on the global target point.

The fact that a specific direction of rotation is given results in agents having to perform only a clockwise rotation to escape from obstacles. This may not always be the most efficient way because it might prolong the path to the target point and limit the application in many areas. However, it is a reliable way to move away from simple concave obstacles preserving the cohesive behavior.

Table 4.1: Parameter setting.

Flocking parameters	d	d_s	r_c	r_s	ε
	7	$0.6 \cdot d$	$1.2 \cdot d$	$1.2 \cdot d_s$	0.1
	a	b	h_α	h_β	m
	5	5	0.2	0.9	2
Weighting of terms	c_1^α	c_1^β	c_1^γ	c_{1v}^γ	
	20	80	30	70	
	c_2^α	c_2^β	c_2^γ	c_{2v}^γ	
	$2\sqrt{c_1^\alpha}$	$2\sqrt{c_1^\beta}$	$2\sqrt{c_1^\gamma}$	$2\sqrt{c_{1v}^\gamma}$	
Rotation parameters	c_1	c_2	$c_i^{\beta r}$	λ_0	
	0	1	1	1	
Escape parameters	$\bar{a}$	$\bar{\mathbf{b}}$	$\bar{\mathbf{c}}$	$\bar{\mathbf{d}}$	
	10	$(2, -2)^T$	$(0, 5)^T$	$(-5, 5)^T$	

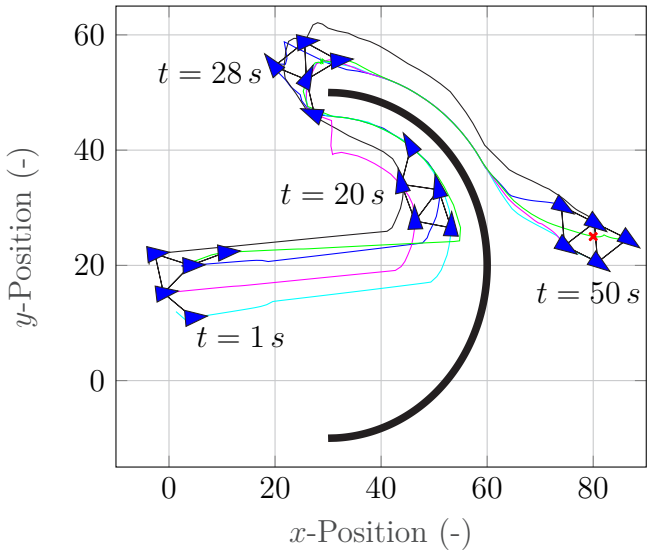


Figure 4.5: Snapshots of a group of $N = 5$ agents during the escape from a concave semi-circular obstacle.

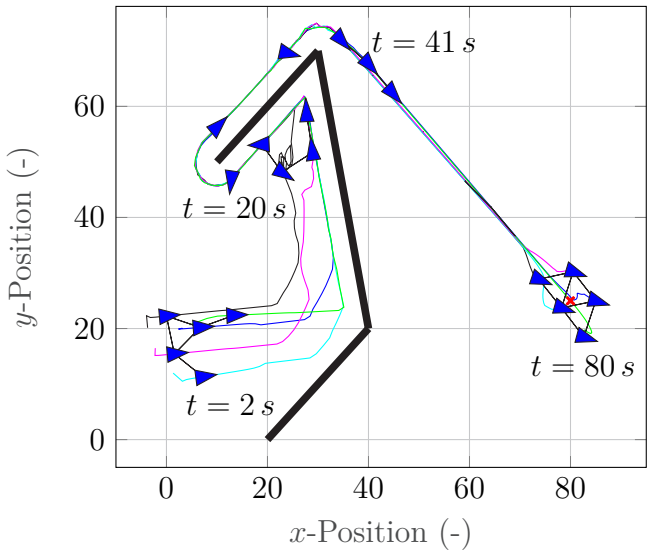


Figure 4.6: Snapshots of a group of $N = 5$ agents during the escape from a concave inclined frame.

4.7 Chapter Highlights

In this chapter, we proposed an information-driven algorithm for flocking systems to escape from simple concave obstacles and to prevent local minima. With this algorithm, each agent can explore the workspace and generate an information map through local information exchange. In addition, the existing flocking algorithm in [100] is extended by judicious design of an extra term to generate a rotational force field. Information exchange and recognition of concave obstacles make it possible for the agents to perform

early escape maneuvers. In this way, the success rate of a mission with multiple agents can be increased. However, for the escape from more complex obstacles, more sophisticated information processing and decision-making algorithms are needed. Moreover, it is possible to extend the concept for variable rotation direction in order to generate more optimal paths for the agents.

The supervised student thesis [160] has contributed to the development of this chapter's results.

5 Collective Navigation Framework for a Multi-Agent System

The content of the following chapter has been published in [158].

Beside potential field-based methods and VFH, which are plagued by being trapped into a local minimum, there are escape-based methods to overcome local minima. Performing a spiral motion around a specific point [88, 49], generating elliptic trajectories [140] and adding a reactive rotational force field [40] are some of these strategies. Cooperative, decentralized navigation of multi-robots through information exchange among group members and data evaluation is another approach for planning optimal collective motions [45]. An important challenge in escape-based approaches are concave and labyrinth-like obstacles, which can appear in indoor applications.

The main contributions of this chapter are summarized as follows:

(i) First, we formulate a single-robot navigation strategy similar to that in study [28], which employs only sensors. This method allows a robot to navigate in an unknown static environment by using only range sensors. We extend it to a decentralized multi-robot navigation framework, where each robot only needs to communicate with its neighbors for motion planning in an independent way.

(ii) Second, by means of local communication and information exchange, we enable the robots to make early decisions and perform simultaneous, optimal collective maneuvers during a task in a distributed manner. Furthermore, using neighbor-to-neighbor communication increases flexibility and scalability with respect to the system size (agent number). In addition, we use potential fields to guarantee inter-agent collision avoidance, preserve proximity and increase safety in collision avoidance with static obstacles. Moreover, in this study, we consider the limited sensor and communication range of the robots to be a challenging factor.

The chapter is organized as follows: Motivated by the local minimum problem in navigation, Section 5.1 presents a navigation scheme for a single robot, which is mainly based on [27]. In Section 5.2, we extend the scheme from the previous section and propose a novel approach for the navigation of a multi-robot system. Subsequently, Section 5.3 presents simulation results to show the effectiveness of the proposed approach in highly complex environments.

5.1 Single Robot Navigation

In this section, we briefly introduce a navigation schema for a single agent ($N = 1$), which we extend with a communication interface, collective motion planning framework and potential forces in later sections. The proposed approach is mainly inspired by the tangential navigation schema presented in [28]. In obstacle-free areas, agents use the control term (4.9) to move toward the desired goal position. However, the proposed approach in [28] is based on the reconstruction of the *navigation* term (4.9) by allocating temporary virtual goal positions considering the obstacle geometry to avoid local minima.

The dynamics of each mobile agent is described by double-integrators as in previous chapters.

$$\dot{\mathbf{p}}_i = \mathbf{v}_i,$$
$$\dot{\mathbf{v}}_i = \mathbf{u}_i,$$

where $(\mathbf{p}_i, \mathbf{v}_i) \in \mathbb{R}^2 \times \mathbb{R}^2$ denote the position and velocity of agents, respectively.

5.1.1 Tangential Navigation

Once an obstacle is in the range of r_{tan} ($r_s > r_{tan} > d_s$), $\|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\| \leq r_{tan}$, the robot starts to perform the tangential navigation and it travels parallel to the obstacle. Firstly, the robot determines the angle γ for the rotation matrix to project the desired goal position as follows

$$\gamma_i = \begin{cases} \beta_i - \alpha_i - 90^\circ, & \text{if } \beta_i \geq 0^\circ \\ \beta_i - \alpha_i + 90^\circ, & \text{if } \beta_i < 0^\circ \end{cases} \tag{5.1}$$

where the angles α_i and β_i are the orientation of the robot relative to the desired goal position and to the closest obstacle point, respectively, as shown in Fig. 5.1. The angle β_i is in the range $]-180, 180]$ and defines the relative position of an obstacle to the robot. If $\beta_i < 0^\circ$, the obstacle is to the right of the robot. If $\beta_i > 0^\circ$, the obstacle is on the left.

According to Eq. (5.1), the rotation direction of the agent depends on its orientation to the obstacle. For a better cohesive behavior in complex environments, we can predefine a primary direction of rotation using the following parameter:

$$c_r = \begin{cases} -1, & \text{for clockwise rotation} \\ 1, & \text{for counter-clockwise rotation} \end{cases} \tag{5.2}$$

In this case, the rotation angle should be calculated as follows:

$$\gamma_i = \beta_i - \alpha_i + c_r \cdot 90^\circ. \tag{5.3}$$

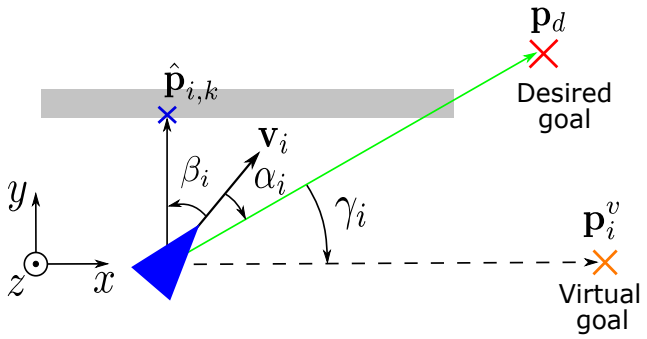


Figure 5.1: Illustration of the tangential navigation schema.

The angle γ_i is used as a rotation angle for the calculation of the virtual goal position $\mathbf{p}_i^v$ defined as

$$\mathbf{p}_i^v = \mathbf{p}_i + \begin{bmatrix} cos(\gamma_i) & -sin(\gamma_i) \\ sin(\gamma_i) & cos(\gamma_i) \end{bmatrix} (\mathbf{p}_d - \mathbf{p}_i). \tag{5.4}$$

The control input for the navigation is calculated similar to Eq. (4.9) as follows

$$\mathbf{u}_i^\gamma = -c_1^\gamma \left(c_n \cdot \frac{\mathbf{p}_i - \mathbf{p}_i^v}{\|\mathbf{p}_i - \mathbf{p}_i^v\|} \right) - c_2^\gamma \mathbf{v}_i, \tag{5.5}$$

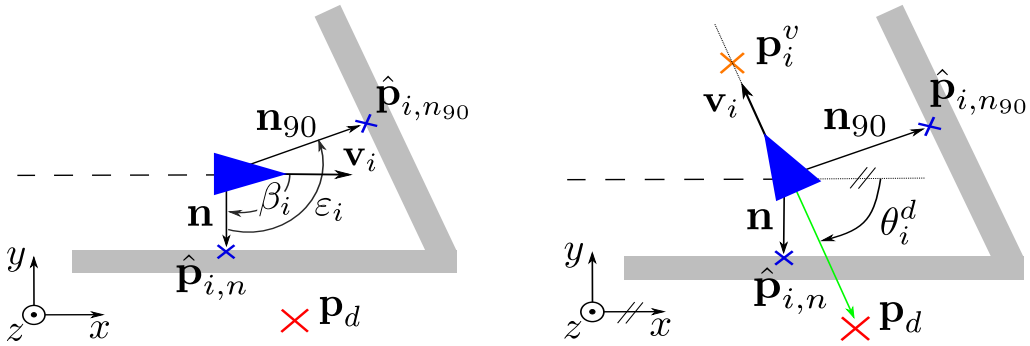
where c_n is a positive constant for specifying a constant acceleration toward the virtual goal position.

5.1.2 Corner Avoidance

The characteristic of a concave corner is that the robot simultaneously senses two different points on an obstacle. Each sampling time (each $0.02\,s$, for our simulation setup), the motion planner checks if two different points are sensed. If two obstacles are detected at the same time, the robot applies the corner avoidance maneuver.

Firstly, the robot classifies the detected obstacle points for the motion planning. Here, the normal vector to the obstacle, which lies in the direction of motion $\mathbf{v}_i$ in front of the agent, is defined as $\mathbf{n}_{90}$, the other one as $\mathbf{n}$. The corresponding sensed obstacle points are declared as $\hat{\mathbf{p}}_{i,n_{90}}$ and $\hat{\mathbf{p}}_{i,n}$. In order to calculate the rotation angle γ_i, we introduce the angle ε_i, which is defined between $\mathbf{n}$ to $\mathbf{n}_{90}$, measured from $\mathbf{n}$ (Fig. 5.2(a)). The rotation angle for projection of a new temporary, virtual goal position is defined as follows:

$$\gamma_i = \begin{cases} \beta_i - \alpha_i - \tau_i, & \text{if } \varepsilon_i \geq 0° \\ \beta_i - \alpha_i + \tau_i, & \text{if } \varepsilon_i < 0° \end{cases} \tag{5.6}$$



(a) Corner detection. (b) Corner avoidance maneuver.

Figure 5.2: Corner avoidance schema.

with

$$\tau_i = |\varepsilon_i| + 90°. \tag{5.7}$$

The angle ε_i defines the necessary direction of rotation in an indirect way, e.g., $\varepsilon_i \geq 0°$ results in a counter-clockwise rotation. Finally, the virtual goal position is calculated as

$$\mathbf{p}_i^v = \mathbf{p}_i + \begin{bmatrix} cos(\gamma_i) & -sin(\gamma_i) \\ sin(\gamma_i) & cos(\gamma_i) \end{bmatrix} \begin{bmatrix} 0.5 \cdot d_s \cdot cos(\theta_i^d) \\ 0.5 \cdot d_s \cdot sin(\theta_i^d) \end{bmatrix}, \tag{5.8}$$

where θ_i^d describes the desired orientation of the robot to the current, temporary goal position (Fig. 5.2(b)). The angle θ_i^d is defined as the angle between the x-axis of the inertial coordinate system and the line linking the agent's position and the desired goal position.[1] Note that the robot sets the virtual goal point to a closer area with the aim of generating a *weaker* attraction force compared to the tangential navigation. In this way, it performs more precise and safer maneuvers in the corners.

5.1.3 Motion Planning at Obstacle Extremities

While a robot follows an obstacle tangentially, it loses the sensing contact shortly after reaching the obstacle endpoint. Without proper motion planning, it would move back to the desired goal position and might get into a local minimum instead of performing a suitable maneuver to turn around the obstacle extremity.

For the endpoint detection, we introduce a binary flag P_{tan} as in [28]. During the tangential navigation, the robot continuously senses the obstacle, $P_{tan} = 1$. The flag is only reset to 0 once there is no longer an obstacle inside the radius r_{tan}. If the flag P_{tan} is reset from 1 to 0, the robot recognizes the case *endpoint* and utilizes the

[1] The angle θ_i^d is measured from the positive x-axis.

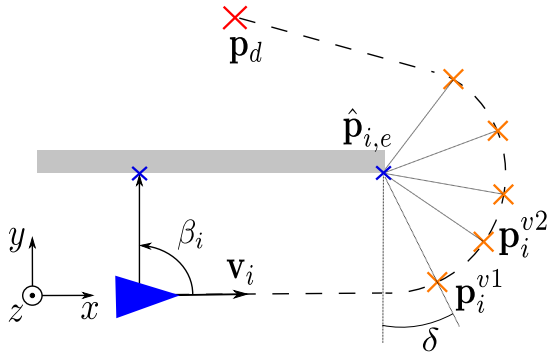


Figure 5.3: Motion planning at the obstacle endpoint.

detected obstacle point from the last time instant ($\hat{\mathbf{p}}_{i,k}(t_{k-1}) = \hat{\mathbf{p}}_{i,e}$), illustrated by the blue-colored cross in Fig. 5.3, as a center point for its rotational motion. Furthermore, the calculated angle ($\beta_i(t_{k-1}) = \beta_{i,e}$) is also stored. Along a circular path, the robot iteratively calculates virtual goal positions rotating around $\hat{\mathbf{p}}_{i,e}$ by a defined angle of rotation δ. The angle of rotation for the maneuver is formulated as:

$$\gamma_i = \begin{cases} +\delta, & \text{if } \beta_{i,e} \geq 0° \\ -\delta, & \text{if } \beta_{i,e} < 0° \end{cases} \tag{5.9}$$

where $\delta > 0°$ represents the value of the predefined rotation angle. In the next step, the virtual goal position is determined on a circular path by the following equation:

$$\mathbf{p}_i^v = \hat{\mathbf{p}}_{i,e} + \begin{bmatrix} cos(\gamma_i) & -sin(\gamma_i) \\ sin(\gamma_i) & cos(\gamma_i) \end{bmatrix} \mathbf{n}_{i,e}, \tag{5.10}$$

with

$$\mathbf{n}_{i,c} = \frac{\mathbf{p}_i(t_k) - \hat{\mathbf{p}}_{i,e}}{\|\mathbf{p}_i(t_k) - \hat{\mathbf{p}}_{i,e}\|} \cdot r_{tan},$$

where the radius of the circular path is equal to r_{tan}. Once the agent reaches a virtual goal position on the circular path, e.g., $\|\mathbf{p}_i^{v1} - \mathbf{p}_i\| < 1$, it determines a new one, e.g., $\mathbf{p}_i^{v2}$ (see Fig. 5.3). In this way, the agent keeps setting new goal positions until the following condition is fulfilled:

$$|\alpha_i| \leq \delta. \tag{5.11}$$

This allows the robot an optimal orientation with respect to the desired goal position $\mathbf{p}_d$ to leave the circular path and to move toward the desired goal position.

5.2 The Proposed Navigation Approach for a Multi-Agent System

The proposed approach is based on the navigation algorithm for a single robot described in Section 5.1 and extends it by means of a novel communication interface for collective motion planning. With the help of inter-agent communication, agents are capable of perceiving and acting upon the environment by using the information from the communication network. In this section, we introduce the communication interface.

The basic principle of the navigation scheme presented in Section 5.1 is the tangential projection of a virtual goal position. In the case of multiple agents, the continuous creation of an individual, virtual goal position may yield a decomposition of the group or collisions. In order to prevent this, the information about the projection of a new, virtual goal should be communicated within the neighborhood of each agent. In addition, the critical points identified on an obstacle, such as corners and endpoints, should also be shared for a collective navigation strategy.

Fig. 5.4 shows the critical points for the communication scenarios. The first scenario depicts the tangential navigation along an obstacle. The second scenario is the detection of a corner and the third one represents the detection of an obstacle extremity. These scenarios are communicated independently from one other. In this way, the agents also acquire knowledge about the area and utilize the information for localization. The desired principles for the communication process in our approach are as follows:

- An agent should consider only the most relevant information for its next action.
- Only the contemporary information in the communication network should be considered.
- Detection before communication: A detected obstacle point has a higher priority than information received through communication in motion planning. Navigation based on the information aggregated from the communication network may result in a collision if an obstacle is detected. Hence, in such cases, the detected obstacle point is primarily taken into consideration for the next action of the agent. In this way, the agent can certainly avoid a possible collision with the obstacle.

5.2.1 Structure of the Communication Interface

The communication network contains information packages aggregated by means of own sensors and the information received from other agents via communication. Note that the information items received from the communication network will be identified later by the upper index c. In our concept, we define three types of information packages: *orientation information*, *information about endpoints* and *information about corners*. Each type of information package consists of several different items.

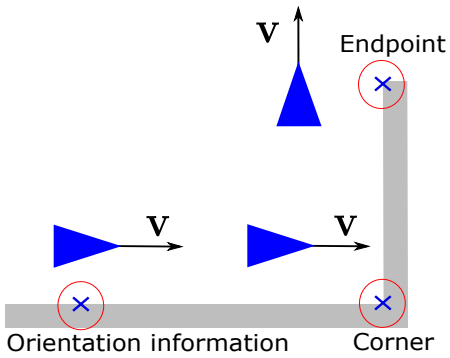


Figure 5.4: The critical points for the communication scenarios.

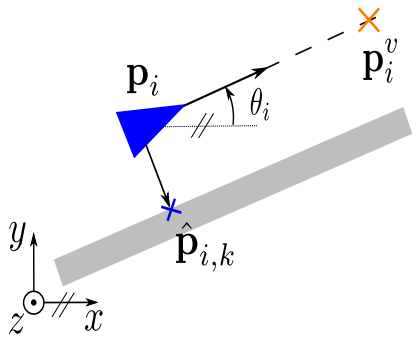


Figure 5.5: Orientation information.

Orientation Information

The information package consists of several items. The core item is the goal orientation θ_i and the detected point on the obstacle $\hat{\mathbf{p}}_{i,k}$. The angle goal orientation describes the angle between the x-axis of the inertial coordinate system and the line linking the agent's position with the current virtual goal position.[2] A further important item is $status_i$, which defines the current action of the agent and is included in the information package. We implemented several statuses that are explained later (see Section 5.2.3), as shown in Table 5.1.[3] Through the assigned statuses, an agent receiving an information package can identify the actions of its neighboring agents. The last item is $\hat{t}_i$, which represents the detection time of an obstacle.

Information about Endpoints

At the endpoint of an obstacle, the agent starts to move on a circular path in the direction of the desired goal position. This motion is planned by means of the last detected point $\hat{\mathbf{p}}_{i,e}$ and the angle $\omega_{i,e}$ between the normal vector from the agent to the obstacle and the x-axis of the inertial coordinate system.[4] Moreover, the goal

[2]The angle θ_i is measured from the positive x-axis.
[3]The transitions between the action statuses are given as pseudo code in Appendix A.
[4]$\omega_{i,e}$ is defined from the positive x-axis.

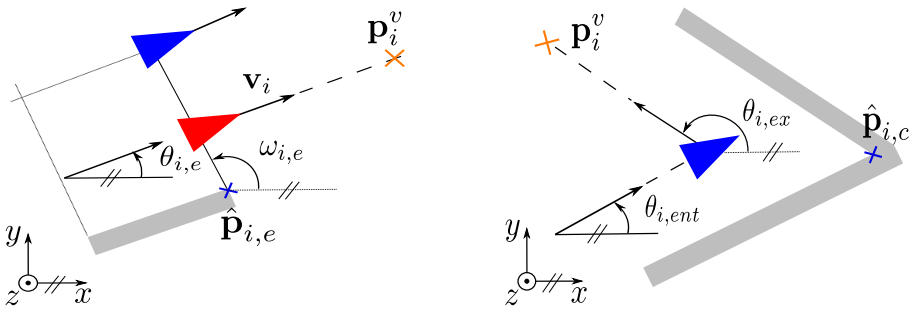
Table 5.1: Definition of action statuses.

$status_i$	Definition
0	Motion toward the desired goal position
1	Obstacle detection and tangential navigation
2	Handling the endpoint of an obstacle
3	Corner avoidance maneuver
4	Orientation phase
5	Tangential navigation based on received information
6	Waiting mode

orientation at the time instant of endpoint detection $\theta_{i,e}$ is another important item in planning the motion of the agent. The agent that identifies an endpoint (shown in red, Fig. 5.6(a)) shares the items $\hat{\mathbf{p}}_{i,e}$, $\omega_{i,e}$, $\theta_{i,e}$ within its neighborhood.

Information about Corners

For collective motion planning, robots require information. An agent stores and communicates its goal orientation during the entry into the corner $\theta_{i,ent}$ and exit from the corner $\theta_{i,ex}$. Another important item in this information package is the corner point $\hat{\mathbf{p}}_{i,c}$ (blue-colored cross in Fig. 5.6(b)), which is determined by calculating the point at which two lines intersect. The agents can calculate the intersection point easily using $\hat{\mathbf{p}}_{i,n}, \hat{\mathbf{p}}_{i,n_{90}}, \theta_{i,ent}$ and $\theta_{i,ex}$. Specifying these parameters allows the group of agents to make a proper collective decision for the corner avoidance maneuver.



(a) Information about an endpoint.

(b) Information about a corner.

Figure 5.6: Information about endpoints and corners.

5.2.2 Evaluation of Information from the Communication Network

In contrast to a single robot navigation, robots in a multi-robot system are exposed to many information pieces at the same time through local communication. For the depth of the information processing, it is not sufficient to evaluate only the position, velocity and heading of neighboring robots. In order to prioritize the acquired information items, each agent evaluates these based on their registration time and sender-related information.

Each agent examines the relevance of the orientation information based on a weighted mean value function and assigns a relevance value *rel* defined in the interval $]-\infty, 10]$. Based on this, each agent makes a decision for its next action in the motion planning. For example, an agent considers an information item with $rel = 10$ as *very relevant*. Moreover, the agent ignores an information item with $rel \leq 0$. In the following, we will define the components of the relevance function with linear sub-functions and discrete expressions.

- **Age of the information:** In order to evaluate the temporal relevance of an information package, we define a linear relationship between the current time t_k and the registration time of the information $\hat{t}^c$ by another agent in the network as follows:

$$rel_t = 10 - \frac{10 \cdot (t_k - \hat{t}^c)}{d_t}, \tag{5.12}$$

  where $d_t \in \mathbb{R}$ is a constant representing the duration time during which an information can have positive relevance.

- **Distance from the detected obstacle:** In order to evaluate the relevance regarding the distance of a communicated obstacle point, it is important to identify the location of the detected point with respect to the agent's motion. If it is located in front of the agent relative to its direction of motion, the distance-based relevance is defined as

$$rel_{dist} = 10 - \frac{10 \cdot \|\hat{\mathbf{p}}^c - \mathbf{p}_i\|}{d_x}, \tag{5.13}$$

  where $d_x \in \mathbb{R}$ is the maximum distance required by $\|\hat{\mathbf{p}}^c - \mathbf{p}_i\|$ for a positive relevance. However, if the point $\hat{\mathbf{p}}^c$ is behind the agent with respect to the agents' motion perspective, the distance relevance obtains a negative value as

$$rel_{dist} = -\frac{10 \cdot \|\hat{\mathbf{p}}^c - \mathbf{p}_i\|}{d_x}. \tag{5.14}$$

  If there is no orientation information in the agent's communication network, the distance relevance is defined as 0.

- **Relevance of the orientation based on the previous time step:** During the tangential navigation, each agent continuously expects only minor changes to

its orientation. For this purpose, each agent compares its current goal orientation $\theta_i(t_k)$ with one of the existing, new orientations θ^c in its communication network. The relevance of expectation of orientation is expressed as

$$rel_{exp} = 10 - \frac{10 \cdot |\theta_i(t_k) - \theta^c|}{d_\theta}, \tag{5.15}$$

where d_θ is the maximum allowed difference between the angles for a positive relevance value.

- **Evaluation of the sender:** Agents also examine the sender of the information. To determine the relevance, it is important to know whether the sender is the owner of the information or just forwards received information from another agent.

$$rel_o = \begin{cases} 10, & \text{for information sent directly} \\ 0. & \text{for information sent indirectly} \end{cases} \tag{5.16}$$

- **Evaluation of information statuses:** Finally, $status^c$ is included in the evaluation. As shown in Table 5.1, an information package contains different statuses that express the sender's current action. The relevance of actions, which represent a reorientation of the agent, is evaluated with $rel_{type} = 10$. However, if an agent is following an obstacle tangentially, then $rel_{type} = 5$. Otherwise, the relevance is zero.

$$rel_{type} = \begin{cases} 10, & \text{if } status^c = 4 \vee status^c = 3 \\ 5, & \text{if } status^c = 1 \\ 0. & otherwise \end{cases} \tag{5.17}$$

We formulate the weighted arithmetic mean of the individual relevance values $\overline{rel}_n$ as follows

$$\overline{rel}_n = \frac{c_{type} \cdot rel_{type} + c_o \cdot rel_o + c_{exp} \cdot rel_{exp} + c_{dist} \cdot rel_{dist} + c_t \cdot rel_t}{c_{type} + c_o + c_{exp} + c_{dist} + c_t}, \tag{5.18}$$

where $n \in \mathcal{S}_i$ and $\mathcal{S}_i$ is the set of all information packages in the communication network of agent i. The parameters c_z represent the weighting factor of the components of the relevance value. Various weighting factors can yield different behaviors in the swarm. Furthermore, the maximum relevance value of the existing information packages in the network is defined as

$$\overline{rel}_{max} = \underset{\overline{rel}_n \in \mathcal{R}_i}{\operatorname{argmax}}(\overline{rel}_n), \tag{5.19}$$

where $\mathcal{R}_i$ is the set of all the relevance values of the available information packages for the agent i. If the following condition holds

$$\overline{rel}_n \geq 0.95 \cdot \overline{rel}_{max}, \tag{5.20}$$

the agent considers an information package to be *relevant* and applies the corresponding goal orientation $\theta_i(t_{k+1}) = \theta^c$ in the next time step t_{k+1}. However, if more than one information package fulfills condition (5.20), the mean value of all corresponding goal orientations from the relevant information packages are determined.

In addition to the introduced relevance criteria, the agent checks its distance to the communicated endpoints and corners as follows

$$\begin{aligned} d_s &\leq \|\mathbf{p}_i - \hat{\mathbf{p}}_e^c\| \leq R_{rel}, \\ d_s &\leq \|\mathbf{p}_i - \hat{\mathbf{p}}_c^c\| \leq R_{rel}, \end{aligned} \tag{5.21}$$

where $R_{rel} \in \mathbb{R}$ denotes the upper boundary of the relevance range. An endpoint or a corner is then considered if it is within a defined relevance range of the agent. Note: If two different critical points are within the specified range, the closer one is utilized for the next action.

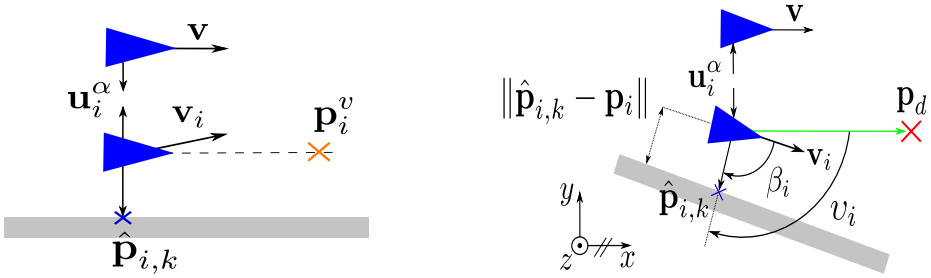
5.2.3 Collective Navigation Using Shared Information

In the navigation approach for a single robot, we defined several scenarios: tangential navigation, corner avoidance, motion planning at obstacle extremities. In this section, we explain how to adapt these to a multi-agent system by systematically communicating the presented action statuses. A detailed explanation of the transitions between the action statuses is given as pseudo codes in Appendix A. At the beginning of the collective navigation, each agent has the relevant information based on the evaluation scheme proposed in Section 5.2.2. The initial status of the agents is defined as **status 0**. An agent with this status follows the desired goal $\mathbf{p}_d$ using the control term in Eq. (4.9) (Algorithm A.1).

5.2.3.1 Collective Tangential Navigation

If an agent detects an obstacle in the range of r_{tan}, it starts to move toward a virtual goal, tangentially to the obstacle. The calculation of the virtual goal is analogous to the navigation algorithm for a single agent. This action is referred to as **status 1** in the communication network and the agent sends information with $status = 1$ to the neighboring agents (Algorithm A.2).

However, the control input $\mathbf{u}_i^{\alpha}$ according to Eq. (4.8) intervenes if the ideal distance d between the agents is exceeded. As a result of this, the agent sensing an obstacle can move away from it through attractive forces and leave the tangential navigation range r_{tan} (Fig. 5.7(a)). This can distort the navigation. In order to prevent this, the



(a) Loss of contact. (b) Incorrect navigation.

Figure 5.7: Illustration of problems due to $\mathbf{u}_i^\alpha$.

agent sensing an obstacle applies only the horizontal component of $\mathbf{u}_i^\alpha$ along the vector $(\mathbf{p}_i^v - \mathbf{p}_i)$. In this way, it is not influenced by forces vertical to the obstacle. The input $\mathbf{u}_i^\alpha$ is reformulated for this case as $\mathbf{u}_i^{\alpha,t}$.

The input $\mathbf{u}_i^\alpha$ can also cause incorrect navigation through an obstacle. If the distance between two adjacent agents is less than the desired distance, $\mathbf{u}_i^\alpha$ generates repulsive forces. This may allow an agent to detect an obstacle unnecessarily and start tracking it (**status 1**). In order to prevent an incorrect transition from **status 0** to **status 1**, we introduced the so-called ignorance condition as follows

$$\begin{aligned}(|\upsilon_i| > 90^\circ \wedge |\beta_i| > 91^\circ) \quad &\vee \\ \|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\| &> (0.3 \cdot (r_{tan} - d_s) + d_s),\end{aligned} \tag{5.22}$$

where υ_i denotes the angle between the vector $(\mathbf{p}_d - \mathbf{p}_i)$ and the normal vector to the obstacle. The first part of Eq. (5.22) ensures that the agent initially moves away from the obstacle and the second part represents an ignorance zone. If condition (5.22) is satisfied, the agent ignores the obstacle for the motion planning and applies the modified input $\mathbf{u}_i^{\alpha,t}$.

Due to previous information received from the communication network, an agent might be inconveniently oriented once it senses an obstacle. In this case, it changes its goal orientation based on the obstacle detected . This can take some time and the orientation phase is described by **status 4** (Algorithm A.5). An agent may have $status = 4$ if it receives new information or detects an obstacle while the following condition is satisfied:

$$|\psi_i - \theta_i| > 45^\circ, \tag{5.23}$$

where ψ_i is the orientation of the agent.[5]

The communication interface allows agents to calculate a virtual goal position based on information received from the communication network. This case is defined by **status**

[5] ψ_i is the angle between the vector $\mathbf{v}_i$ and the x-axis and measured from the positive x-axis.

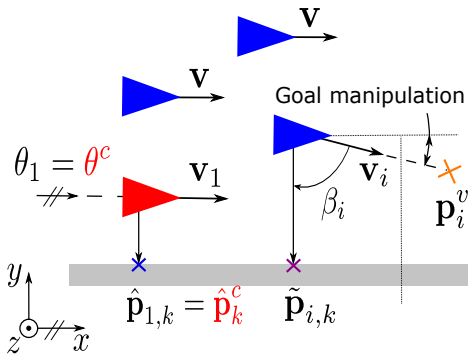


Figure 5.8: Distance manipulation of an agent toward an obstacle.

5 (Algorithm A.6). Based on the communicated goal orientation θ^c, the virtual goal position can be determined by the agent according to the following equation:

$$\mathbf{p}_i^v = \mathbf{p}_i + \begin{bmatrix} \cos(\theta^c) & -\sin(\theta^c) \\ \sin(\theta^c) & \cos(\theta^c) \end{bmatrix} \cdot \mathbf{e}_x \cdot s, \tag{5.24}$$

where $\mathbf{e}_x$ is the unit vector in direction of the x-axis. Agents have a constant tracking acceleration through the positive gain s. In addition, each agent follows an individual temporary goal. The vertical distances among agents are ensured by $\mathbf{u}_i^{\alpha,t}$.

The disadvantage of this approach is that the swarm may perceive the endpoint of an obstacle belatedly. The problem is illustrated in Fig. 5.8. The red-colored agent detects the obstacle and shares the information items θ^c and $\hat{\mathbf{p}}_k^c$ within the network[6]. The first agent follows the obstacle just above the sensing radius r_s, but cannot detect the obstacle itself. Thus, the end of the obstacle can only be perceived belatedly by the red-colored agent. We introduce the following conditions to optimize this behavior.

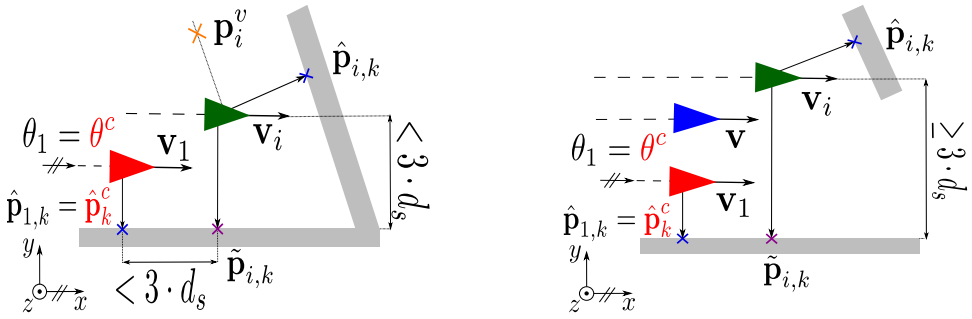
$$\theta_i(t_{k+1}) = \begin{cases} \theta^c - 20°, & \|\tilde{\mathbf{p}}_{i,k} - \mathbf{p}_i\| < 1.5 \cdot d \ \wedge \ \beta_i < 0° \\ \theta^c + 20°, & \|\tilde{\mathbf{p}}_{i,k} - \mathbf{p}_i\| < 1.5 \cdot d \ \wedge \ \beta_i \geq 0° \end{cases} \tag{5.25}$$

where $\tilde{\mathbf{p}}_{i,k}$ is the orthogonally projected position of the agent i onto the obstacle k, which is described by $\hat{\mathbf{p}}_k^c$ and θ^c as a virtual line. In this way, the agent can estimate its distance to the obstacle. The first condition ensures that there is no other agent between the agent and the obstacle. If it is fulfilled, θ_i is manipulated depending on β_i in the next time step.

5.2.3.2 Collective Corner Avoidance

The scenario for avoiding corners described in Section 5.1.2 is represented by **status 3** in the collective navigation (Algorithm A.4).

[6]The red-colored parameters in figures are shared among the neighborhood of agents. Thus, they are known to all agents via communication after some time instants.



(a) Estimation of a corner via communication.

(b) Waiting for decision-making.

Figure 5.9: Collective corner avoidance.

In status 3, an agent detects two obstacle points at the same time. Another scenario is the combination of two information items: a self-detected obstacle point with the range sensor $\hat{\mathbf{p}}_{i,k}$ and a point projected onto an obstacle $\tilde{\mathbf{p}}_{i,k}$ by using shared information items $(\hat{\mathbf{p}}_k^c, \theta^c)$. With the help of these items, an agent builds two intersecting virtual lines and estimates a corner's position (see Fig. 5.9). However, there might be a passage through which the swarm can proceed. In order to prevent such a misestimation, the following condition is introduced:

$$\|\tilde{\mathbf{p}}_{i,k} - \mathbf{p}_i\| < 3 \cdot d_s \ \wedge \ \|\tilde{\mathbf{p}}_{i,k} - \hat{\mathbf{p}}_k^c\| < 3 \cdot d_s. \tag{5.26}$$

If the above condition is fulfilled, the agent assumes that a corner exists (see the green-colored agent in Fig. 5.9(a)). Thus, similarly to Section 5.1.2, it determines $\hat{\mathbf{p}}_{i,n_{90}}$ and $\hat{\mathbf{p}}_{i,n}$ by considering $\tilde{\mathbf{p}}_{i,k}$ and $\hat{\mathbf{p}}_{i,k}$. Then, a new virtual goal position is determined by applying Eqs. (5.6)-(5.8).

In this case, the agent changes its status to 4 and it additionally sends information about the corner. If the conditions (5.26) are not satisfied, the agent stores its current location $\mathbf{p}_i(t_k)$ as $\mathbf{p}_i^{wait}$ and waits at this position until it makes a decision for an orientation by using items from the communication network (Fig. 5.9(b)). The state of waiting is represented by **status 6** (Algorithm A.7). The virtual target is defined in this case as

$$\mathbf{p}_i^v = \mathbf{p}_i^{wait}. \tag{5.27}$$

At *status 6*, the agent evaluates the relevance of information in its neighborhood network. If orientation information with $status = 3$ or $status = 4$ is declared as relevant, the agent applies it and leaves the status of waiting. If, however, information with another status is classified as relevant, the agent examines whether an information about a nearby endpoint is available in the communication network. If it is, the agent applies the direction of rotation of the circular motion around the endpoint and tracks the obstacle. If there is information with $status = 1$ in its network and if the information corresponds to the same obstacle, the agent uses the orientation information and tracks the obstacle.

5.2.3.3 Collective Motion at Obstacle Extremities

If an agent detects the endpoint of an obstacle, it follows the virtual target positions defined on a circular path as described in Section 5.1.3. This navigation approach is represented by **status 2** in the multi-agent setting (Algorithm A.3). Here, each agent individually sets virtual goals along a circular path.

Fig. 5.10 illustrates the described scenario with *status* = 2. The group tracks the obstacle tangentially. The red-colored agent detects the endpoint of the obstacle and shares the information items ($\hat{\mathbf{p}}_{i,e}$, $\omega_{i,e}$, $\theta_{i,e}$) (red-colored variables) within its network. Once an agent receives the information about the endpoint of an obstacle via the communication network, it examines its current location to start a circular motion. For this purpose, firstly, it defines a virtual straight line by using the items received ($\hat{\mathbf{p}}_e^c, \omega_e^c$) from the communication network, the *start line* for the circular motion. By means of orthogonal projection, the front agent, represented in green, can project its current position $\mathbf{p}_i$ onto this line and obtains the point $\mathbf{p}_i^*$ (see Fig. 5.11(a)). The agent evaluates the following condition before it starts a circular motion:

$$(\mathbf{p}_i^* - \mathbf{p}_i) = -k \cdot (\mathbf{p}_i^v - \mathbf{p}_i), \tag{5.28}$$

where $k \in \mathbb{R}_{\geq 0}$ is a constant. The agent uses condition (5.28) to examine whether it has reached the start line. If the condition is **not** fulfilled, the agent continues to set tangential goal positions and adjusts its velocity by replacing $\mathbf{u}_i^\gamma$ with $\mathbf{u}_{i,e}^\gamma$ as follows:

$$\mathbf{v}_{i,ref} = \frac{\mathbf{p}_i^v - \mathbf{p}_i}{\|\mathbf{p}_i^v - \mathbf{p}_i\|} \cdot \frac{d_s}{r_{i,e}^{max}} \cdot v_{max}, \tag{5.29}$$

$$\mathbf{u}_{i,e}^\gamma = -c_e^\gamma (\mathbf{v}_i - \mathbf{v}_{i,ref}), \tag{5.30}$$

where v_{max} represents the predefined maximum allowed speed at the endpoint. In addition, $r_{i,e}^{max}$ denotes the maximum distance of an agent to an obstacle communicated in the neighborhood of agent i. In this way, the speed of the agent is adapted to the minimum possible speed required for the circular motion.

If condition (5.28) holds, the agent starts to perform a circular motion similar to the presented schema in Section 5.1.3. During the circular motion, $\mathbf{u}_i^{\alpha,t}$, the component of $\mathbf{u}_i^\alpha$ along the vector $(\mathbf{p}_i^v - \mathbf{p}_i)$, acts again to ensure that the agent reaches its current goal position on the circular path. In addition, each agent examines the angle between the line linking $(\mathbf{p}_i^v, \hat{\mathbf{p}}_{i,e})$ and the line $(\mathbf{p}_i, \hat{\mathbf{p}}_{i,e})$, defined from the vector $(\mathbf{p}_i^v, \hat{\mathbf{p}}_{i,e})$. If the angle has a positive sign (Fig. 5.11(b), green-colored agent), it means the agent has passed its virtual goal and should set a new goal position by applying Eqs. (5.9)-(5.10). Fig. 5.10 shows that the agents follow an individual circular path with a radius $r_{i,e}$, depending on their distance from the obstacle. As a result, the agents on an inner circular path travel a shorter path than the agents on an outer circular path. The second term in Eq. (4.8) ensures the velocity consensus for a translational motion. For the case of a circular motion, the consensus is based on the equality of the angular

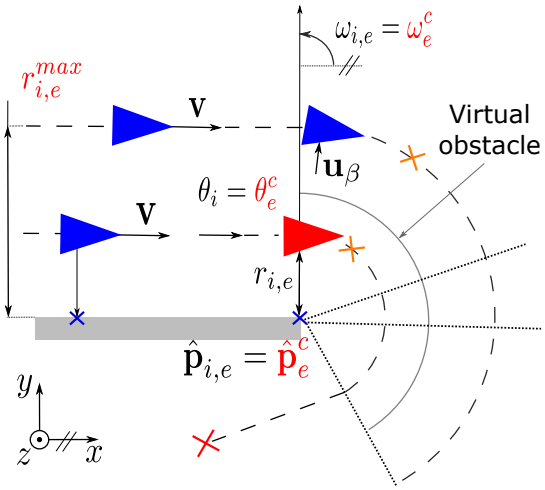


Figure 5.10: Collective maneuver at the endpoint of an obstacle.

velocities. Thus, in the state of consensus, the minimum desired angular velocity of agents is defined as

$$\omega_{min} = \frac{v_{max}}{r_{i,e}^{max}}, \tag{5.31}$$

where $r_{i,e}^{max}$ represents the maximum radius for the circular motion. The navigation input $\mathbf{u}_i^\gamma$ is replaced by $\mathbf{u}_{i,e}^\gamma$ described in the following steps

$$\mathbf{v}_{i,ref} = \frac{\mathbf{p}_i^v - \mathbf{p}_i}{\|\mathbf{p}_i^v - \mathbf{p}_i\|} \cdot \omega_{min} \cdot r_{i,e}, \tag{5.32}$$

$$\mathbf{u}_{i,e}^\gamma = -c_e^\gamma(\mathbf{v}_i - \mathbf{v}_{i,ref}), \tag{5.33}$$

where $\mathbf{v}_{i,ref}$ is the velocity required for the consensus of the angular velocities.

Proposition 5.1. *A group of agents $G(\mathbf{p}(t))$ with $status_i = 2$ applies (4.8) and (5.33). Assume that $G(\mathbf{p}(t))$ is a connected, undirected graph based on positions $\mathbf{A} = (a_{ij}(\mathbf{p}))$ and each agent moves tangentially to its circular path so that the angular velocity of each agent can be described approximately as*

$$\omega_i = \frac{\|\mathbf{v}_i\|}{r_{i,e}}, \tag{5.34}$$

then

$$\lim_{t\to\infty} \omega_i(t) = \omega_{min}, \quad \forall i \in G, \tag{5.35}$$

i.e., consensus of angular velocities is achieved.

Proof. For the stability analysis of the system, analogous to the proof of [100, Theorem 2] (cf. Section 4.2.3), we use moving reference states defined as:

$$\tilde{\mathbf{p}}_i = \mathbf{p}_i - \mathbf{p}_c \quad \text{and} \quad \tilde{\omega}_i = \omega_i - \omega_c,$$

where $\mathbf{p}_c = \frac{1}{N}\sum_{i=1}^{N} \mathbf{p}_i$ is the average position and ω_c denotes the angular velocity of the flock center point. The Hamiltonian representing the structural energy of the system with $status_i = 2$ can be defined as

$$H(\tilde{\mathbf{p}}, \tilde{\omega}) = U(\tilde{\mathbf{p}}) + K(\tilde{\omega}), \tag{5.36}$$

where $K(\tilde{\omega})$ is the kinetic energy and $U(\tilde{\mathbf{p}})$ denotes the aggregate potential function (cf. [100]). Since we consider a rotational motion, we can define the kinetic energy as

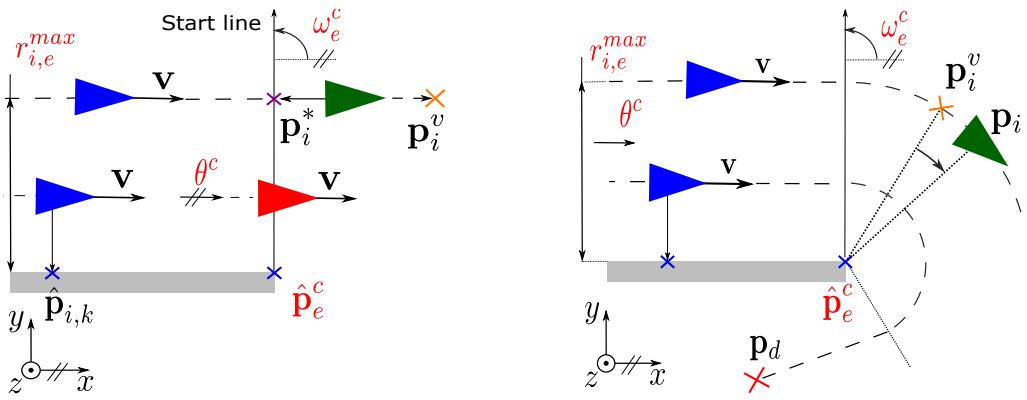
$$K(\omega) = (1/2)\sum_{i} \omega_i^2. \tag{5.37}$$

Assume that $\mathbf{v}_{i,ref}$ and $\mathbf{v}_i$ are tangential to the circular path if the angle $\delta/2$ is small enough to apply a small-angle approximation. The structural energy of the system should be monotonically decreasing. The derivation of structural energy of the system can be defined as

$$\dot{H} = -c_e^{\gamma}(\tilde{\omega}^{\top}\tilde{\omega}) - \tilde{\omega}^{\top}\hat{\mathbf{L}}\tilde{\omega} < 0, \tag{5.38}$$

with $\tilde{\omega} = (\tilde{\omega}_1, \tilde{\omega}_1, \tilde{\omega}_2, \tilde{\omega}_2 \ldots \tilde{\omega}_N, \tilde{\omega}_N)^{\top} \in \mathbb{R}^{2N}$ and $\hat{\mathbf{L}} \in \mathbb{R}^{(2N \times 2N)}$ being a positive semidefinite Laplacian matrix (see (2.27)).

From LaSalle's invariance principle, $\dot{H} = 0$ implies that $\tilde{\omega} = \mathbf{0}$. Therefore, the angular velocity of all agents asymptotically matches $\tilde{\omega}_i = 0 \rightarrow \omega_i = \omega_{min}$. ■



(a) Crossing the virtual start line.

(b) Inspection during the circular motion.

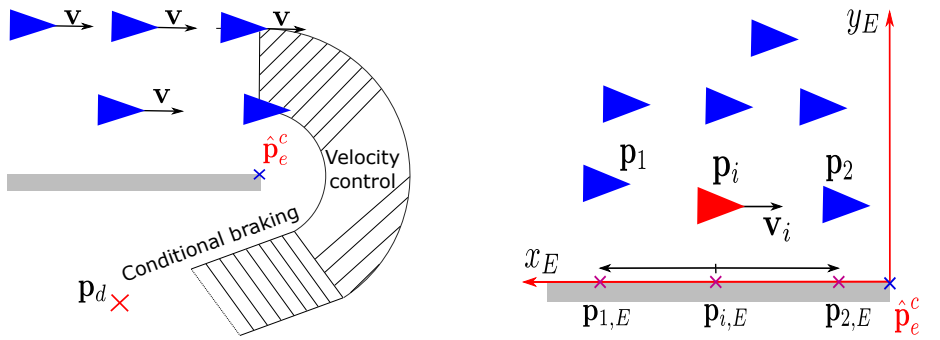
Figure 5.11: Motion planning on a circular path.

Conditional Braking

Each agent leaves the circular path by satisfying condition (5.11) at different time instants. The acceleration toward $\mathbf{p}_d$ after the circular motion with controlled angular velocity might yield a separation of the system. In order to optimize this behavior, we integrated a conditional braking mechanism that supports the cohesive motion of the group. This is illustrated in Fig. 5.12(a).

The agents use d_s to calculate the necessary speed that is the lowest possible speed during the circular motion of the swarm. However, they have to determine the *backmost agent* in the group, which is identified by the index $\bar{b}$. The conditional braking is only canceled by the *backmost agent* once it has completed its circular motion. For this purpose, the communication interface is extended by the variables $(\bar{b}, c_{reset})$. The variable c_{reset} denotes the signal for the end of the reformation phase, ($c_{reset} = 0$: velocity control for the reformation, $c_{reset} = 1$: end of the reformation). This signal is communicated by the backmost agent.

In order to identify the *backmost agent*, first, we define a new coordinate system E with the origin at $\hat{\mathbf{p}}_{i,e} = \mathbf{0}_E$, as shown in Fig. 5.12(b). The orientation of the axis of the new coordinate system is given by the two communicated angles $\theta_{i,e}$ and $\omega_{i,e}$. For further analysis, the current positions of the agents are projected onto the x-axis of the new coordinate system E. The projected position of the agent onto x_E is referred to as $\mathbf{p}_{i,E}$, and the neighboring agents, referred to as $\mathbf{p}_{j,E},\ j \in \mathcal{N}_i^{\alpha}$ (see Fig. 5.12(b)).



(a) Illustration of the velocity control and the conditional braking regions.

(b) Identification of the backmost agent of a group.

Figure 5.12: Conditional braking mechanism.

During the whole circular navigation, each agent examines its neighborhood. In order to determine the backmost member, all agents utilize the index-dependent function given as

$$d(i) = \begin{cases} \|\mathbf{p}_{i,E} - \mathbf{0}_E\|, & \text{if } (\mathbf{p}_{i,E} - \mathbf{0}_E) = k \cdot \mathbf{e}_{x_E} \\ -\|\mathbf{p}_{i,E} - \mathbf{0}_E\|, & \text{if } (\mathbf{p}_{i,E} - \mathbf{0}_E) = -k \cdot \mathbf{e}_{x_E} \end{cases} \tag{5.39}$$

where $\mathbf{e}_{x_E}$ denotes the unit vector in the positive direction of x_E and $k \in \mathbb{R}_{>0}$ is a positive constant.

Proposition 5.2. *Each agent can determine the index of the global backmost member $\bar{b}$ with respect to the motion of direction in the whole network $G(\mathbf{p}(t))$ if the graph $G(\mathbf{p}(t))$ is connected and undirected.*

Proof. Let $b_i(t_k)$ be the index of the determined backmost agent in the communication network of agent i at the time instant t_k and $\mathcal{B}_i$ be the set of all indices in the neighborhood of agent i. For $t_k = 0$ at which the first information about an endpoint is registered, $b_i(0) = i$. Each agent applies a gossip-like communication:

$$b_i(t_{k+1}) = \underset{b_j \in \mathcal{B}_i}{\operatorname{argmax}}\, d(b_j(t_k)), \quad j \in \mathcal{N}_i^{\alpha}, \tag{5.40}$$

so that the index of the determined backmost member with the greatest d in the neighborhood is adopted. By iterating (5.40), each agent determines the index of the global backmost member in the whole network,

$$\lim_{t_k \to \infty} b_i(t_k + 1) = \bar{b}, \quad \forall i \in G.$$

■

Once the global backmost agent identified completes its circular motion, it sends a reset signal $c_{reset} = 1$. The rest of the agents break the conditional braking once they receive the reset signal through the communication.

Watchdog Timer

During the collective navigation, some agents can grind to a halt due to communication errors. In this case, the agent hardly moves and cannot reach the target. To recover such malfunctions, each agent is equipped with a self-monitoring mechanism, a so-called watchdog timer. It is activated if the speed of the agent falls below a threshold, e.g., $\|\mathbf{v}_i\| < 0.3$.

When the watchdog timer is activated, the agent saves its current position $\mathbf{p}_{i,wd} = \mathbf{p}_i(t_{i,wd})$ with the time instant $t_{i,wd}$. If the agent moves 1.5 further than the saved location,

$$\|\mathbf{p}_{i,wd} - \mathbf{p}_i(t_k)\| \geq 1.5,$$

and the watchdog timer is still activated, it is deactivated again because the agent assumes that a possible deadlock has broken out.

If no deactivation signal is registered after $15\,s$ and the agent has moved very little compared to its position at the watchdog activation time,

$$\|\mathbf{p}_{i,wd} - \mathbf{p}_i(t_k)\| < 1.5,$$

status 0 is activated and the agent moves toward the desired goal position.

One drawback of this feature is the fragmentation of the swarm. However, the watchdog timer ensures that each agent reaches the desired goal in finite time despite an environment or communication-based deadlock.

5.3 Simulation Studies

In this section, we present the simulation results of the proposed collective navigation approach. We consider two scenarios for demonstrating the effectiveness of the proposed method. The simulation parameters for all scenarios are given in Table 5.2 and the step size is $0.02\,s$. There is no predefined primary direction of rotation for the tangential navigation. The navigation task involves $N = 12$ holonomic agents. They are randomly placed in the domain $[-10, 10]^2$ and the initial velocity $\mathbf{v}_i(0) \in \mathbb{R}^2$ of each agent is $(0, 0)^\top$.

Table 5.2: Parameters for the simulation.

Flocking and Navigation				**Communication**	
d	7	δ	16°	d_t	$0.6\,s$
r_c	$1.2 \cdot d$	c_1^α	20	d_x	15
d_s	$0.6 \cdot d$	c_1^β	80	d_θ	90°
r_s	$5 \cdot d_s$	c_1^γ	30	c_{type}	5
r_{tan}	$1.2 \cdot d_s$	c_2^α	$2 \cdot \sqrt{c_1^\alpha}$	c_o	1
ε	0.1	c_2^β	$2 \cdot \sqrt{c_1^\beta}$	c_{exp}	1
a	5	c_2^γ	$2 \cdot \sqrt{c_1^\gamma}$	c_{dist}	1
b	5	c_e^γ	50	c_t	5
h_α	0.2	v_{max}	4	R_{rel}	30
h_β	0.3	s	100		

In the first scenario, a zigzag obstacle is considered, which might cause the problem of local minimum with potential field-based approaches. The desired goal position is defined at $\mathbf{p}_d = (150, 0)^\top$. Figure 5.13 shows the consecutive snapshots and trajectories the robots followed during navigation for important time instants. The blue triangles represent the position of robots, and the heading of each triangle specifies the direction of motion of each robot. In addition, the illustrated lines linking robots denote the proximity of agents. At $t = 15.8\,s$, an obstacle is detected by the front agents and the group starts the tangential navigation along the border of the obstacle. At $t = 25.4\,s$,

an agent identifies a corner and through communication, the rest of the group changes its orientation. Then, an endpoint is perceived and the group performs a circular motion (Fig. 5.13(c) and Fig. 5.13(d)). At such obstacle extremities, robots reduce their speeds to achieve a reliable circular motion by preserving cohesion and collision-avoidance properties. The average speed curve of the swarm is given in Fig. 5.14. Red curves demonstrate the behavior of the group at obstacle extremities where the robots slow down. Furthermore, at $t = 138\,s$, it can be observed that the front agents have approached the obstacle systematically by applying Eq. (5.25) to optimize the navigation.

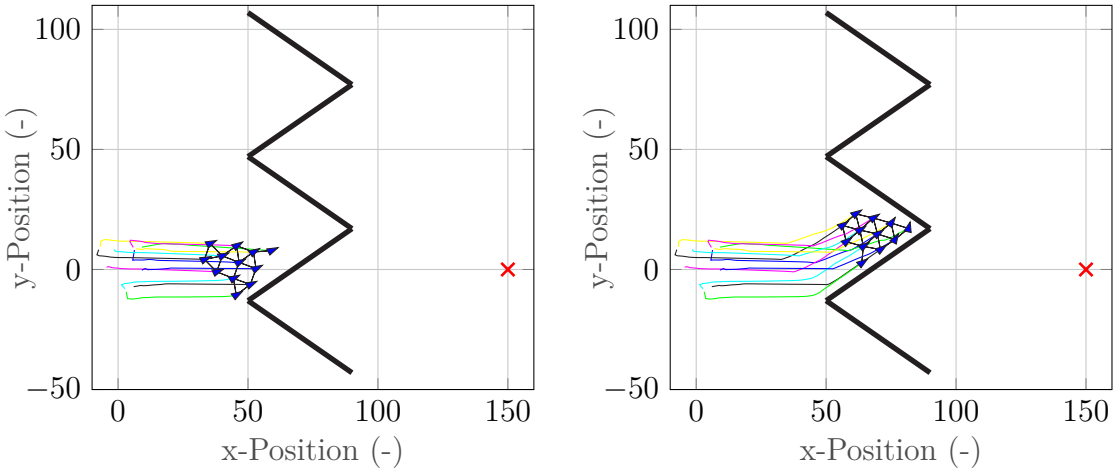
Fig. 5.15 shows the second selected scenario, in which the same group of robots navigates through a corridor with obstacles. The initial state of robots is the same as in the first scenario. The coordinates of the desired goal position are $\mathbf{p}_d = (210, 5)^\top$. The capability of robots to rotate to any direction when encountering obstacles enables agility and flexibility. It is clear that the robots can successfully avoid the obstacles and find a proper path toward the desired goal position applying the collective navigation approach.

Guideline for Parameter Choice

There are further remarks on the proposed approach that might be useful for the application of our framework. In this approach, the agents share only critical points for motion planning instead of constructing a full map of the workspace by storing all utilized sensing points. This releases the memory of agents. One can easily integrate a memory function to the approach to store all sensed points, or critical points (corners, endpoints) as proposed in [28]. Storing and utilizing critical points would improve motion planning in labyrinths. For more detail, the reader is referred to [28].

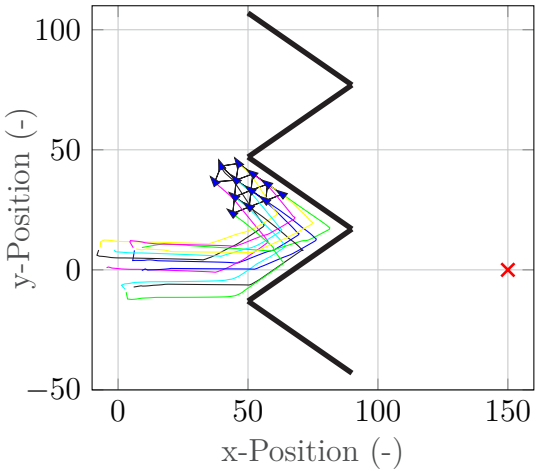
Our simulation analyses have revealed that the parameters of the relevance function (d_t, d_x, d_θ) and the weighting factors $(c_{type}, c_o, c_{exp}, c_{dist}, c_t)$ (see Eq. (5.18)) have a significant influence on the collective behavior of the multi-agent system.

The parameters chosen for our simulations are suitable for small-sized multi-agent systems. In this study, a system with 6-12 agents is defined as small-sized depending on the space between obstacles. The parameter set in this case can be selected so that the MAS exhibits a **global** behavior. This means that all agents perform identical actions approximately at the same time. If, for example, an agent in the group detects a corner, all agents make similar navigation maneuvers. This global behavior is suitable if the distribution of agents in the workspace is small with respect to the space between the obstacles. In order to have a global system behavior, the evaluation of the temporal relevance and status of the information are essential for the next action. Hence, the weighting factors of these relevance values, c_t and c_{type}, should be chosen higher than the rest. Highly prioritizing the temporal relevance of an information package (age of information - rel_t) ensures that an agent will consider mainly current information packages. Moreover, assigning a high priority to the evaluation of the actions of the neighboring agents rel_{type} ensures that all agents in the group will consider informa-

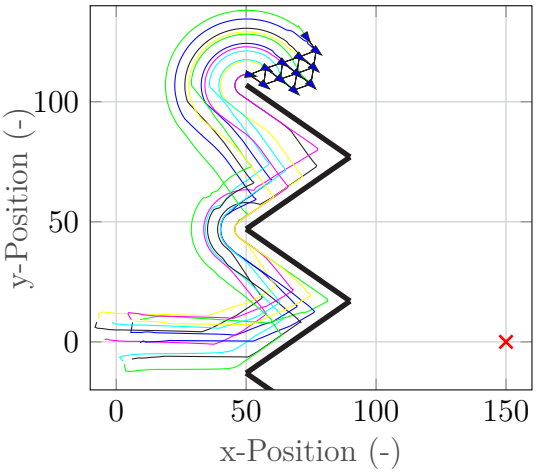


(a) $t = 15.8\,s$

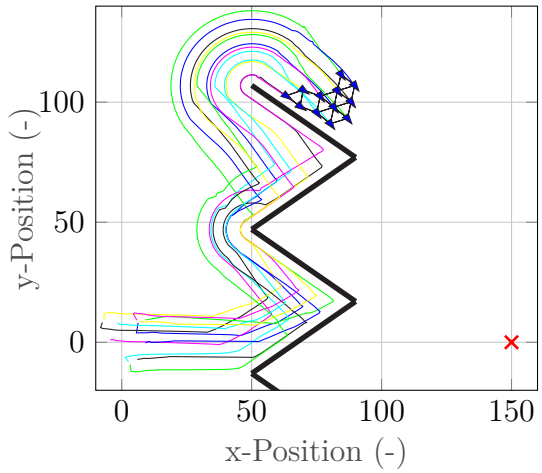
(b) $t = 25.4\,s$



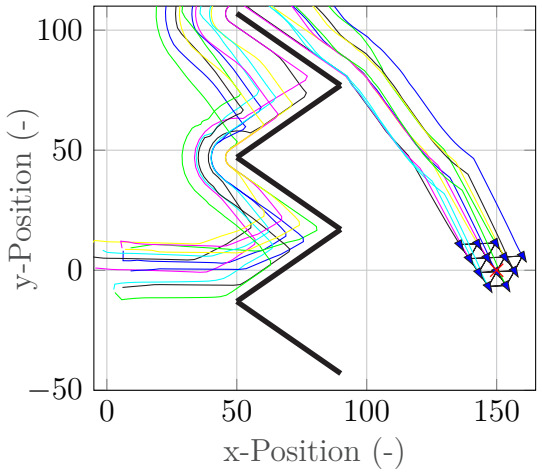
(c) $t = 36.2\,s$



(d) $t = 130\,s$



(e) $t = 138\,s$



(f) $t = 205\,s$

Figure 5.13: Consecutive snapshots of the collective navigation with $N = 12$ agents escaping from a zigzag obstacle.

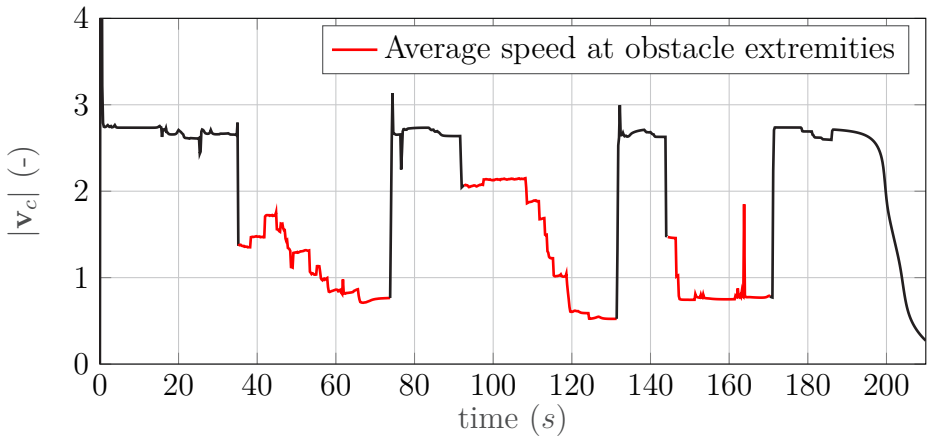
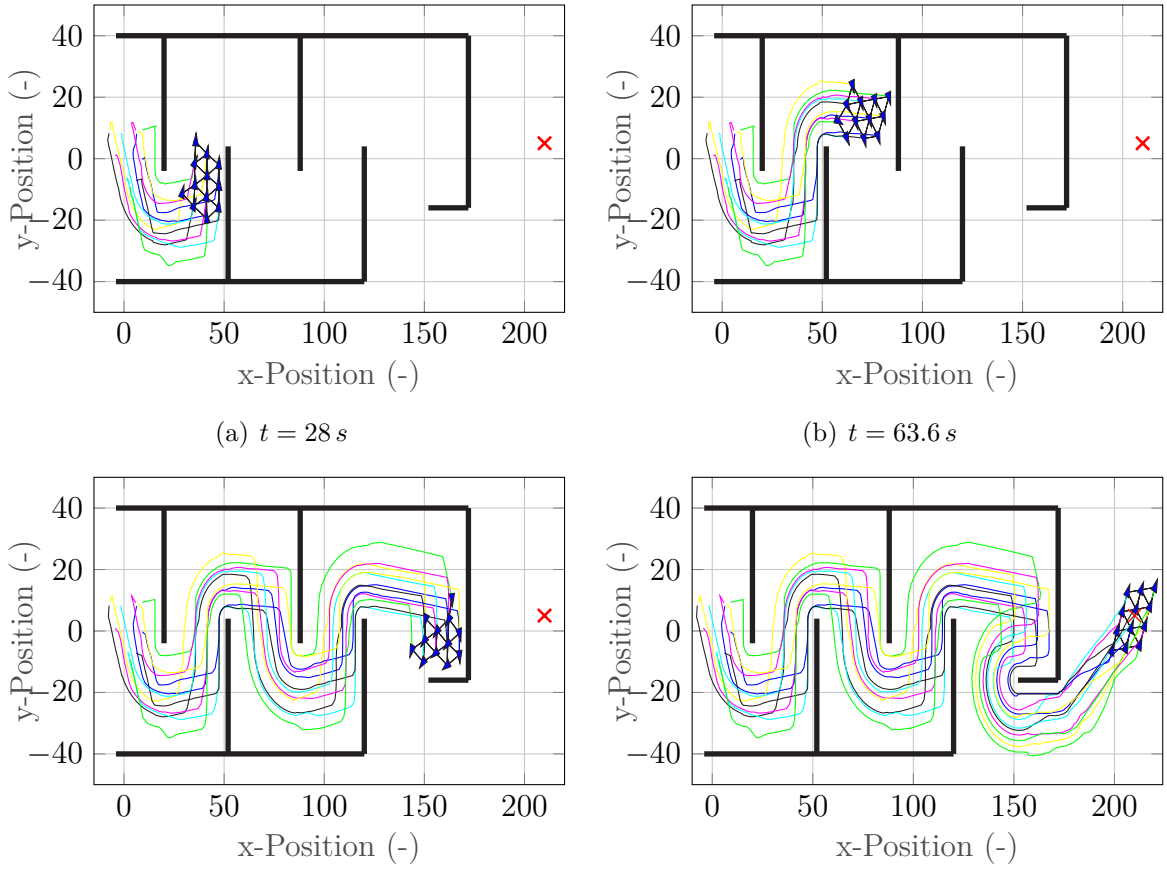


Figure 5.14: Average speed of the swarm $|\mathbf{v}_c|$ during the escape from the zigzag obstacle.



(a) $t = 28\,s$ (b) $t = 63.6\,s$

(c) $t = 156.6\,s$ (d) $t = 240\,s$

Figure 5.15: Consecutive snapshots of the collective navigation with $N = 12$ agents navigating through a corridor.

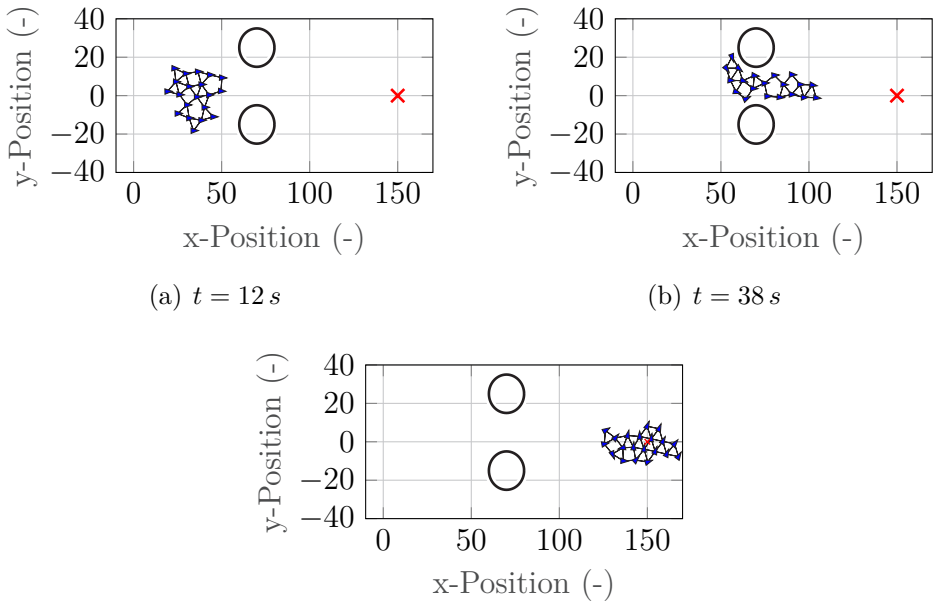
tion packages, which impose a reorientation. In this way, a group of agents can avoid obstacles in a collective manner.

However, agents in large groups, e.g., with 20 agents for our obstacle configuration, often have many relevant information packages at the same time. This means that a global decision might yield suboptimal behaviors. Hence, the agents have to make proper, **local** decisions. In this case, the evaluation of the information relevance should be predominantly based on the individual location of an agent. For this purpose, the relevance of the distance to the obstacle rel_{dist}, the relevance of the expected orientation for the next action rel_{exp} and the relevance of the sender rel_o should be weighted more than rel_{type} and rel_t. This yields the example parameter set given in Table 5.3: $c_{dist} = c_{exp} = c_o > c_t > c_{type}$. Taking this into account would allow an agent to give greater consideration to information packages received from agents close by and yield only a small deviation from the expected orientation of motion.

Table 5.3: Parameters for large systems.

Communication	
d_t	$0.6\,s$
d_x	15
d_θ	$90°$
c_{type}	5
c_o	1
c_{exp}	1
c_{dist}	1
c_t	5
R_{rel}	30

Using the communication parameters in Table 5.3, we perform tests with $N = 20$ agents considering further scenarios, which include a narrow passage, a semi-circular obstacle and many small obstacles. The initial state of robots are chosen in an identical way as in the previous simulations and the desired goal position is defined at $\mathbf{p}_d = (150, 0)^\top$. Figure 5.16 shows an example for a squeezing maneuver. In this configuration, agents perceive both obstacles via communication and the group can avoid them without splitting up. In Figure 5.17, we deal with a semi-circular obstacle. Until the agents arrive at the endpoint, they apply the regular maneuver of tangential navigation. However, apparently at $t = 58\,s$, a communication error occurs because of the complex communication structure and the group splits up. This can also be seen in the video at `https://youtu.be/EYEl35pD4H0`. By checking action statuses, agents that ignore the endpoint correct their direction of motion and return to the right direction. In this



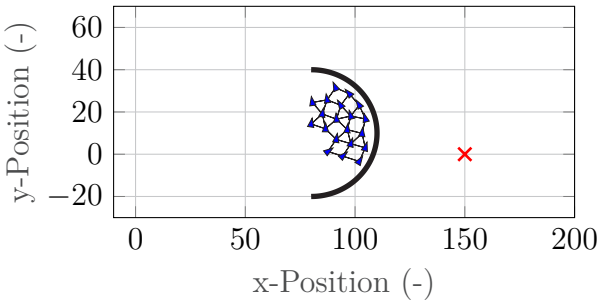
(a) $t = 12\,s$ (b) $t = 38\,s$

(c) $t = 70\,s$

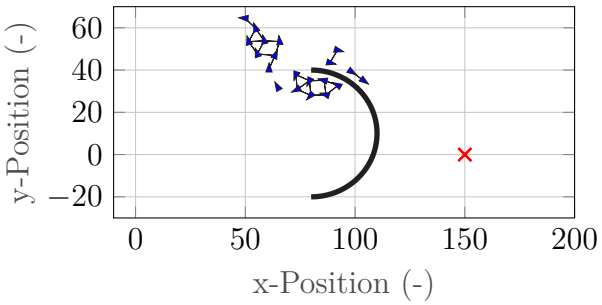
Figure 5.16: Consecutive snapshots of the collective navigation with $N = 20$ agents - two circular obstacles.

way, they can nevertheless complete navigation as a small group. In Figure 5.18, we are concerned with path planning through small circular obstacles, which are narrowly spread in the workspace. Hereby, reacting to different obstacles yields fragmentation of the group. This is a natural behavior because agents are exposed to many different information pieces about the environment and their priority is to avoid collisions with these obstacles. Finally, they still manage to reach the desired goal.

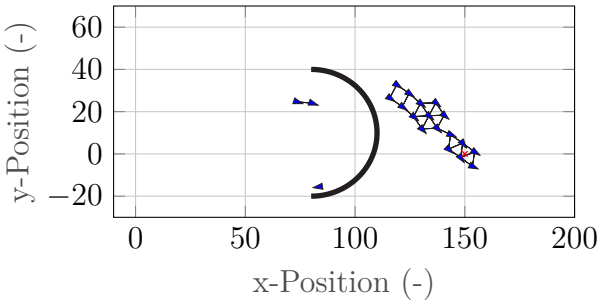
For a further qualitative impression, videos of all simulation scenarios are available at `https://youtu.be/EYEl35pD4H0`.



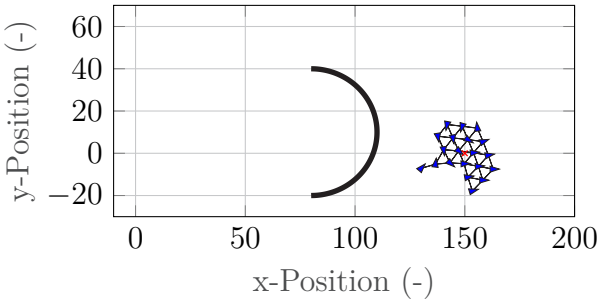
(a) $t = 38\,s$



(b) $t = 58\,s$

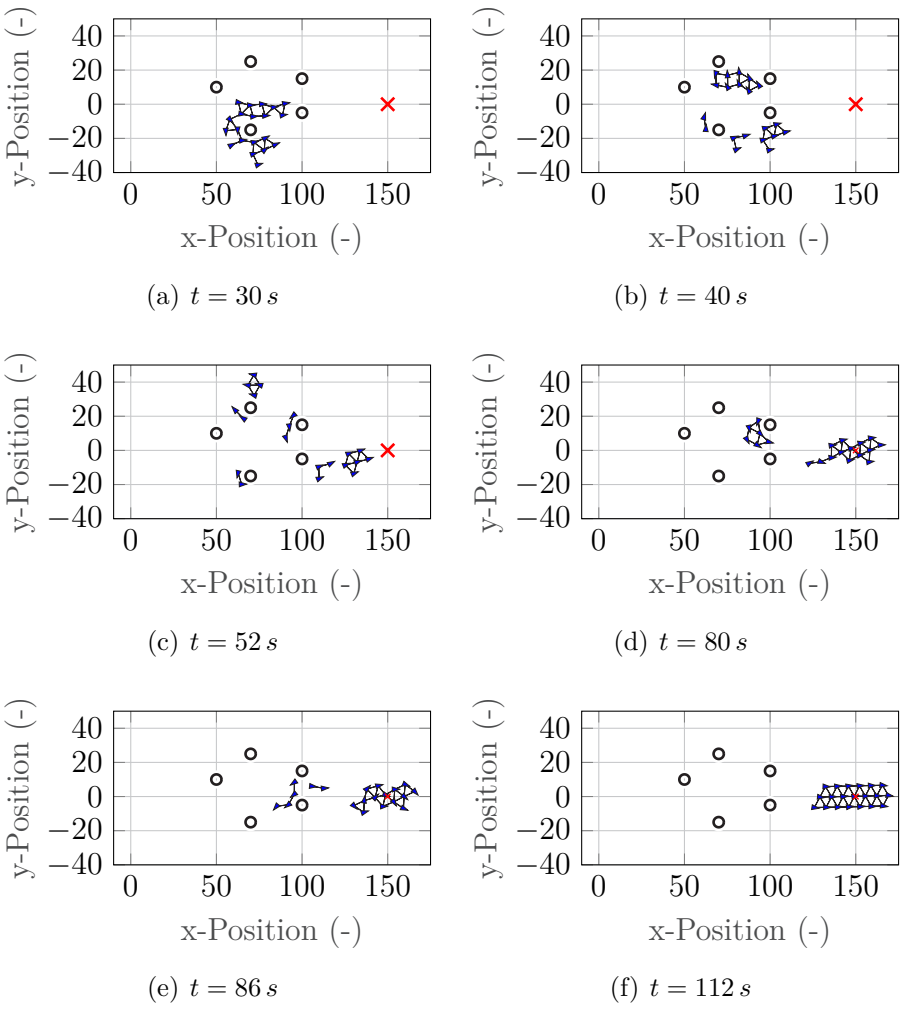


(c) $t = 96\,s$



(d) $t = 150\,s$

Figure 5.17: Consecutive snapshots of the collective navigation with $N = 20$ agents - semi-circular obstacle.



(a) $t = 30\,s$ (b) $t = 40\,s$

(c) $t = 52\,s$ (d) $t = 80\,s$

(e) $t = 86\,s$ (f) $t = 112\,s$

Figure 5.18: Consecutive snapshots of the collective navigation with $N = 20$ agents - small circular obstacles.

5.4 Chapter Highlights

In this chapter, we presented our collective navigation strategy for swarming, autonomous robots without a priori knowledge about the environment. The proposed framework is based on a tangential escape schema and information sharing via communication network. The communication protocol allows multiple robots to efficiently explore an unknown area by exchanging local information about critical points and actions of neighboring robots. Moreover, artificial forces generated through potential fields allow the robots to perform collision-free, cooperative maneuvers and a flock-like behavior.

The groups with a large dispersion relative to the navigation area can tend toward fragmentation. In order to optimize this behavior, another parameter set for constants in relevance function and weighting factors can be chosen. Furthermore, in some cases a single agent might lose the connection to the swarm or to a fragment thereof. However, it can still navigate as a single agent and reach the defined goal position on its own. The proposed navigation approach can also be used for the motion planning of nonholonomic robots. However, it is important to select a large enough safety distance to obstacles and to other robots, and the robots should be operated at a low speed to prevent possible collisions.

The supervised student theses [166] and [162] have contributed to the development of this chapter's results.

6 Cooperative Exploration with a Sensor Network

Parts of the following chapter have been published in [157].

Efficient area coverage poses a great challenge in many fields. Applications range from environmental monitoring, surveillance systems, floor cleaning and lawn moving in the consumer sector to the search for survivors in natural disasters. Especially in the coverage of unknown areas that have obstacles or inaccessible domains, the generation of maps is of importance [44, 83]. In recent years, mobile sensor networks (MSN) have received considerable attention due to their flexibility and good scalability for a relatively low cost. The reasons for this are the progress made in the development of communication technologies and in energy storage components [6, 149, 151].

The ability of robots to localize themselves in a map and to plan elaborated motions are the basics of many coverage approaches. Cooperative, multiple robots can be employed in order to accelerate coverage-oriented missions. Deployment of multi-agent systems for area exploration increases the efficiency and success in many missions. However, decentralized coordination of multiple agents requires the communication ability, algorithms for cooperative motion and decision-making through information exchange. The advantages of a mobile, distributed system compared to a stationary, centralized one are of universal applicability as well as the high responsiveness to disturbances and environmental changes [93, 149, 50].

There are several approaches for area coverage with single or multiple agents. Some of these methods are based on optimal control [94], game theory [119] and reinforcement learning [3]. In dynamic area coverage, mobile agents have limited sensor ranges. Therefore, efficient motion planning is the key for mission success. The anti-flocking approach is an option for multiple agents. In anti-flocking algorithms, agents try to move away from each other to improve coverage and explore new spaces in contrast to the flocking behavior of multiple agents [91].

Contribution: In this chapter, we consider a group of mobile agents with limited communication bandwidths and sensing performance. As a result of that, they can communicate only with the agents within their communication range and detect objects in their sensing range. The main contribution is cooperative identification and coverage of an unknown environment with multiple autonomous agents. Our approach builds upon the anti-flocking algorithm [51] and extends it with improved obstacle avoidance and recognition of inaccessible areas. In this chapter, we propose a sensor-based framework

to cover a given space simultaneously with multiple mobile agents in a cooperative fashion without any prior knowledge of the environment. Through local information exchange and judicious path planning for the recognition of restricted areas inside of obstacles in the exploration domain, the agents are capable of avoiding collisions with differently shaped obstacles and autonomously constructing a map of the whole area by identifying inaccessible domains in the map.

The chapter is organized as follows: Section 6.1 explains the mechanisms of control that provide the networked agents the desired dynamics for coverage-oriented motion planning. Particularly, anti-flocking and the map generation are relevant for this chapter. Section 6.2 describes the investigated problem in sensor-based area coverage. In Section 6.3, the proposed cooperative exploration scheme is presented. This includes collective motion planning and map creation. In Section 6.4, we reuse the tangential navigation scheme from Chapter 5 for motion planning around obstacles. Section 6.5 introduces an algorithm for the identification of inaccessible areas in the workspace by interpreting the sensor data. This section is partly based on [157], a contribution by the author of this thesis, which is improved with a new approach. The proposed exploration scheme is evaluated through simulation scenarios in Section 6.6.

6.1 Mobile Sensor Networks

Mobile Sensor Networks (MSN) are self-organized and dynamic multi-agent systems, in which mobile agents have resources to perceive their environment and communicate with each other. In coverage tasks, the objective is often to explore an unknown area (abbreviated EA for *Exploration Area*) and additionally, to find targets with unknown positions in an efficient way [91, 53]. The focus in area coverage approaches can be on the time efficiency [148], the energy efficiency [149, 3], the avoidance of various obstacles, or on the ability to track dynamic goals [147, 129].

Motion behavior of MSNs can be implemented in three ways: *Flocking, anti-flocking* and *semi-flocking*. The objective of flocking algorithms for MSNs is to match the positions and velocities of the agents during the exploration. In contrast to this, anti-flocking algorithms aim to achieve the highest possible efficiency in exploration by minimizing overlapping search regions [91]. The advantages of both approaches, high coverage rate and good tracking behavior, can be combined by semi-flocking algorithms [128].

6.1.1 Anti-Flocking

The anti-flocking concept is inspired by the behavior of *solitarily* living animal species, such as tigers and spiders, which spend most of their lives alone. The motivation behind the widespread distribution of such animals over a large area is to minimize overlapping of their hunting grounds and thus increase the success in finding food. In nature, communication and delimitation of the individual territories often takes place via scents [91]. Anti-flocking can thus be described as cooperative behavior in which

all individuals benefit from the optimal use of the given resources [50]. With this motivation, [91] introduced three rules for engineering applications of the anti-flocking concept:

1. *Collision avoidance*
2. *De-centering*
3. *Selfishness*

Analogously to flocking, *Collision Avoidance* describes the avoidance of collisions with other agents. *De-centering* is the effort to distribute all agents as far as possible over a given area. If the exploration regions of two individuals overlap, they strive to increase the distance between them. *Selfishness* defines the struggle of an agent to increase its chance for success with respect to a previously defined goal, e.g., area coverage. Thus, each agent plans its motion to obtain the greatest benefit.

In order to maximize area coverage, so-called anti-flocking algorithms for mobile sensor networks were proposed [147, 148, 128]. These algorithms distribute agents spatially in a systematic way. The combination of anti-flocking with the potential field approach enables agents to detect targets in a large area by means of communication and to avoid obstacles [51, 52]. Due to battery limitations, energy efficiency is an issue in area coverage tasks, which was considered in studies [149, 70].

Area coverage can be performed with decomposition-based approaches through partitioning of the exploration domain. These methods include Voronoi diagram [66, 108, 90], Boustrophedon decomposition [70] and Morse decomposition [1]. Furthermore, there are ergodicity-based frameworks that consider target probability distribution [126, 14, 64].

6.1.2 Map Generation

In order to implement the property of *selfishness*, the individual benefit values for the coverage of the largest possible area in different directions have to be calculated. For this purpose, it is useful to introduce a so-called *information map*[1] for each agent [50]. This map stores the information about which parts of the EA have already been explored by the agent and which parts are still unexplored. In order to do this, the EA is usually partitioned. In this work, a grid of rectangular or square cells is utilized for this purpose. In the information map, which is implemented as a matrix, each element represents a cell of the discretized EA. The elements of the matrix include the information whether - and if so, when - the corresponding cell of the EA was last visited [53].

Fig. 6.1 shows a decomposed area. All cells whose center is within the blue-colored circle around the agent are marked as covered. In MSNs, the agents often have the ability to communicate with each other. In this way, each agent can update its information map by exchanging missing information with the agents within its communication range to avoid multiple coverage of identical cells. If local communication is integrated to

[1]The information map in this chapter differs from the one presented in Chapter 4.

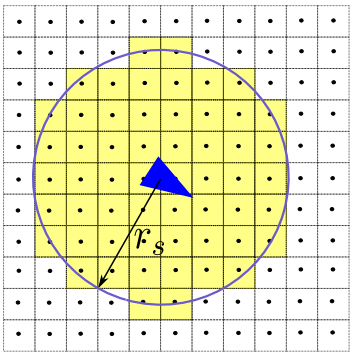


Figure 6.1: Visual representation of area sensing. The yellow-colored cells are identified by the agent.

the motion planning, the overlapping of explored areas with multiple agents can be drastically reduced.

6.1.3 Decomposition-Based Methods

In the literature, different decomposition methods for area coverage and exploration tasks exist. Decomposition algorithms split an area into subareas to increase the coverage efficiency. One example of this is the *Voronoi decomposition* approach, which assigns to each agent a subarea to explore depending on its position [67]. Some further approaches are the *Morse* and *Boustrophedon* decomposition algorithms.

Voronoi Decomposition

A classical partitioning according to Voronoi splits an area into subareas close to given center points c_i. For each center point, there is a corresponding subarea consisting of all points of this subregion to that center point than to any other [109]. In the context of MSNs, the center points of the Voronoi cells correspond to the positions of the individual agents.

For an efficient area coverage, the initial distribution of agents over the area plays an important role. This problem is shown in Fig. 6.2. In the left half of the area, where the centroids are more widely distributed, the cells become very large, while the cell size in the right half is smaller because of the higher cell density. Assuming that all agents have identical resources due to the heterogeneous initial distribution, the agents on the right cover their territory faster than the agents on the left. This basic approach was adapted by [67] to scenarios with inhomogeneously distributed multi-targets in an EA. Hereby, a so-called *density function* $\varphi(x, y)$ over the region is introduced to quantify the relevance of each subarea. The density function describes the probability of finding a target in a subarea. Applying this, Voronoi cells that are located in areas with high density values are designed smaller to cover these more efficiently.

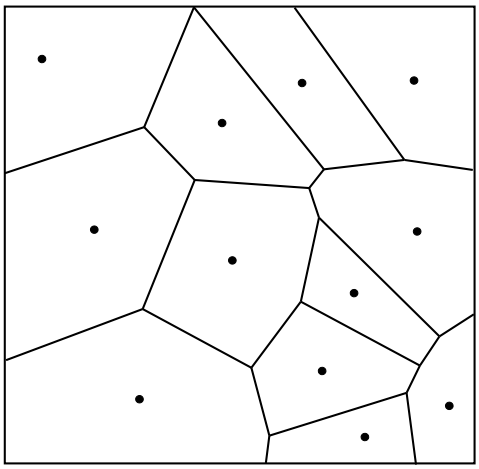


Figure 6.2: Voronoi decomposition adapted from [67].

Moreover, [109] proposed a further approach to optimize the Voronoi decomposition for MSNs considering varying performance of the individual agents. For this purpose, the size of a subarea assigned to an agent is determined depending on its resources, such as battery life or sensor range.

Boustrophedon and Morse Decomposition

Boustrophedon and Morse decomposition approaches aim at the optimal exploration of areas occupied by obstacles. Critical points of the obstacles, through which the boundaries of individual partitions are determined, are identified (see Fig. 6.3). In Morse decomposition, the obstacles can have arbitrary shapes, while Boustrophedon decomposition only handles polygon-shaped obstacles.

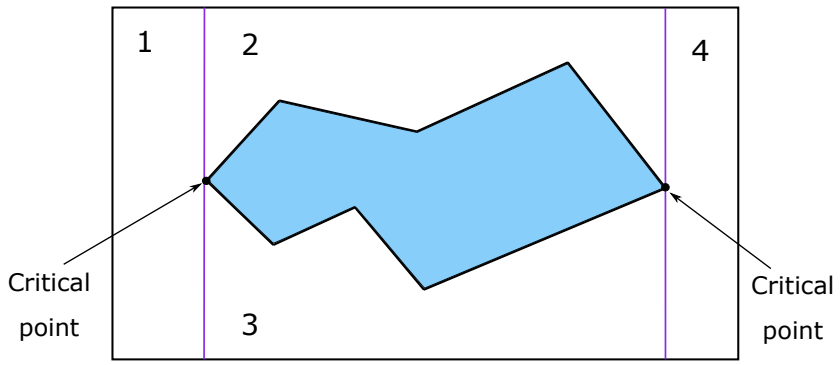


Figure 6.3: Boustrophedon decomposition around an obstacle [113].

6.1.4 Ergodicity Algorithms

In probability theory, *ergodicity* describes the property of a system in which the probability to reach every point in the state space is equal. Ergodicity algorithms in the

context of MSNs usually employ all-to-all communication so that each agent takes the motion of every other agent into account for its motion planning. In this way, the spatial distribution of trajectories of the agents is compared with a desired probability distribution. The difference between the desired and actual distribution is then minimized using optimization methods, which give the optimal trajectory for each agent [127, 64].

6.2 Problem Description

In this problem set, we consider agents, which can only receive the relative position of its neighbors and exchange information only with other agents within their limited communication bandwidth. Moreover, each agent is able to localize itself on the map. The measurement noise and sensor uncertainties are also neglected in this chapter.

Over the years, various sensor-based approaches have been proposed to cover areas with multiple agents. Most of them are based on information exchange to plan an optimal coverage motion without overlapping of the sensing regions. In *grid*-based methods, which divide the exploration area into very fine cells such as in [52, 51, 147], the coverage task is accomplished completely when the agents have sensed or identified each cell in the exploration area.

In real-life applications, area boundaries are usually given for the search task and obstacle locations are not known. The agents should autonomously identify each *cell* in the virtual map. However, some cells in the area are inaccessible because of obstacles or some other restrictions in the domain (Fig. 6.4). Hence, they cannot be sensed by the agents and remain *unidentified*.

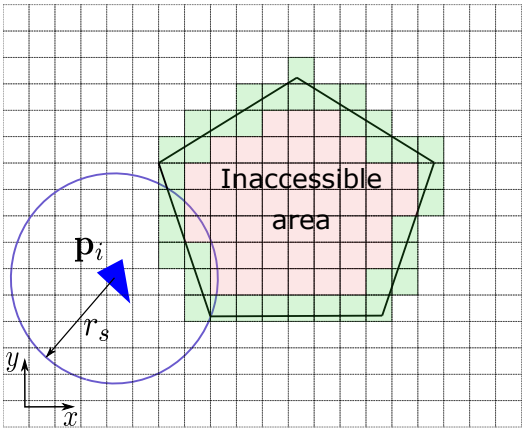


Figure 6.4: Illustration of the identification issue in an exploration task. Green-colored cells represent identifiable areas of the restricted region.

6.3 Cooperative Exploration Schema

In order to address the described issue in Section 6.2, we propose an exploration method which consists of four components: Motion planning, map generation using local communication, circumnavigation and recognition of inaccessible areas.

6.3.1 Motion Planning of Mobile Agents

The system considered in this study has the double-integrator dynamics given by

$$\begin{aligned} \dot{\mathbf{p}}_i &= \mathbf{v}_i, \\ \dot{\mathbf{v}}_i &= \mathbf{u}_i^{\alpha} + \mathbf{u}_i^{\beta} + \mathbf{u}_i^{\gamma}, \end{aligned} \tag{6.1}$$

where $\mathbf{u}_i^{\alpha}$ is the control input for the inter-agent *collision avoidance*, $\mathbf{u}_i^{\beta}$ is for the *obstacle avoidance* and $\mathbf{u}_i^{\gamma}$ is for the *tracking* of a target point. Each agent individually defines its target point. Moreover, each agent can receive the position of other agents within the communication range r_c and also the relative position of the closest point on an obstacle within its sensor range r_s. In this way, agents can identify the set of neighboring agents

$$\mathcal{N}_i^{\alpha} = \{j \in \mathcal{V} : \|\mathbf{p}_j - \mathbf{p}_i\| < r_c\}, \tag{6.2}$$

and the set of detected obstacle points

$$\mathcal{N}_i^{\beta} = \{\hat{\mathbf{p}}_{i,k} \Big| \ \|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\| < r_s\}, \tag{6.3}$$

at a given time. The control inputs $\mathbf{u}_i^{\alpha}$ and $\mathbf{u}_i^{\beta}$ are defined identically to (4.8) and (4.29), respectively.

In contrast to the previous chapters, the (α, α)-interaction is not the combination of attractive-repulsive forces but rather of purely repulsive forces, since the lattice-type cohesive behavior is not necessary for the area coverage task. The repulsive potential forces for the (α, α)-interaction and for the obstacle avoidance are defined by using the bump function (4.6) as follows:

$$\varphi_{\alpha}(z) = \rho_{h_{\alpha}}(z/d_{\alpha})(\sigma_1(z - d_{\alpha}) - 1), \tag{6.4}$$

$$\varphi_{\beta}(z) = \rho_{h_{\beta}}(z/d_{\beta})(\sigma_1(z - d_{\beta}) - 1), \tag{6.5}$$

where $d_{\beta} < r_{\beta}$ with $d_{\beta} = \|d_s\|_{\sigma}$, $r_{\beta} = \|r_s\|_{\sigma}$ and $d_{\alpha} = \|d\|_{\sigma}$. Finally, the control term for the *navigation* is defined as

$$\mathbf{u}_i^{\gamma} = -c_1^{\gamma}(\mathbf{p}_i - \mathbf{p}_i^t(t_k)) - c_2^{\gamma}\mathbf{v}_i, \tag{6.6}$$

where $\mathbf{p}_i^t \in \mathbb{R}^m$ represents the individual *target* position of agent i at a time instant $t_k > 0$ and corresponds to the γ-agent. Each agent has a different time-varying target

position, which is determined by using the collective map explained in the next section. Hereby, c_1^γ and c_2^γ are positive constant parameters.

6.3.2 Collective Map Creation

In order to maximize the identified domains, we apply the anti-flocking algorithm presented in study [51]. First, similarly to [147], the exploration area is considered as a grid consisting of equal-sized cells. The set of all cell center points, which are located in the row $\mu \in \{1, \dots, q\}$ and column $\nu \in \{1, \dots, n\}$ of the grid, are referred to as $\mathcal{P}_{\mu,\nu}$.

Each agent generates its own information map $\mathbf{M}_i \in \mathbb{R}^{q \times n}$, which is implemented as a matrix. The element $m_i(\mathbf{x}_{\mu,\nu}, t_k)$ in the μ-th row and ν-th column of $\boldsymbol{M}_i$ represents the (μ, ν)-th cell of the discretized EA at the given time t_k. The set of all points belonging to an obstacle within the EA is defined as $\mathcal{O}$. Thus, $\hat{\mathbf{p}}_{i,k} \in \mathcal{O}$ holds. A cell is considered *identified* if the coordinates of its center point $\mathbf{x}_{\mu,\nu}$ is within the sensing range r_s of an α-agent. Furthermore, a cell is considered *occupied* if there is an obstacle point detected in it. In order to store this information, $m_i(\mathbf{x}_{\mu,\nu}, t_k)$ is defined as follows

$$m_i(\mathbf{x}_{\mu,\nu}, t_k) = \begin{cases} 1, & \text{if } \|\mathbf{x}_{\mu,\nu} - \mathbf{p}_i(t_n)\| \leq r_s \wedge (\mathcal{O} \cap \mathcal{P}_{\mu,\nu} = \emptyset) \\ -1, & \text{if } \|\mathbf{x}_{\mu,\nu} - \mathbf{p}_i(t_n)\| \leq r_s \wedge (\mathcal{O} \cap \mathcal{P}_{\mu,\nu} \neq \emptyset) \\ 0, & \text{otherwise} \end{cases} \tag{6.7}$$

Remark 6.1. $m_i(\mathbf{x}_{\mu,\nu}, t_k) \in \{0, 1, -1\}$ is also called the sensing information. $m_i(\mathbf{x}_{\mu,\nu}, t_k) = 0$ denotes an *unidentified* cell. $m_i(\mathbf{x}_{\mu,\nu}, t_k) = 1$ represents an *identified* cell and $m_i(\mathbf{x}_{\mu,\nu}, t_k) = -1$ denotes an *occupied* cell. △

In the anti-flocking concept, each agent calculates its *target* position $\mathbf{p}_i^t$ by evaluating the benefit function given by

$$\xi_i = (1 - |m_i(\mathbf{x}_{\mu,\nu}, t_k)|)(\rho_\gamma + (1 - \rho_\gamma)\lambda_i(\mathbf{x}_{\mu,\nu})), \tag{6.8}$$

where ρ_γ is a constant and the function λ_i is defined as

$$\lambda_i(\mathbf{x}_{\mu,\nu}) = exp(-\kappa_1\|\mathbf{p}_i - \mathbf{x}_{\mu,\nu}\| - \kappa_2\|\mathbf{p}_i^t - \mathbf{x}_{\mu,\nu}\|) \tag{6.9}$$

with the positive constants κ_1 and κ_2. The benefit function serves as a metric for the individual benefit of an agent to explore different cells of the EA. As can be seen in (6.8) and (6.9), the benefit function includes the distance between the center of the cell and the current position of the agent as well as the current target $\mathbf{p}_i^t$. In this way, a benefit value is assigned to each cell using (6.8). The target position is then selected for the next time instant $t_k + 1$ as follows:

$$\mathbf{p}_i^t(t_k + 1) = \underset{\mathbf{x}_{\mu,\nu} \in \tilde{\mathcal{X}}_i}{\operatorname{argmax}} \, \xi_i(\mathbf{x}_{\mu,\nu}, t_k), \tag{6.10}$$

where $\tilde{\mathcal{X}}_i = \{\mathbf{x}_{\mu,\nu} \big| \mathbf{x}_{\mu,\nu} \in \mathcal{X},\ \|\mathbf{x}_{\mu,\nu} - \mathbf{p}_j\| \geq \|\mathbf{x}_{\mu,\nu} - \mathbf{p}_i\| > r_s,\ j \in \mathcal{N}_i^{\alpha}\}$. Hereby, $\mathcal{X}$ is the set of all cell center points.

For each agent, whose communication range is r_c, the information map is updated in the following way:

- In the beginning of the exploration ($t_k = 0$), all cells in the information map $\mathbf{M}_i$ are unidentified.
- Once an area is sensed, the cells $m_i(\mathbf{x}_{\mu,\nu})$ with $\|\mathbf{x}_{\mu,\nu} - \mathbf{p}_i\| \leq r_s$ are defined as *identified* (Fig. 6.1). In addition, if an obstacle is detected, the cell which includes the sensed obstacle point $\hat{\mathbf{p}}_i$ is saved as *occupied*.
- If an agent j is in the communication range of the agent i,

  $$\|\mathbf{p}_j - \mathbf{p}_i\| < r_c,$$

  the agents can exchange missing information and complete each others' maps at the time instant t_k,

  $$\mathbf{M}_i(t_k) = \mathbf{M}_j(t_k).$$

In addition, we have implemented the *recalculation* criteria described in [51] to reduce overlaps of sensing ranges and to minimize the traveling effort. In order to avoid frequent changes in the direction of the α-agents and to generate smooth trajectories, the γ-agents are not updated in each time step. The target position $\mathbf{p}_i^{\gamma}$ is updated based on the following four *recalculation* criteria. If none of the following criteria is met, $\mathbf{p}_i^t(t_k + 1) = \mathbf{p}_i^t(t_k)$ applies.

Criterion 1
The agent has reached its target position:
$\|\mathbf{p}_i^t(t_k) - \mathbf{p}_i(t_k)\|_{<} r_s$.

Criterion 2
The agents i and j communicate with each other and in the information map of the agent j, the cell in which the target position of the agent i is located is marked as *identified*:
$\|\mathbf{p}_j(t_k) - \mathbf{p}_i(t_k)\| < r_c \wedge |m_j(\mathbf{p}_i^t(t_k), t_k)| = 1$.

Criterion 3
The agents i and j are adjacent and the distance between the target positions belonging to i and j is less than the sensing range. The agent i updates its target position if it has a longer distance to $\mathbf{p}_i^t(t_k)$ than the agent j has to its target position $\mathbf{p}_j^t(t_k)$:
$\|\mathbf{p}_j(t_k) - \mathbf{p}_i(t_k)\|_{<} r_c \wedge \|\mathbf{p}_j^t(t_k) - \mathbf{p}_i^t(t_k)\| < r_s \wedge \|\mathbf{p}_i^t(t_k) - \mathbf{p}_i(t_k)\| > \|\mathbf{p}_j^t(t_n) - \mathbf{p}_j(t_k)\|$.
If the distance between $\mathbf{p}_i(t_k)$ and $\mathbf{p}_i^t(t_k)$ is less than that between $\mathbf{p}_j(t_k)$ and

$\mathbf{p}_j^t(t_k)$, the agent j recalculates its target position and $\mathbf{p}_i^t(t_k)$ remains the same.

Criterion 4

This criterion requires no recalculation, but an exchange of the target positions. Two agents i and j exchange their target positions with each other if the distance between the agent and the target position of the other agent is smaller than that between it and its own target position. Another prerequisite is that agents i and j are within communication range of each other:
$\mathbf{p}_i^t(t_k+1) = \mathbf{p}_j^t(t_k)$ and $\mathbf{p}_j^t(t_k+1) = \mathbf{p}_i^t(t_k)$, if
$\|\mathbf{p}_j(t_k) - \mathbf{p}_i(t_k)\| < r_c \wedge \|\mathbf{p}_i^t(t_k) - \mathbf{p}_i(t_k)\| > \|\mathbf{p}_j^t(t_k) - \mathbf{p}_i(t_k)\| \wedge \|\mathbf{p}_j^t(t_k) - \mathbf{p}_j(t_k)\| > \|\mathbf{p}_i^t(t_k) - \mathbf{p}_j(t_k)\|$.

6.4 Circumnavigation

The presented motion planning approach with map creation allows agents to completely identify an area without obstacles. However, an obstacle between an agent and its target position may yield a suboptimal performance in the area coverage due to the problem of local minima (see Section 4.4).

In addition, to overcome problems in area coverage described in Section 6.2, agents require *circumnavigation* skills, which means a complete navigation around an entire obstacle. Hence, the strategy presented in Section 5.1 is integrated into the motion planning algorithm.

In this way, once an obstacle is in the range of r_{tan} ($r_s > r_{tan} > d_s$), the circumnavigation[2] is activated. In the first step, the agent determines its rotation angle γ and calculates the *temporary target* position $\mathbf{p}_i^{tt}$ in an identical way to that in Section 5.1.1[3].

The navigation of the agent is performed using a similar equation to (6.6) as follows

$$\mathbf{u}_i^{\gamma} = -c_1^{\gamma}\left(c_n \cdot \frac{\mathbf{p}_i - \mathbf{p}_i^{tt}}{\|\mathbf{p}_i - \mathbf{p}_i^{tt}\|}\right) - c_2^{\gamma}\mathbf{v}_i,$$

where c_n is a positive constant weighting factor to have a controlled acceleration during the tangential navigation.

Once the agent simultaneously senses two different points on an obstacle, the corner avoidance strategy is activated and a new temporary target position $\mathbf{p}_i^{tt}$ is calculated (see Section 5.1.2).

If the distance between the agent and an obstacle holds $\|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\| > r_{tan}$ during tangential following, the agent switches to a new strategy utilizing the last sensed point $\hat{\mathbf{p}}_{i,e} = \hat{\mathbf{p}}_{i,k}(t_k - 1)$ on the obstacle with $\|\hat{\mathbf{p}}_{i,k}(t_k - 1) - \mathbf{p}_i\| \leq r_{tan}$ (see Section 5.1.3).

[2]In this chapter, circumnavigation is referred to as the navigation strategy proposed in Section 5.1.
[3]Hereby, $\mathbf{p}_i^t$ can be seen as $\mathbf{p}_d$ and $\mathbf{p}_i^{tt}$ represents $\mathbf{p}_i^v$ to apply the equations from Section 5.1

Once the first $\mathbf{p}_i^{tt}$ is reached ($\|\mathbf{p}_i^{tt} - \mathbf{p}_i\| < 0.7$), the agent sets a new temporary target position using (5.10). If a new point $\hat{\mathbf{p}}_{i,k}$ with $\|\hat{\mathbf{p}}_{i,k} - \mathbf{p}_i\| \leq r_{tan}$ is sensed, the navigation is performed identically to that described is Section 5.1.1. In the meantime, the following condition should be satisfied to leave the obstacle extremity:

$$|\alpha_i| \leq \omega, \tag{6.11}$$

where ω represents the tolerance angle range (cf. (5.11)). In the absence of a sensed point, condition (6.11) helps the agent stop setting new temporary target positions around an obstacle if the deviation between its velocity $\mathbf{v}_i$ and the vector from its current position $\mathbf{p}_i$ to the target position $\mathbf{p}_i^t$ is in a certain range.

6.5 Recognition of Restricted Areas and Obstacles

Circumnavigation allows an agent to reach its target position even if there is an obstacle between the agent and its target position. However, in the case of large two-dimensional obstacles/restricted areas, the benefit function of a cell inside the obstacle may have the maximum value and the target position is thus set into the restricted area. Through the circumnavigation, the agent would continuously circle around the obstacle in this case, since the condition for leaving the obstacle extremity (6.11) is never satisfied and usually none of the recalculation criteria from Section 6.3.2 is fulfilled. This problem occurs at the latest when all cells in the EA are marked as identified, except the cells occupied by the obstacles. Although the agents can identify the cells at the edge of obstacles as occupied, they cannot identify the cells behind the edges due to their inaccessibility. Thus, the inaccessible areas (see Fig. 6.4) permanently have the status of *unidentified* in the information map of the agents.

The algorithm we propose here addresses this problem in that the agents reconstruct the obstacle autonomously and in real time by solely utilizing the sensed obstacle points (β-agents). In this way, the agents can mark the inaccessible cells within the obstacles as *occupied* in their information map. This enables the agents to make the decision for finalizing the exploration task on their own.

As outlined in Section 2.1.3, the agents do not perceive obstacles as a whole object, but only artificially interact with the sensed point on an obstacle that has the least distance to them. Thus, the positions of the β-agents are the only information an agent has about an obstacle. In order to analyze the shape of an obstacle, each agent is capable of memorizing the sensed obstacle points $\hat{\mathbf{p}}_{i,k}(t_k)$ at any time when an obstacle is detected.

Obstacle recognition can then be easily included in the *area exploration* framework by implementing the following scheme:

1) During circumnavigation, the agent stores all cells containing a sensed obstacle point. These cells are separately marked as occupied in the information map of the agent.

2) The set of the stored cells is defined as $\mathcal{C}_i = \{\mathbf{c}_{i,k}(t_k), \mathbf{c}_{i,k}(t_k+1), \ldots, \mathbf{c}_{i,k}(t_k+n)\}$. Moreover, the set is considered as a graph $G(\mathcal{C}_i)$. After having 6 stored cells[4] in $\mathcal{C}_i$, the algorithm checks the *cyclicity* of the graph G to detect if the obstacle has a closed geometry. A cycle graph is a graph in which nodes are connected to each other in a circular structure and at least one edge has the same start and end node [20]. In this way, the algorithm examines for each stored cell $\mathcal{C}_i$ the following cyclicity conditions:
   - The previously registered cell should be adjacent to the currently detected cell.
   - The first registered cell should be adjacent to the currently detected cell.

   This allows agents to identify *closed* geometries and thus restricted domains in the exploration area, which are not accessible.
3) If the cyclicity conditions are fulfilled for all cells in the list $\mathcal{C}_i$, the cells inside the *closed* geometry are marked as occupied by using the proposed algorithm in Section 6.5.2.
4) Finally, the list of cells is automatically deleted for a new obstacle recognition task.

Remark 6.2. The list of cells is also deleted if the agent terminates the circumnavigation mode or if the agent changes its direction of rotation, which means that the β_i (see Section 5.1) changes its sign. △

6.5.1 Implementation of the Proximity Investigation

For the cyclicity conditions from the item 2), the proximity of each registered cell in $\mathcal{C}_i$ has to be investigated. Using the following algorithm steps, which are visualized in Fig. 6.5, we can examine if a specific cell is adjacent to another one in the grid.

- The algorithm iteratively goes through each cell in the list $\mathcal{C}_i$. In each iteration, one cell is closely observed and serves as a reference (x^*, y^*). Starting from this cell, we assign column and row indexes to the neighboring cells.
- In addition, by subtracting the column and row indices of the reference cell from the consecutive cells in proximity, we define the *difference vector* $\mathbf{d}_i = (x, y)$ for each cell.
- Then, the Euclidean norm of every single difference vector is determined as: $s_i = \|\mathbf{d}_i\|$.
- Using the Euclidean norm of difference vectors, the following conditions to determine the close proximity of the reference cell are defined:
  i. Euclidean norm of the difference vector $s_i = 1$ **or**
  ii. Euclidean norm of the difference vector $s_i = \sqrt{2}$.

[4]The stored cells denote the nodes of the graph G.

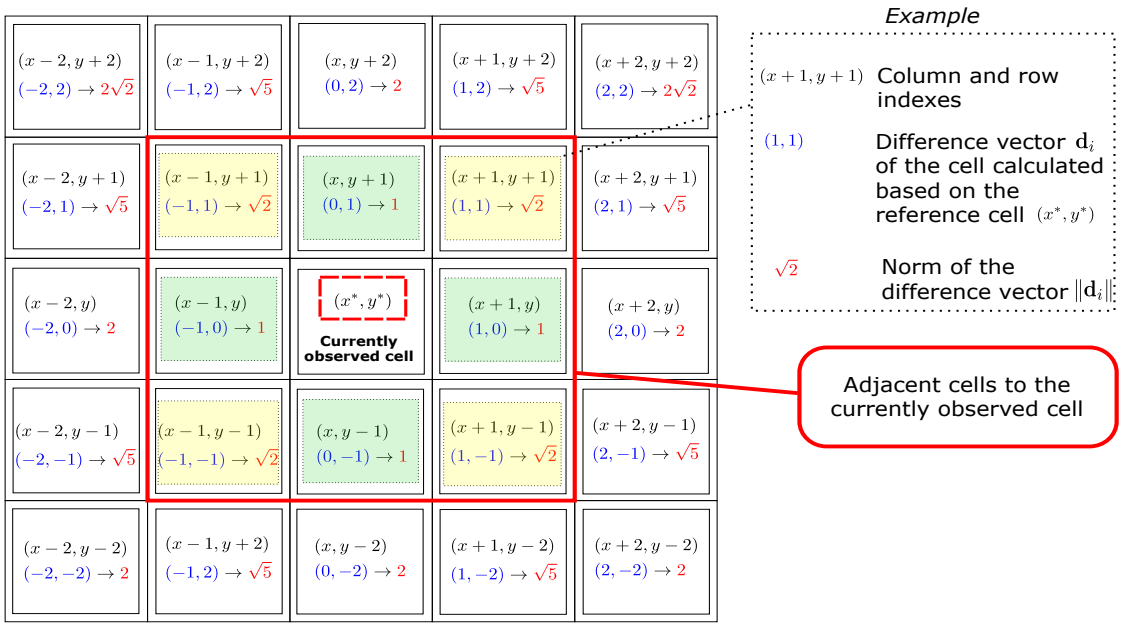


Figure 6.5: Visual representation of proximity investigation.

If a cell in the proximity of (x^*, y^*) fulfills one of the above conditions, it is adjacent to the currently observed cell in the iteration (see Fig. 6.5).

- If one of the above conditions is not fulfilled, all cells up to the currently detected one are deleted from the list $\mathcal{C}_i$.

6.5.2 Determination of the Cells within a Restricted Domain

Estimating restricted areas is a challenging task for the agents. For this purpose, we developed an iterative procedure to make a statement about the cells behind the *closed* obstacles in the information map. During circumnavigation, each agent creates a list of all sensed obstacle points $\mathcal{P}_i = \{\hat{\mathbf{p}}_{i,k}(t_k), \hat{\mathbf{p}}_{i,k}(t_k+1), \ldots, \hat{\mathbf{p}}_{i,k}(t_k+n)\}$. The algorithm puts the section, which is probably occupied by the obstacle, under the scope by isolating it on the information map. The isolation is based on the extreme coordinates of the obstacle:

$$\boldsymbol{p}_{x_{min}} = \underset{\hat{\mathbf{p}}_{i,z} \in \mathcal{P}_i}{\operatorname{argmin}} \left[(1\,,\,0) \cdot \hat{\mathbf{p}}_{i,z} \right], \tag{6.12a}$$

$$\boldsymbol{p}_{x_{max}} = \underset{\hat{\mathbf{p}}_{i,z} \in \mathcal{P}_i}{\operatorname{argmax}} \left[(1\,,\,0) \cdot \hat{\mathbf{p}}_{i,z} \right], \tag{6.12b}$$

$$\boldsymbol{p}_{y_{min}} = \underset{\hat{\mathbf{p}}_{i,z} \in \mathcal{P}_i}{\operatorname{argmin}} \left[(0\,,\,1) \cdot \hat{\mathbf{p}}_{i,z} \right], \tag{6.12c}$$

$$\boldsymbol{p}_{y_{max}} = \underset{\hat{\mathbf{p}}_{i,z} \in \mathcal{P}_i}{\operatorname{argmax}} \left[(0\,,\,1) \cdot \hat{\mathbf{p}}_{i,z} \right]. \tag{6.12d}$$

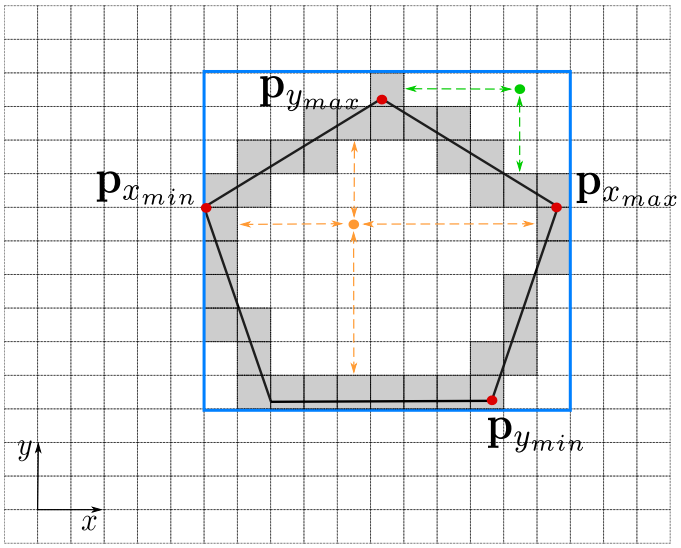


Figure 6.6: Estimation of the occupied cells.

As an example, we consider a polygon illustrated in Fig. 6.6. In this case, the corner points would be used as extreme points (see (6.12)) to isolate the section in EA. In Fig. 6.6, the isolated section is indicated with a blue-colored frame.

ν_{min} is the column of the information map, where $\mathbf{p}_{x_{min}}$ is located and ν_{max} is the column where $\mathbf{p}_{x_{max}}$ is located. Moreover, μ_{max} indicates the row of the cell of the information map, in which $\mathbf{p}_{y_{max}}$ is located, and μ_{min} is the row that includes $\mathbf{p}_{y_{min}}$.

In order to determine whether a cell (μ_o, ν_o) of the isolated section is inside a restricted domain, we check if it is surrounded by occupied cells in all directions: *above, below, left* and *right*. If this is the case, the cell is within the restricted domain and is also marked as occupied on the information map. An example for such a cell is the cell with an orange-colored point in Fig. 6.6. However, the green marked cell is outside the obstacle and cannot be marked as occupied, because although there are occupied cells below and on the left side of it, there are no occupied cells above and on the right side. The exact procedure is described as pseudo-code in Algorithm 6.1.

6.6 Simulation Studies

In this section, the effectiveness of the proposed approach is evaluated. For this purpose, different scenarios for covering an unknown area are simulated and analyzed.

A characteristic measure used to evaluate the approach is the *coverage* $\vartheta(t)$ in %. This is calculated comparing the ratio of the cells covered by all agents to the total number of cells in the EA. A cell is considered *covered* if it is either identified by one of the agents or marked as *occupied* by the method presented in Section 6.5. The time in seconds required to completely cover an area is defined as t_{100}. Thus, $\vartheta(t_{100}) = 100\%$.

Algorithm 6.1 : Identification of cells within a restricted domain

Data : Information map $\mathbf{M}_i$ of the agent i, indices of the rows μ_{min}, μ_{max} and columns ν_{min}, ν_{max} of the isolated domain

Result : Information map $\mathbf{M}_i$ of agent i

```
for μ_o = μ_min : μ_max do
    for ν_o = ν_min : ν_max do
        ▷ (μ_o, ν_o) is the currently examined cell.
        m_i(x_{μ_o,ν_o}) = M_i(μ_o, ν_o);
        above=false, below=false, left=false, right=false;
        for μ = μ_min : μ_o − 1 do
            ▷ All cells in the same column above (μ_o, ν_o).
            if m_i(x_{μ,ν_o}) = −1 then
                above=true;
        for μ = μ_o + 1 : μ_max do
            ▷ All cells in the same column below (μ_o, ν_o).
            if m_i(x_{μ,ν_o}) = −1 then
                below=true;
        for ν = ν_min : ν_o − 1 do
            ▷ All cells in the same row left of (μ_o, ν_o).
            if m_i(x_{μ_o,ν}) = −1 then
                left=true;
        for ν = ν_o + 1 : ν_max do
            ▷ All cells in the same row right of (μ_o, ν_o).
            if m_i(x_{μ_0,ν}) = −1 then
                right=true;
        if above = true and below = true and left = true and right = true then
            m_i(x_{μ_o,ν_o}) = −1;
        M_i(μ_o, ν_o) = m_i(x_{μ_o,ν_o});
    return M_i
```

In the figures of this section, unidentified cells are highlighted in blue, identified and unoccupied cells in yellow. Moreover, cells identified by agents as occupied are illustrated in gray. Obstacles are indicated by black lines. The positions of the agents are visualized with white-colored triangles and the green-colored crosses denote the positions of agents' target points.

The values of the parameters for the anti-flocking, the weighting factors of the applied control terms and the parameters for the circumnavigation are given in Table 6.1. Lengths, velocities and accelerations are considered as dimensionless variables. All simulations were performed with these parameter values and in an EA with identical dimensions. The coordinate system used has its origin in the center of the EA.

In the simulation, we consider a system of $N = 3$ agents in the $m = 2$ dimensional plane. The agents are randomly placed in the EA and their initial velocities $\mathbf{v}_i(0) \in \mathbb{R}^2$ are $(0, 0)^T$. The EA is defined in a domain $[-60, 60] \times [-60, 60]$ with a cell size of $[4 \times 4]$. In the simulation, the input $|\mathbf{u}_i^\gamma|$ is bounded by $|\mathbf{u}_{\gamma,max}| = 100$ to avoid too high accelerations. The priorities of the control objectives are defined: *collision avoidance*, *obstacle avoidance* and *tracking*, $c_1^\beta > c_1^\alpha > c_1^\gamma$.

Table 6.1: Parameter setting.

Anti-flocking	r_c	r_s	r_{tan}	d_s
	40	14	10	8
	h_α	h_β	ε	d
	0.2	0.9	0.08	12
	κ_1	κ_2	ρ_γ	
	0.04	0.01	0.2	
Weighting of the control terms	c_1^α	c_1^β	c_1^γ	
	60	60	30	
	c_2^α	c_2^β	c_2^γ	
	$2\sqrt{c_1^\alpha}$	$2\sqrt{c_1^\beta}$	$2\sqrt{c_1^\gamma}$	
Circumnavigation	δ	ω	c_n	
	$20°$	$10°$	4	

6.6.1 Area Coverage without Restricted Domains

First, we evaluate the influence of the exchange of information maps among the agents on the area coverage. In order to analyze this dependency in an isolated way, we have a simulation setting without obstacles with $N = 3$ agents. In the first scenario, we observe the case without communication between the agents. Then, we perform

simulations with agents which can communicate and have a communication range of $r_c = 30$, $r_c = 60$ and $r_c = 90$. Fig. 6.7 shows the evolution of the coverage $\vartheta(t)$ of the four simulations performed.

As shown in Fig. 6.7, coverage through multiple agents without communication increases identically to that with a local communication schema in the beginning. This can be explained with the fact that the initial positions of the agents and the corresponding γ-agents at $t_k = 0$ are identical in all simulations. If the distance between the agents are sufficiently large, as in this case, and their initial γ-agents are widely distributed, there is a very low probability of the sensing ranges overlapping at the beginning.

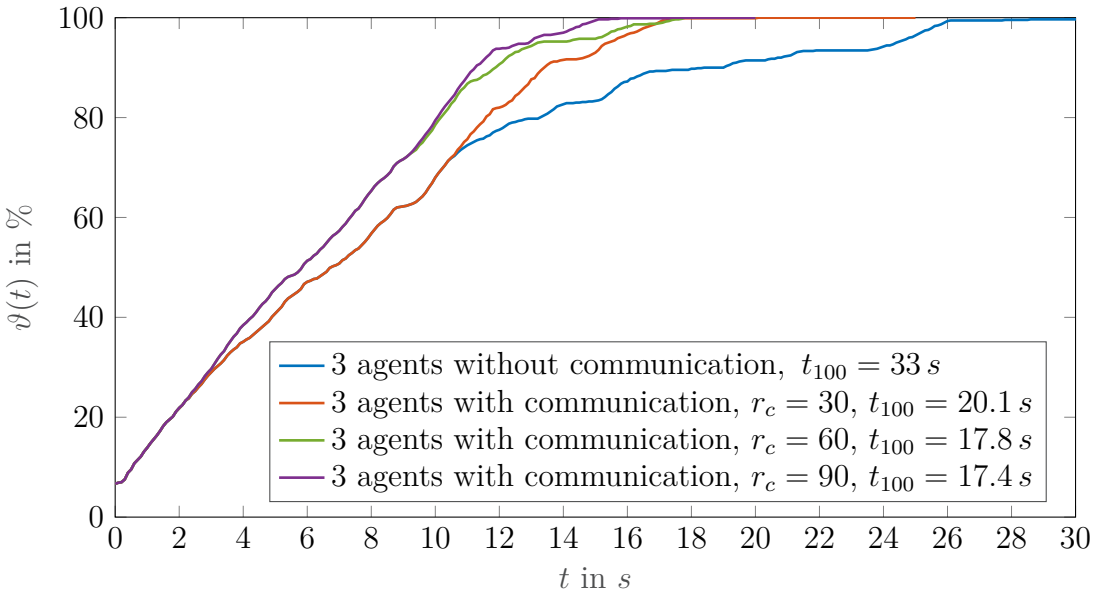
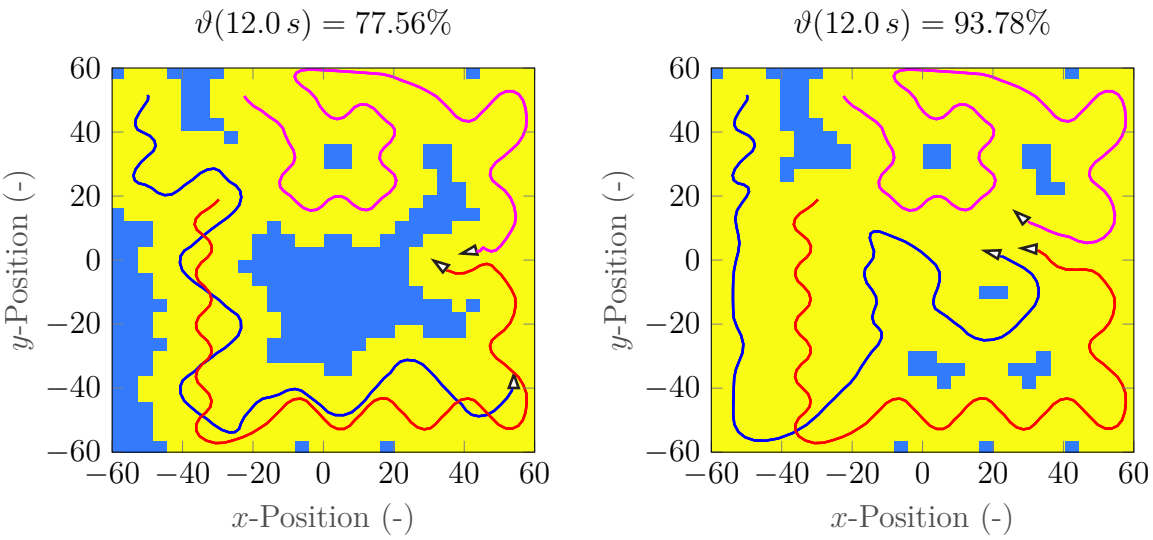


Figure 6.7: Influence of communication on the area coverage.

After a certain amount of time (at $t \approx 3s$), however, multiple identification of some cells increasingly arises because each agent without communication strives to cover the entire area by itself and has no information about the maps of the other agents. Fig. 6.8 depicts the snapshots of the simulations *with* and *without* communication at $t = 12\,s$. While the area in Fig. 6.8(b) is mostly covered, the percentage of identified cells in Fig. 6.8(a) is only 77.56%. The trajectories of the agents in the left figure illustrate the problem of an anti-flocking algorithm without information exchange. Two of the three agents are moving on nearly identical paths instead of exploring the area in a distributed way. Apparently, without communication, the coverage performance of the agents is limited.

6.6.2 Autonomous Recognition of Restricted Domains

Recognition of restricted areas is a significant part of the concept proposed in this chapter. Fig. 6.9 depicts the positions and trajectories of agents during an exploration mission without using the obstacle recognition scheme at time instant $t = 68.6\,s$. The



(a) Coverege at $t = 12\,s$ without communication.

(b) Coverage at $t = 12\,s$ with communication, $r_c = 90$.

Figure 6.8: Coverage of the EA by 3 agents with and without exchanging the information maps.

EA and the area coverage by all agents are also illustrated in this figure. The EA contains two polygon-shaped and one circular obstacles that are not known to the agents beforehand. Hereby, it is observed that the agents are in a continuous circumnavigation mode and try to cover the inner areas of obstacles.

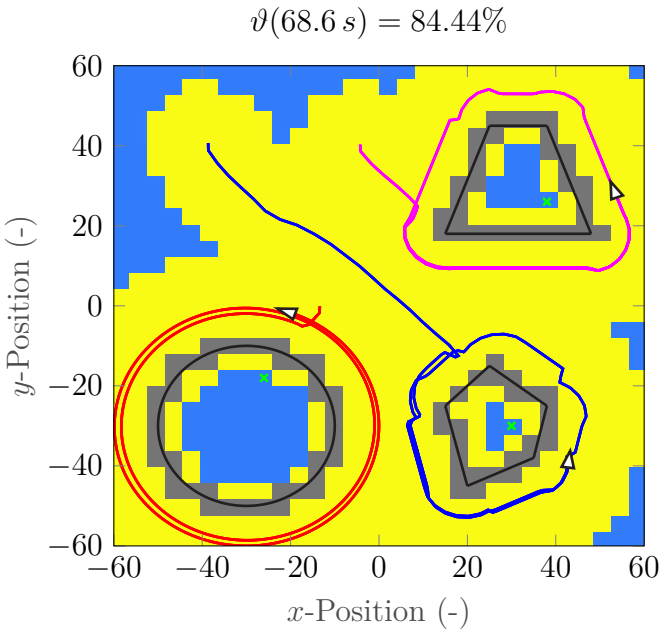


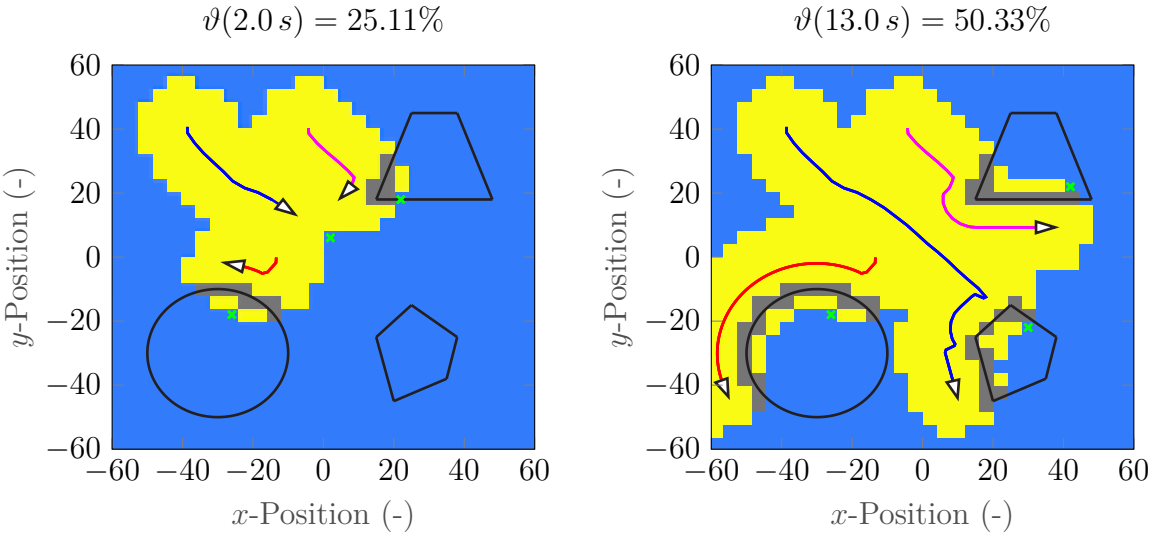
Figure 6.9: Exploration task without obstacle recognition with 3 agents at $t = 68.6\,s$.

In order to show the effectiveness of our approach, we perform another simulation with identical initial conditions as in the previous one. Fig. 6.10 shows the positions and trajectories of the agents using obstacle recognition and tangential navigation schemes for important time instants. With this, the exchange of information map enables the group to make optimal decisions about the next target position for efficient exploration. Using the proposed scheme, agents start to perform circumnavigation at $t = 13\,s$. Although the agents complete the circumnavigation and also the recognition process at $60\,s$, the one agent identifying the circular obstacle apparently does not satisfy condition (6.11). Hence, it continues with the circumnavigation. However, finally, all agents can recognize obstacles and identify inaccessible areas on their own.

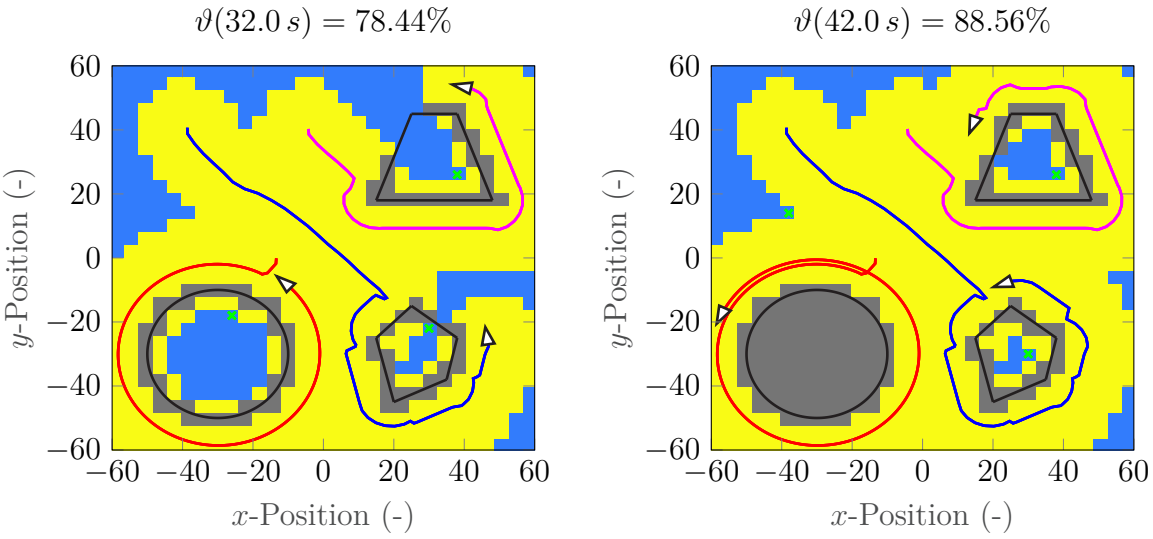
6.7 Chapter Highlights

In this chapter, a novel, sensor-based exploration framework is introduced, which allows agents to identify each cell in the exploration area by recognizing inaccessible domains autonomously. With the proposed scheme, multiple agents can efficiently explore an unknown area with predefined boundaries as a group through local information exchange. However, the main contribution is that the agents are capable of identifying restricted areas on their own without having prior knowledge. The combination of the proposed obstacle recognition with the circumnavigation approach makes it possible for the agents to identify inaccessible areas by evaluating only sensor data.

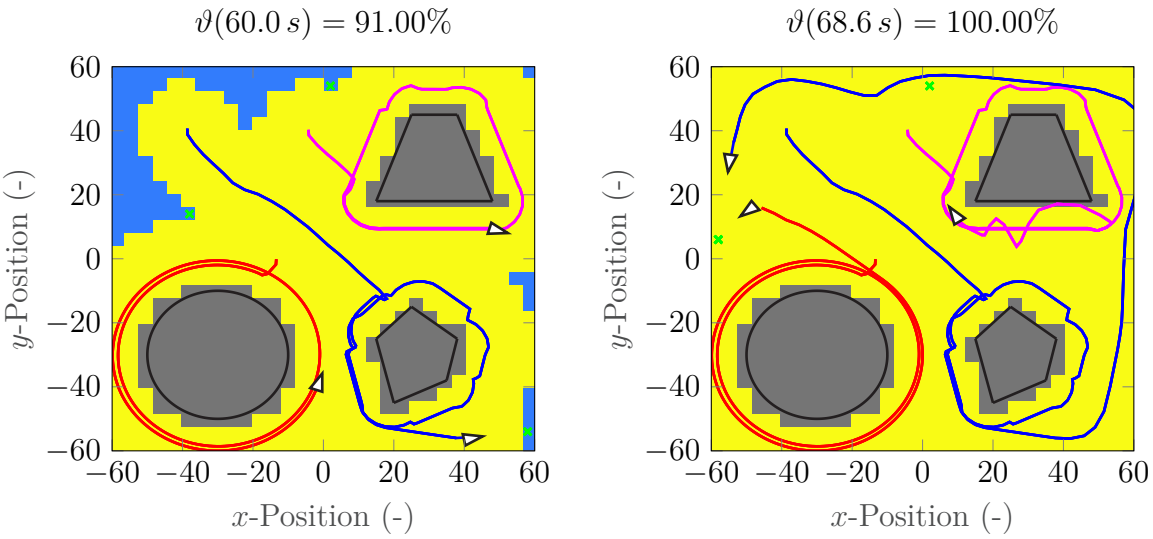
The supervised student thesis [161] has contributed to the development of this chapter's results.



(a) $t = 2\,s$ (b) $t = 13\,s$



(c) $t = 32\ s$ (d) $t = 42\,s$



(e) $t = 60\,s$ (f) $t = 68.6\,s$

Figure 6.10: Consecutive snapshots of an exploration task with 3 agents with obstacle recognition.

7 Optimal Control of Swarming Agents

Parts of the following chapter have been published in [154].

In this chapter, we are concerned with the control of a multi-agent system including an external leader-agent, which guides a group of autonomous agents with limited information to desired objectives. The problem of herding sheep can simply exemplify the scenario investigated in this chapter. There have been several attempts to influence the behavior of biological groups, such as fishes, cockroaches and birds, through external stimuli generated by robots [11, 12, 57, 110]. Furthermore, similar problem settings have been investigated to control swarming agents with self-designed dynamics through single or multiple intelligent agents, the so-called *external leaders* [58, 4, 26].

The basic idea of our leader-agent concept is that only the leader-agent knows the predetermined group objective and the environment. The leader tries to steer the entire swarm toward the objectives by using *optimal* interventions. For this purpose, we apply an optimal control strategy. Possible objectives for the group include moving to a target position, tracking a given trajectory, and avoiding collisions with obstacles in the two-dimensional space.

The rest of this chapter is structured as follows: Section 7.1 presents basic preliminaries on optimization and optimal control required to understand the following sections. Section 7.2 gives an overview about solvers for nonlinear programming problems. In Section 7.3, the global path planning problem we work on in this chapter is briefly formulated. Section 7.4 describes the system dynamics in detail and the formulation of the optimal control problem (OCP) through the guidance of a leader-agent. This section is based on [154], a contribution by the author of this thesis. The starting point of this section is [26], which employs an MPC scheme to drive a flock with deterministic dynamics to a desired position through leadership. In Section 7.5, simulation results are illustrated.

7.1 Preliminaries: Optimal Control

In this section, we briefly introduce the necessary basics about optimal control.

In general, the term *optimization* denotes the search for an optimal solution regarding a specific objective under given circumstances. The objective of an optimization may be

the effort optimization, e.g., energy-optimal routes for electric vehicles, the trajectory that takes the least time to accomplish a task, i.e. a time-optimal trajectory for a race-car, or determination of the shortest feasible path in path planning. There are static and dynamic optimization problems.

In static optimization problems, the goal is to minimize or maximize an objective function by considering given constraints. However, the goal in dynamic optimization problems is finding variables as functions of time that minimize a cost functional under consideration of constraints.

Optimal control is thus a function that minimizes or maximizes a given cost functional subject to differential equations under several state and input constraints. The mathematical formulation of such problems leads to the optimization problem. A general mathematical formulation of dynamic optimization problems is given by

$$\min_{\mathbf{x},\mathbf{u}} \; J = V(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t))\,\mathrm{d}t \tag{7.1}$$

$$\text{subject to } \dot{\mathbf{x}}(t) = f(\mathbf{u}(t), \mathbf{x}(t)), \quad \mathbf{x}(t_0) = \mathbf{x}_0 \tag{7.2}$$

$$\mathbf{h}(\mathbf{x}(t), \mathbf{u}(t)) \leq \mathbf{0}, \tag{7.3}$$

$$\mathbf{g}(\mathbf{x}(t_f)) = \mathbf{0}, \quad \forall t \in [t_0, t_f] \tag{7.4}$$

where the variable $\mathbf{x} \in \mathbb{R}^n$ is the state of the system and $\mathbf{x}_0$ represents a given initial state. The term J in (7.1) represents the cost functional. It consists of the so-called Mayer term or the final state evaluation $V(\mathbf{x}(t_f))$ and the integral part, which is also known as the Lagrangian form ($\int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t))\,\mathrm{d}t$). The $\mathbf{u} \in \mathbb{R}^m$ is the control input of the nonlinear system (7.2). In addition, an optimization problem may be subject to a set of constraints such as (7.3) and (7.4). The $\mathbf{h}$ represents the inequality constraints, and $\mathbf{g}$ denotes the equality constraints. The end conditions $\mathbf{g}$ describe a desired state $\mathbf{x}_f$ to be reached at a time t_f, e.g., ($\mathbf{g}(\mathbf{x}(t_f)) = \mathbf{x}(t_f) - \mathbf{x}_f$). The end time t_f may be predetermined or be a part of the solution of the optimization problem. Eq. (7.3) describes inequality constraints, such as the limitation of a control variable or a state. Boundaries of the control variable $\mathbf{u}$ or the state $\mathbf{x}$ of the system can be expressed by

$$\mathbf{u}(t) \in U \subseteq \mathbb{R}^m\,, \quad \forall t \in [t_0, t_f]\,, \tag{7.5}$$

$$\mathbf{x}(t) \in X \subseteq \mathbb{R}^n\,, \quad \forall t \in [t_0, t_f]\,. \tag{7.6}$$

In practice, such restrictions may be voltage limitation of a motor, maximum force of an actuator or restrictions in workspace. For the solution of dynamic optimization problems, numerical methods are usually required. Since solving the *Hamilton-Jacobi-Bellman equation*[1] is limited to a small number of state variables, *indirect* and *direct* numerical methods are mostly considered for solving optimal control problems.

The indirect methods rely on solving the problem indirectly by formulating the optimal control problem as a boundary value problem [120]. Then, an optimal solution

[1] The Bellman equation can be solved by backwards induction, either analytically or numerically.

is found that satisfies boundary values. Further information about indirect methods can be found in [18, 107]. In the direct approaches, the infinite-dimensional optimal control problem is approximated with a finite-dimensional problem. By discretizing the problem in time and defining the states and control variables at those time points, the control problem can be formulated as a *nonlinear programming* (NLP) problem. The NLP problem can be efficiently solved with numerical methods of nonlinear optimization. Direct approaches for the solution of constrained OCPs are more widespread than indirect methods due to their simple applicability and robustness. The best known direct approaches are: Direct single-shooting, direct multiple-shooting, direct collocation and pseudospectral collocation [120]. For a detailed introduction of the nonlinear optimization methods, the reader is referred to the literature [18, 121, 10].

In this study, we implement the problem as direct multiple-shooting. In the direct multiple-shooting method, the control trajectory is parameterized (e.g., piecewise constants) and common numerical adaptive integration methods (e.g., Runge-Kutta) are used for the time discretization of the continuous time dynamics. After converting the problem into an NLP problem, an appropriate solver is required to obtain the optimal control trajectory.

The constrained NLP problem is defined as

$$\min_{\mathbf{x}} \; J(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n \tag{7.7}$$

s. t. :

$$\mathbf{h}(\mathbf{x}) \leq \mathbf{0}, \tag{7.8}$$

$$\mathbf{g}(\mathbf{x}) = \mathbf{0}. \tag{7.9}$$

Thereby, J represents the cost functional, which is subject to the inequality and the equality constraints, $\mathbf{h}$ and $\mathbf{g}$, respectively. In the present work, the *interior-point* (IP) solver is used. For the implementation of a nonlinear IP approach, the reader is referred to [142].

7.2 NLP Solver

There is a wide range of software packages for solving dynamic optimization problems using the direct methods (e.g., ACADO [62], Pyomo [60], GEKKO [16], CasADi [8], GAMS [19], DIDO [124], AMPL [47], PSOPT [17], GPOPS-II [111], ICLOCS2 [95]). Fundamentally, these differ with respect to the supported problem classes, the interface (Python, Matlab, Fortran, C++), the calculation method for the required derivatives, the transcription method for the transformation of the OCP into an NLP and the available interfaces for the NLP solvers.

In this study, we need a free software framework with an interface to MATLAB, which calculates the required derivatives using algorithmic differentiation (AD) and provides

interfaces to efficient NLP solvers like IPOPT (Interior Point OPTimizer). These requirements were fulfilled by the open-source software CasADi.

CasADi is a symbolic toolbox for algorithmic differentiation (AD) and numeric optimization with a syntax taken from computer algebra systems (CAS) [9, 7]. It is primarily a gradient-based numerical optimization tool mainly focused on finding optimal control inputs. This symbolic framework allows the user to define symbolic expressions for Eqs. (7.1)-(7.4), and evaluate the derivatives using algorithmic differentiation. In order to solve the optimization problem, the formulated OCP (7.1)-(7.4) must be converted into an NLP. CasADi does not take over this step, but offers some functionalities that considerably simplify the transcription.

7.3 Problem Formulation

In this chapter, we once again investigate a flocking system assuming, as in the previous chapters, that each agent has a mechanism to communicate with its neighboring agents and receives their relative positions. Most of the leader-follower approaches allow all agents to track a leader, a target position, and to detect the obstacles within their sensing ranges. In contrast to the schemes presented in previous chapters, we assume by now that the agents cannot perceive obstacles and interact with them. In addition, the agents do not know the group objective. **The aim of this study** is to develop a control concept for steering a flocking multi-agent system with limited perceptive abilities and limited information about the environment to an unknown target position by ensuring collision avoidance with obstacles.

The interaction between the agents is based on the potential function presented in Section 2.2. The state of each agent is represented by $(\mathbf{p}_i, \mathbf{v}_i) \in \mathbb{R}^m \times \mathbb{R}^m$, which denote the position and velocity of the i-th agent in m-dimensional space, respectively. The double-integrator dynamics of a group of N agents are identical to those in [101, Algorithm 1] and given by

$$\begin{aligned}\dot{\mathbf{p}}_i &= \mathbf{v}_i \\ \dot{\mathbf{v}}_i &= \sum_{j\in\mathcal{N}_i} \varphi_\alpha(\|\mathbf{p}_j - \mathbf{p}_i\|_\sigma)\mathbf{n}_{ij} + \sum_{j\in\mathcal{N}_i} a_{ij}(\mathbf{p})(\mathbf{v}_j - \mathbf{v}_i)\end{aligned}$$

for $i = 1, \ldots, N$. The neighborhood of agent i is defined as

$$\mathcal{N}_i = \{j \in \mathcal{V} : \|\mathbf{p}_j - \mathbf{p}_i\| < r_c\},$$

where $\mathcal{V}$ is a set of all agents and r_c denotes the interaction range. Furthermore, each agent tries to keep the predefined distance d to its neighbors and the mathematical boundary condition is given as follows:

$$\|\mathbf{p}_j - \mathbf{p}_i\| = d, \quad \forall j \in \mathcal{N}_i(\mathbf{p})$$

According to observations in nature, the overall goal is not always known to the entire group. In this case, the group forms a hierarchical structure to achieve the objective like a wolf pack with an alpha wolf (leader), which rules the group [86]. With this motivation, we extended the flocking algorithm proposed in [100] with an external leader-agent that can interact with all other agents in the group while it is in the interaction range of them. The leader agent is not followed by other agents. However, it is perceived as a *special* agent. In this respect, all agents receive stronger artificial interaction forces from the leader and thus, the leader can influence the whole group.

7.4 Optimal Control through Guidance of a Leader

In the present setting, the leader-agent is the only one that priorly knows the group objective and the environment. The uninformed agents[2] have weak sensors with limited sensor bandwidths. As a result, they can only communicate with the agents in their sensing range and do not detect any obstacle or know the coordinates of the predefined target point.

7.4.1 Extended Model with a Leader-Agent

In the following, we extend the flocking model considered in [100, Algorithm 1] appropriately for our setting by considering the leadership concept proposed in [146]. The leader agent is identified with the index 1 and the index b refers to the uninformed agents. The system has N agents including the leader. The extended model with the action of an externally controlled leader-agent is described by the following system of differential equations

$$\begin{aligned}\dot{\mathbf{p}}_1 &= \mathbf{v}_1\\ \dot{\mathbf{v}}_1 &= \underbrace{\sum_{b\in\mathcal{N}_1}\varphi_\alpha(\|\mathbf{p}_b-\mathbf{p}_1\|_\sigma)\mathbf{n}_{1b}}_{\mathbf{G}_1}+\underbrace{\sum_{b\in\mathcal{N}_1}a_{1b}(\mathbf{p})(\mathbf{v}_b-\mathbf{v}_1)}_{\mathbf{C}_1}+\mathbf{u}\end{aligned} \tag{7.10}$$

$$\begin{aligned}\dot{\mathbf{p}}_b &= \mathbf{v}_b\\ \dot{\mathbf{v}}_b &= \underbrace{\sum_{j\in\mathcal{N}_b}\varphi_\alpha(\|\mathbf{p}_j-\mathbf{p}_b\|_\sigma)\mathbf{n}_{bj}}_{\mathbf{G}_b}+\underbrace{\sum_{j\in\mathcal{N}_b}a_{bj}(\mathbf{p})(\mathbf{v}_j-\mathbf{v}_b)}_{\mathbf{C}_b}+\mathbf{L}_b\end{aligned} \tag{7.11}$$
$$\text{for}\quad b=2,\dots,N$$

where $\mathbf{u}$ is the control input that acts only on the velocity dynamics of the leader-agent. The interaction between agents (acceleration term) includes two significant terms (cf.

[2]From now on, we will refer to all agents with no prior information about the environment and the group objective as *uninformed* agents.

Section 4.2.1). The terms $\mathbf{G}_1$ and $\mathbf{G}_b$ represent the gradient-based term of the leader and uninformed agents, respectively. The term $\mathbf{G}_1$ allows the controlled leader agent to interact with the uninformed agents by applying repulsive-attractive forces. Moreover, $\mathbf{G}_b$ quantifies the force to maintain the α-lattice form among the followers. The second terms $(\mathbf{C}_1, \mathbf{C}_b)$ in both leader and follower dynamics represent the consensus term (cf. Section 4.2.1).

A further interaction between the uninformed agents and the leader is described by the term $\mathbf{L}_b$ in (7.11). Similarly to the refined flocking model in the study [26], the leader action is defined using the gradient-based term, which controls the attraction and repulsion between the uninformed agents and the leader. The leader action term is given by

$$\mathbf{L}_b = \lambda \cdot \varphi_\alpha(\|\mathbf{p}_b - \mathbf{p}_1\|_\sigma)\mathbf{n}_{b1}, \tag{7.12}$$

where the strength of the interaction between the b-th agent and the leader can be adjusted by the positive constant $\lambda > 0$. In this way, the uninformed agents can be influenced through the actions of the leader agent, which will be optimally controlled.

7.4.2 Controllability of the Extended Model

Within the scope of the problem considered in this chapter, the question arises whether the given system can actually be influenced by the control variable $\mathbf{u}(t)$ in such a way that the desired objectives are achieved. In order to answer this, the controllability analysis of the system introduced in Section 7.4.1 is crucial, since this is an elementary property for the realization and functionality of the concept. For a detailed introduction to the controllability of a system, readers are referred to the literature [2, 35, 97].

The extended model is a nonlinear system. A nonlinear system is defined as

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \tag{7.13}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ and $\mathbf{u}(t) \in \mathbb{R}^m$ are the state vector and control input of the system, respectively. $\mathbf{f}$ represents a vector function. Alternatively, the system (7.13) can also be written as

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{B}\mathbf{u}(t), \tag{7.14}$$

where the $\mathbf{B} \in \mathbb{R}^{n \times m}$ denotes the input matrix. The controllability is a global system property for linear systems. However, the statement does not always hold for nonlinear systems. There are nonlinear systems that are only controllable on a subset of their reachable sets. This system property is called *local controllability* (see [2, p. 183]).

In order to prove the local controllability of a nonlinear system, the system is linearized at an equilibrium point.

Definition 7.1 (Linearized system)**.** The linearized system of the nonlinear system $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$ at an equilibrium state $(\mathbf{x}_e, \mathbf{u}_e)$ is given by

$$\begin{aligned}\dot{\mathbf{x}} &= \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_e,\mathbf{u}_e} \mathbf{x} + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}_e,\mathbf{u}_e} \mathbf{u} \\ &= \mathbf{A}\,\mathbf{x} + \mathbf{B}\,\mathbf{u} \end{aligned} \tag{7.15}$$

For the equilibrium point $(\mathbf{x}_e, \mathbf{u}_e)$, it holds $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}_e, \mathbf{u}_e) = \mathbf{0}$.

▲

The next step in the controllability analysis of a system is the definition of a suitable controllability criterion (see [81]). The well-known controllability criteria are e.g., Kalman's criterion, Gilbert's controllability criterion and Hautus lemma. In this study, Kalman's criterion is used. This criterion refers to the controllability matrix given as

$$\mathbf{C}(\mathbf{A}, \mathbf{B}) = [\mathbf{B}\ \ \mathbf{AB}\ \ \ldots\ \ \mathbf{A}^{n-1}\mathbf{B}] \in \mathbb{R}^{n \times nm} \tag{7.16}$$

Theorem 7.1 (Kalman rank condition)**.** *According to Kalman, a linear system is fully controllable if the controllability matrix* $\mathbf{C}$ *has full rank:*

$$rank\,\mathbf{C} = n.$$

Remark 7.1. The system (7.14) is locally controllable if the linearized system (7.15) is controllable according to Theorem 7.1. △

7.4.2.1 Equilibrium Point of the Multi-Agent System

With the above theorem, the local controllability of the extended model can be analyzed. However, before doing this, the possible equilibrium point of the system has to be defined. The asymptotic behavior of the system over time is of particular importance for the definition of the equilibrium point. The asymptotic behavior of a multi-agent system can be analyzed, for example, by means of dispersion and dissent (see Section 3.1.2). For a system with $i = 1, \ldots, N$ agents, e.g., the dissent $V(t) = 0$ (see 3.11) means that the velocities of all agents are matched, which is desired for the flocking behavior $t \to T$.

A flocking system is basically in a quasi-static state if velocity consensus emerges through local interactions among the agents.

Definition 7.2 (Velocity consensus)**.** Within a multi-agent system with $i = 1, \ldots, N$ agents, the consensus state is reached if the following holds:

1. $\lim\limits_{t\to\infty} \mathbf{v}_i(t) = \bar{\mathbf{v}}$ *or* equivalent
2. $\lim\limits_{t\to\infty} \Lambda(t) = 0.$

with $\bar{\mathbf{v}}(t) = \frac{1}{N}\sum_{i=1}^{N}\mathbf{v}_i(t)$. ▲

In addition to the velocity consensus, cohesion (lattice-type structure) is another requirement for flocking behavior.

Definition 7.3 (State of the lattice-type configuration)**.** If a multi-agent system has an α-lattice configuration, the following holds

$$\nabla_{\mathbf{p}_i} V(\|\mathbf{p}_j - \mathbf{p}_i\|) = \rho_h\left(\frac{||\mathbf{p}_j - \mathbf{p}_i||_\sigma}{r_\alpha}\right)\varphi(||\mathbf{p}_j - \mathbf{p}_i||_\sigma - d_\alpha) = 0, \text{ for } j \in \mathcal{N}_i \quad (7.17)$$

where $r_\alpha = \|r_c\|_\sigma$ and $d_\alpha = \|d\|_\sigma$. V denotes the smooth artificial potential (see (2.21)) and $\nabla_{\mathbf{p}_i} V = 0$ denotes its global minimum. ▲

The state in which the agents fulfill both conditions from Definition 7.2 and Definition 7.3, is interpreted as the equilibrium point of a flocking multi-agent system.

7.4.2.2 Controllability Analysis

In this section, we analyze the local controllability of the extended multi-agent system (7.10)-(7.11). For the extended model, the vector function $\mathbf{f}(\mathbf{x})$ and the input matrix $\mathbf{B}$ are defined as

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_N \\ \mathbf{G}_1 + \mathbf{C}_1 \\ \mathbf{G}_2 + \mathbf{C}_2 + \mathbf{L}_2 \\ \vdots \\ \mathbf{G}_N + \mathbf{C}_N + \mathbf{L}_N \end{pmatrix}, \qquad \mathbf{B} = \begin{pmatrix} \mathbf{0}_{m,m} \\ \mathbf{0}_{m,m} \\ \vdots \\ \mathbf{0}_{m,m} \\ \mathbf{I}_m \\ \mathbf{0}_{m,m} \\ \vdots \\ \mathbf{0}_{m,m} \end{pmatrix}, \quad (7.18)$$

where $\mathbf{0}_{m,m} \in \mathbb{R}^{m\times m}$ is the null matrix and $\mathbf{I}_m$ is an m-dimensional unit matrix. With the state vector $\mathbf{x}$ defined as

$$\mathbf{x} = \begin{pmatrix} \mathbf{p}_q \\ \mathbf{v}_q \end{pmatrix}, \quad (7.19)$$

where the vector $\mathbf{p}_q := (\mathbf{p}_1^\top, \mathbf{p}_1^\top, \ldots, \mathbf{p}_N^\top)^\top \in \mathbb{R}^{m \cdot N}$ includes the positions of the agents and $\mathbf{v}_q := (\mathbf{v}_1^\top, \mathbf{v}_1^\top, \ldots, \mathbf{v}_N^\top)^\top \in \mathbb{R}^{m \cdot N}$ are the velocities of the agents. The vector function is rewritten as

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} \mathbf{f}_p \\ \mathbf{f}_v \end{pmatrix}, \tag{7.20}$$

with

$$\mathbf{f}_p = \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_N \end{pmatrix}, \quad \mathbf{f}_v = \begin{pmatrix} \mathbf{G}_1 + \mathbf{C}_1 \\ \mathbf{G}_2 + \mathbf{C}_2 + \mathbf{L}_2 \\ \vdots \\ \mathbf{G}_N + \mathbf{C}_N + \mathbf{L}_N \end{pmatrix}. \tag{7.21}$$

where $\mathbf{f}_p \in \mathbb{R}^{m \cdot N}$ and $\mathbf{f}_v \in \mathbb{R}^{m \cdot N}$ represent the parts of system dynamics with respect to position and velocity. The linearized multi-agent system can thus be written as $\dot{\mathbf{x}} = \mathbf{A}\,\mathbf{x} + \mathbf{B}\,\mathbf{u}$, where $\mathbf{A}$ denotes the Jacobian matrix of $\mathbf{f}(\mathbf{x})$ at the equilibrium point $\mathbf{x}^* = (\,\mathbf{p}_q^*,\, \mathbf{v}_q^*\,)^\top$. The Jacobian matrix is given by

$$\mathbf{A} = \nabla \mathbf{f}(\mathbf{x}^*) = \begin{pmatrix} \nabla_p \mathbf{f}_p & \nabla_v \mathbf{f}_p \\ \nabla_p \mathbf{f}_v & \nabla_v \mathbf{f}_v \end{pmatrix} \tag{7.22}$$

with

$$\begin{aligned} &\nabla_p \mathbf{f}_p = \mathbf{0}_{m \cdot N, m \cdot N}, \\ &\nabla_v \mathbf{f}_p = \mathbf{I}_{m \cdot N}, \\ &\nabla_p \mathbf{f}_v = \nabla_p \mathbf{G}_i + \nabla_p \mathbf{C}_i + \nabla_p \mathbf{L}_b, \\ &\nabla_v \mathbf{f}_v = \nabla_v \mathbf{C}_i. \end{aligned} \tag{7.23}$$

The individual derivative terms from (7.23) are as follows:

$$\begin{aligned} &\nabla_p \mathbf{G}_i = [\partial \mathbf{G}_{ij}], \qquad \partial \mathbf{G}_{ij} := \left(\frac{\partial \mathbf{G}_i}{\partial \mathbf{p}_j} \right), \\ &\nabla_p \mathbf{C}_i = [\partial \mathbf{C}_{ij}], \qquad \partial \mathbf{C}_{ij} := \left(\frac{\partial \mathbf{C}_i}{\partial \mathbf{p}_j} \right), \\ &\nabla_p \mathbf{L}_b = [\partial \mathbf{L}_{ij}], \qquad \partial \mathbf{L}_{ij} := \left(\frac{\partial \mathbf{L}_i}{\partial \mathbf{p}_j} \right), \\ &\nabla_v \mathbf{C}_i = [\partial \mathbf{C}_{ij}], \qquad \partial \mathbf{C}_{ij} := \left(\frac{\partial \mathbf{C}_i}{\partial \mathbf{v}_j} \right), \end{aligned} \tag{7.24}$$

with the partial derivatives

$$\begin{aligned}\frac{\partial \mathbf{G}_i}{\partial \mathbf{p}_i} = \sum_{j\neq i, j\in\mathcal{N}_i}^{N} & \frac{\partial \mathbf{n}_{ij}}{\partial \mathbf{p}_i}\,\rho_h(||\mathbf{p}_j-\mathbf{p}_i||_\sigma/r_\alpha)\,\varphi(||\mathbf{p}_j-\mathbf{p}_i||_\sigma-d_\alpha)\\ & - \mathbf{n}_{ij}\,\frac{\partial \rho_h(||\mathbf{p}_j-\mathbf{p}_i||_\sigma/r_\alpha)}{\partial \mathbf{p}_i}\,\varphi(||\mathbf{p}_j-\mathbf{p}_i||_\sigma-d_\alpha)\\ & + \mathbf{n}_{ij}\,\rho_h(||\mathbf{p}_j-\mathbf{p}_i||_\sigma/r_\alpha)\,\frac{\partial \varphi(||\mathbf{p}_j-\mathbf{p}_i||_\sigma-d_\alpha)}{\partial \mathbf{p}_i},\end{aligned} \tag{7.25}$$

$$\begin{aligned}\frac{\partial \mathbf{G}_i}{\partial \mathbf{p}_j} = & \frac{\partial \mathbf{n}_{ij}}{\partial \mathbf{p}_j}\,\rho_h(||\mathbf{p}_j-\mathbf{p}_i||_\sigma/r_\alpha)\,\varphi(||\mathbf{p}_j-\mathbf{p}_i||_\sigma-d_\alpha)\\ & + \mathbf{n}_{ij}\,\frac{\partial \rho_h(||\mathbf{p}_j-\mathbf{p}_i||_\sigma/r_\alpha)}{\partial \mathbf{p}_j}\,\varphi(||\mathbf{p}_j-\mathbf{p}_i||_\sigma-d_\alpha)\\ & + \mathbf{n}_{ij}\,\rho_h(||\mathbf{p}_j-\mathbf{p}_i||_\sigma/r_\alpha)\,\frac{\partial \varphi(||\mathbf{p}_j-\mathbf{p}_i||_\sigma-d_\alpha)}{\partial \mathbf{p}_j},\end{aligned} \tag{7.26}$$

$$\frac{\partial \mathbf{C}_i}{\partial \mathbf{p}_i} = \sum_{j\neq i, j\in\mathcal{N}_i}^{N} (\mathbf{v}_j-\mathbf{v}_i)\,\frac{\partial \rho_h(||\mathbf{p}_j-\mathbf{p}_i||_\sigma/r_\alpha)}{\partial \mathbf{p}_i}, \tag{7.27}$$

$$\frac{\partial \mathbf{C}_i}{\partial \mathbf{p}_j} = (\mathbf{v}_j-\mathbf{v}_i)\,\frac{\partial \rho_h(||\mathbf{p}_j-\mathbf{p}_i||_\sigma/r_\alpha)}{\partial \mathbf{p}_j}, \tag{7.28}$$

$$\frac{\partial \mathbf{C}_i}{\partial \mathbf{v}_i} = -\sum_{j\neq i, j\in\mathcal{N}_i}^{N} \mathbf{I}_m\,\rho_h\left(\frac{||\mathbf{p}_j-\mathbf{p}_i||_\sigma}{r_\alpha}\right), \tag{7.29}$$

$$\frac{\partial \mathbf{C}_i}{\partial \mathbf{v}_j} = \mathbf{I}_m\,\rho_h\left(\frac{||\mathbf{p}_j-\mathbf{p}_i||_\sigma}{r_\alpha}\right), \tag{7.30}$$

$$\begin{aligned}\frac{\partial \mathbf{L}_b}{\partial \mathbf{p}_1} = \lambda\Bigg(& \frac{\partial \mathbf{n}_{b1}}{\partial \mathbf{p}_1}\,\rho_h(||\mathbf{p}_1-\mathbf{p}_b||_\sigma/r_\alpha)\,\varphi(||\mathbf{p}_1-\mathbf{p}_b||_\sigma-d_\alpha)\\ & + \mathbf{n}_{b1}\,\frac{\partial \rho_h(||\mathbf{p}_1-\mathbf{p}_b||_\sigma/r_\alpha)}{\partial \mathbf{p}_1}\,\varphi(||\mathbf{p}_1-\mathbf{p}_b||_\sigma-d_\alpha)\\ & + \mathbf{n}_{b1}\,\rho_h(||\mathbf{p}_1-\mathbf{p}_b||_\sigma/r_\alpha)\,\frac{\partial \varphi(||\mathbf{p}_1-\mathbf{p}_b||_\sigma-d_\alpha)}{\partial \mathbf{p}_1}\Bigg),\end{aligned} \tag{7.31}$$

$$\begin{aligned}\frac{\partial \mathbf{L}_b}{\partial \mathbf{p}_b} = \lambda \Bigg(& \frac{\partial \mathbf{n}_{b1}}{\partial \mathbf{p}_b} \, \rho_h(||\mathbf{p}_1 - \mathbf{p}_b||_\sigma / r_\alpha) \, \varphi(||\mathbf{p}_1 - \mathbf{p}_b||_\sigma - d_\alpha) \\ & + \mathbf{n}_{b1} \, \frac{\partial \rho_h(||\mathbf{p}_1 - \mathbf{p}_b||_\sigma / r_\alpha)}{\partial \mathbf{p}_b} \, \varphi(||\mathbf{p}_1 - \mathbf{p}_b||_\sigma - d_\alpha) \\ & + \mathbf{n}_{b1} \, \rho_h(||\mathbf{p}_1 - \mathbf{p}_b||_\sigma / r_\alpha) \, \frac{\partial \varphi(||\mathbf{p}_1 - \mathbf{p}_b||_\sigma - d_\alpha)}{\partial \mathbf{p}_b} \Bigg).\end{aligned} \tag{7.32}$$

The partial derivatives of the σ-norm for the i-th and j-th agent are given as

$$\frac{\partial ||\mathbf{p}_j - \mathbf{p}_i||_\sigma}{\partial \mathbf{p}_j} = \sqrt{\frac{1}{1 + \varepsilon \, ||\mathbf{p}_j - \mathbf{p}_i||^2}} \, (\mathbf{p}_j - \mathbf{p}_i)^\top, \tag{7.33}$$

$$\begin{aligned}\frac{\partial ||\mathbf{p}_j - \mathbf{p}_i||_\sigma}{\partial \mathbf{p}_i} &= -\sqrt{\frac{1}{1 + \psi \, ||\mathbf{p}_j - \mathbf{p}_i||^2}} \, (\mathbf{p}_j - \mathbf{p}_i)^\top \\ &= -\frac{\partial ||\mathbf{p}_j - \mathbf{p}_i||_\sigma}{\partial \mathbf{p}_j}.\end{aligned} \tag{7.34}$$

Analogously, the partial derivatives of the vector $\mathbf{n}_{ij}$, the functions $\rho_h(z)$ and $\varphi(p)$ of the action function φ_α can be written as follows:

$$\begin{aligned}\frac{\partial \mathbf{n}_{ij}}{\partial \mathbf{p}_j} = & \frac{\mathbf{I}_d}{\sqrt{1 + \varepsilon \, ||\mathbf{p}_j - \mathbf{p}_i||^2}} \\ & - \varepsilon \, \Big(1 + \, \varepsilon \, ||\mathbf{p}_j - \mathbf{p}_i||^2\Big)^{-\frac{3}{2}} \, (\mathbf{p}_j - \mathbf{p}_i)(\mathbf{p}_j - \mathbf{p}_i)^\top,\end{aligned} \tag{7.35}$$

$$\frac{\partial \rho_h(z)}{\partial \mathbf{p}_j} = \begin{cases} 0, & z \in [0, h) \\ -\frac{\pi}{2\,(1-h) r_\alpha} \frac{\partial ||\mathbf{p}_j - \mathbf{p}_i||_\sigma}{\partial \mathbf{p}_j} \sin\left(\frac{\pi(z-h)}{1-h}\right), & z \in [h, 1] \\ 0, & \text{otherwise} \end{cases} \tag{7.36}$$

with $z = \frac{||\mathbf{p}_j - \mathbf{p}_i||_\sigma}{r_\alpha}$,

$$\frac{\partial \varphi(k)}{\partial \mathbf{p}_j} = \frac{1}{2}(a+b) \, \frac{\partial ||\mathbf{p}_j - \mathbf{p}_i||_\sigma}{\partial \mathbf{p}_j} \Big(1 + (k+c)^2\Big)^{-\frac{1}{2}} \Big(1 - (k+c)^2 \Big(1 + (k+c)^2\Big)^3\Big), \tag{7.37}$$

with $k = ||\mathbf{p}_j - \mathbf{p}_i||_\sigma - d_\alpha$.

$$\frac{\partial \mathbf{n}_{ji}}{\partial \mathbf{p}_i} = -\frac{\partial \mathbf{n}_{ji}}{\partial \mathbf{p}_j}, \tag{7.38}$$

$$\frac{\partial \rho_h(z)}{\partial \mathbf{p}_i} = -\frac{\partial \rho_h(z)}{\partial \mathbf{p}_j}, \tag{7.39}$$

$$\frac{\partial \varphi(k)}{\partial \mathbf{p}_i} = -\frac{\partial \varphi(k)}{\partial \mathbf{p}_j}. \tag{7.40}$$

All partial derivatives not listed are zero. According to Definition 7.1 and Definition 7.2, the following should hold at the equilibrium point of the extended model:

$$\rho_h\left(\frac{||\mathbf{p}_j^* - \mathbf{p}_i^*||_\sigma}{r_\alpha}\right) \varphi(||\mathbf{p}_j^* - \mathbf{p}_i^*||_\sigma - d_\alpha) = 0 \tag{7.41}$$

and

$$\mathbf{v}_1^* = \mathbf{v}_2^* = \cdots = \mathbf{v}_N^*. \tag{7.42}$$

Thus, the matrix $\mathbf{A} = \nabla \mathbf{f}(\mathbf{x}^*)$ at an equilibrium point can be formulated using the following derivative terms:

$$\begin{aligned}\left.\frac{\partial \mathbf{G}_i}{\partial \mathbf{p}_i}\right|_{\mathbf{x}^*} = &- \sum_{j \neq i, j \in \mathcal{N}_i}^{N} \mathbf{n}_{ij} \frac{\partial \rho_h(||\mathbf{p}_j^* - \mathbf{p}_i^* i||_\sigma / r_\alpha)}{\partial \mathbf{p}_i} \varphi(||\mathbf{p}_j^* - \mathbf{p}_i^*||_\sigma - d_\alpha) \\ &+ \mathbf{n}_{ij}\, \rho_h(||\mathbf{p}_j^* - \mathbf{p}_i^*||_\sigma / r_\alpha) \frac{\partial \varphi(||\mathbf{p}_j^* - \mathbf{p}_i^*||_\sigma - d_\alpha)}{\partial \mathbf{p}_i},\end{aligned} \tag{7.43}$$

$$\begin{aligned}\left.\frac{\partial \mathbf{G}_i}{\partial \mathbf{p}_j}\right|_{\mathbf{x}^*} = &\, \mathbf{n}_{ij} \frac{\partial \rho_h(||\mathbf{p}_j^* - \mathbf{p}_i^*||_\sigma / r_\alpha)}{\partial \mathbf{p}_j} \varphi(||\mathbf{p}_j^* - \mathbf{p}_i^*||_\sigma - d_\alpha) \\ &+ \mathbf{n}_{ij}\, \rho_h(||\mathbf{p}_j^* - \mathbf{p}_i^*||_\sigma / r_\alpha) \frac{\partial \varphi(||\mathbf{p}_j^* - \mathbf{p}_i^*||_\sigma - d_\alpha)}{\partial \mathbf{p}_j},\end{aligned} \tag{7.44}$$

$$\left.\frac{\partial \mathbf{C}_i}{\partial \mathbf{p}_i}\right|_{\mathbf{x}^*} = \mathbf{0}, \tag{7.45}$$

$$\left.\frac{\partial \mathbf{C}_i}{\partial \mathbf{p}_j}\right|_{\mathbf{x}^*} = \mathbf{0}, \tag{7.46}$$

$$\left.\frac{\partial \mathbf{L}_b}{\partial \mathbf{p}_1}\right|_{\mathbf{x}^*} = \lambda \left(\mathbf{n}_{b1} \frac{\partial \rho_h(||\mathbf{p}_1^* - \mathbf{p}_b^*||_\sigma / r_\alpha)}{\partial \mathbf{p}_1} \varphi(||\mathbf{p}_1^* - \mathbf{p}_b^*||_\sigma - d_\alpha) + \mathbf{n}_{b1}\, \rho_h(||\mathbf{p}_1^* - \mathbf{p}_b^*||_\sigma / r_\alpha) \frac{\partial \varphi(||\mathbf{p}_1^* - \mathbf{p}_b^*||_\sigma - d_\alpha)}{\partial \mathbf{p}_1} \right), \tag{7.47}$$

$$\left.\frac{\partial \mathbf{L}_b}{\partial \mathbf{p}_b}\right|_{\mathbf{x}^*} = \lambda \left(\mathbf{n}_{b1} \frac{\partial \rho_h(||\mathbf{p}_1^* - \mathbf{p}_b^*||_\sigma / r_\alpha)}{\partial \mathbf{p}_b} \varphi(||\mathbf{p}_1^* - \mathbf{p}_b^*||_\sigma - d_\alpha) + \mathbf{n}_{b1}\, \rho_h(||\mathbf{p}_1^* - \mathbf{p}_b^*||_\sigma / r_\alpha) \frac{\partial \varphi(||\mathbf{p}_1^* - \mathbf{p}_b^*||_\sigma - d_\alpha)}{\partial \mathbf{p}_b} \right), \tag{7.48}$$

$$\left.\frac{\partial \mathbf{C}_i}{\partial \mathbf{v}_i}\right|_{\mathbf{x}^*} = - \sum_{j \neq i, j \in \mathcal{N}_i}^{N} \mathbf{I}_d\, \rho_h \left(\frac{||\mathbf{p}_j^* - \mathbf{p}_i^*||_\sigma}{r_\alpha} \right), \tag{7.49}$$

$$\left.\frac{\partial \mathbf{C}_i}{\partial \mathbf{v}_j}\right|_{\mathbf{x}^*} = \mathbf{I}_d\, \rho_h \left(\frac{||\mathbf{p}_j^* - \mathbf{p}_i^*||_\sigma}{r_\alpha} \right). \tag{7.50}$$

The linearized system $\dot{\mathbf{x}} = \mathbf{A}\,\mathbf{x} + \mathbf{B}\,\mathbf{u}$ with $\mathbf{A} = \nabla \mathbf{f}(\mathbf{x}^*)$ is controllable if the controllability matrix

$$\mathbf{C}(\mathbf{A}, \mathbf{B}) = [\mathbf{B} \;\; \mathbf{AB} \;\; \mathbf{A}^2\mathbf{B} \;\; \ldots \;\; \mathbf{A}^{2mN-1}\mathbf{B}] \tag{7.51}$$

has full rank.

7.4.3 Formulation of the Optimal Control Problem

Now we can write the optimal control problem subject to the previously introduced dynamics (7.10) and (7.11) for $t \in [0, T]$.

$$\min_{\mathbf{p},\mathbf{u}} \; J(\mathbf{p}, \mathbf{u}) = \frac{1}{2}||\mathbf{p}_1(t) - \mathbf{p}_d(T)||^2 + \frac{\mu}{2} \sum_{b=1}^{N} \int_0^T ||\mathbf{p}_1(t) - \mathbf{p}_b(t)||^4 \, \mathrm{d}t + \frac{\nu}{2} \int_0^T ||\mathbf{u}(t)||^2 \, \mathrm{d}t \tag{7.52}$$

$$\begin{aligned} \text{subject to} \quad & \dot{\mathbf{p}}_1 = \mathbf{v}_1, \\ & \dot{\mathbf{v}}_1 = \mathbf{G}_1 + \mathbf{C}_1 + \mathbf{u}, \\ & \dot{\mathbf{p}}_b = \mathbf{v}_b, \\ & \dot{\mathbf{v}}_b = \mathbf{G}_b + \mathbf{C}_b + \mathbf{L}_b, \quad b = 2, \ldots, N \end{aligned} \tag{7.53}$$

with the given initial states

$$\mathbf{p}(t_0) = \mathbf{p}(t = 0) = \mathbf{p}_0,$$
$$\mathbf{v}(t_0) = \mathbf{v}(t = 0) = \mathbf{v}_0.$$

The parameters μ and ν are positive constants. $\mathbf{p}_d$ denotes the desired target position. The first term of the cost functional (7.52) aims at minimizing the distance between the leader and the desired position. The second term is for the minimization of the distance between the leader and the uninformed agents considering the additional parameter T that scales the final time. The power of this term is heuristically determined. The third term is responsible for the minimum effort. For the objective of trajectory tracking, $\mathbf{p}_d(T)$ is replaced by $\mathbf{p}_d(t)$, which represents the desired trajectory.

In order to plan a collision-free trajectory, we consider static obstacles, which are located between the agents and the desired target position. For this purpose, we include the following inequality constraint:

$$h(\mathbf{p}(t), \mathbf{u}) : ||\mathbf{p}_i(t) - \mathbf{O}_k|| - r_k > d_s, \quad d_s > 0, \tag{7.54}$$

with $k \in \mathcal{O}$ and $\mathcal{O}$ denotes the set of all obstacles. Moreover, d_s is the safe distance. We consider only circular obstacles with radius r_k and center position $\mathbf{O}_k$.

The state variables are not constraint. However, we have an additional final condition:

$$g(\mathbf{p}(t), \mathbf{u}) : ||\mathbf{p}_1(t) - \mathbf{p}_d(T)|| = 0. \tag{7.55}$$

7.5 Simulation Studies

In this section, we present the results of numerical studies with the proposed optimal control scheme. The OCP is implemented using MATLAB and the CasADi framework. In order to solve the OCP, we used the interior point solver (IPOPT) which has an internal maximum iteration number of 3000 iterations. For the discretization, we applied a simple fixed-step fourth-order Runge-Kutta (RK4), which is implemented using CasADi symbolics. The optimization problem is solved using the software package IPOPT for large-scale nonlinear optimization [141]. The implementation is done in MATLAB 2018b using CasADi 3.2.2 on a computer with Intel i7-4770 CPU @ 3.4 GHz.

In order to evaluate the effectiveness of the proposed optimal control scheme, we consider two tasks. The first task is steering the group to a target point by avoiding obstacles and the second task is tracking a given trajectory. Corresponding to tasks, the values of the parameters of the system for the simulations are given in Table 7.1. In addition, the parameters for discretization of the OCP are shown in Table 7.2.

Table 7.1: Parameters of the system.

		Task	
		1	2
Model parameters	d	2	1.5
	r_c	10	7.5
	ε	0.1	0.1
	a	5	5
	b	10	10
	h	0.2	0.2
	λ	1.4	1.4
Cost functional	μ	1	1
	ν	1	1
Obstacle avoidance	d_s	0.5	-

Table 7.2: Parameters of discretization for the OCP.

Notation	Parameters	Value	
		Task 1	Task 2
Time horizon	T	10	20
Number of control intervals	I	300	600
Runge–Kutta order	M	4	4

7.5.1 Task 1: Moving to a Target Point with Obstacle Avoidance

In this task, the target position is defined at $\mathbf{p}_d = (15, 20)^\top$. The leader's starting position is $\mathbf{p}_1 = (4, 5)^\top$ and its initial velocity is $\mathbf{v}_1 = (1, 1)^\top$. The uninformed agents $b = 2, \ldots, N$ are randomly placed in a rectangular area $[3.5, 6] \times [4, 6.5]$ and their initial velocities $\mathbf{v}_b(0) \in \mathbb{R}^2$ are randomly selected from $[-0.8, -0.8]$. The initial guess for solving the optimal control problem is a straight line linking the leader-agent's initial position and the target position. We use a similar initial guess for the rest of the group. However, the lines are shifted by the value of their starting positions and they have identical slopes. We do not assume any initial guess for the control input $\mathbf{u}(t)$, which is a piece-wise continuous function with constant values on intervals. In simulations, we consider three circular obstacles.

Fig. 7.1 shows the positions of the agents for the intervals $I = [0, 75, 150, 225, 300]$ and the optimal trajectory of the leader guiding 6 agents. The blue triangles represent the positions and the directions of motion of the agents and the red triangle corresponds to the leader-agent.

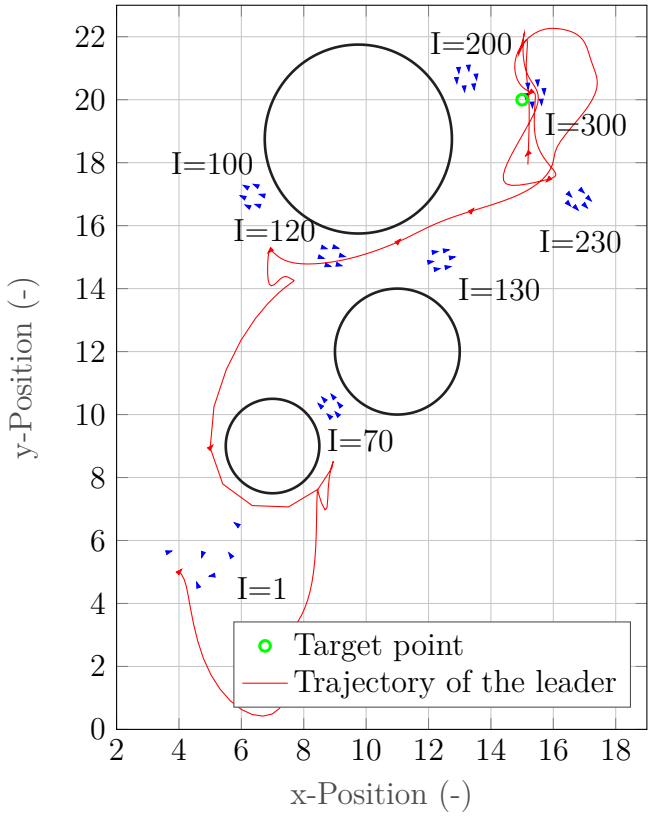


Figure 7.1: Interval snapshots of a group of $N = 7$ agents performing Task 1.

Fig. 7.2 illustrates the trajectories of the agents. Note that the initial values of the agents heavily influence the optimal solution and also the trajectories of the agents.

Furthermore, Fig. 7.3 represents the evolution of the dissent[3] between the uninformed agents. It can be seen that the dissent oscillates within a certain range. The reason for this is the agile maneuvers of the leader-agent. For example, the leader-agent hides itself quickly from the group at $I = 70$ in order to reduce its influence on the group. We can observe this rapid change in dissent (Fig. 7.3) as well.

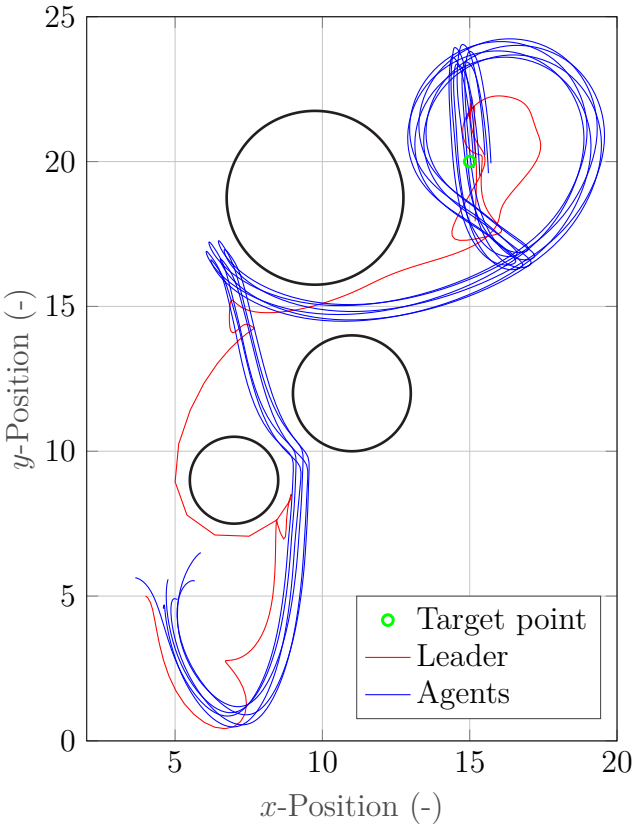


Figure 7.2: Trajectories of a group of $N = 7$ agents performing Task 1.

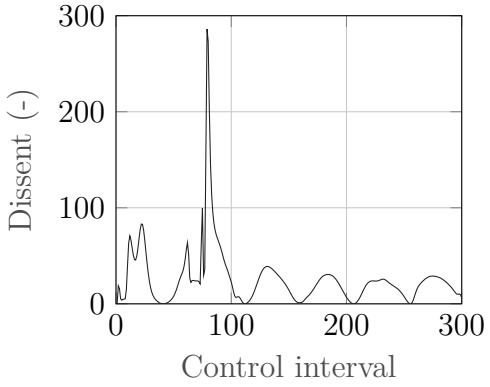


Figure 7.3: Dissent of a group of $N = 7$ agents during Task 1.

7.5.2 Task 2: Trajectory Tracking

The difference between the cost functionals of the first and second task is that the desired target position is time-variant $\mathbf{p}_d = \mathbf{p}_d(t)$. For each time step, the target position $\mathbf{p}_d$ travels along the desired trajectory. The initial guess for the solution of the trajectories consists of two parts: A line linking the leader-agent's position and a given point on the circular trajectory and the desired circular trajectory itself. The defined line in the first part of the initial guess touches the circle tangentially. Moreover, for the rest of agents, we have the same initial trajectory guesses, but they are shifted by

[3]Dissent was defined in Eq. (3.11).

the value of their starting positions relatively to the position of the leader-agent. In the simulation, the reference trajectory is a circle with the radius 4.5 and the center at $(5.5, 8)^\top$. The group objective is moving along the circular path twice. The leader is placed at $\mathbf{p}_1 = (12.5, 10.5)^\top$ with the initial velocity $\mathbf{v}_1 = (-0.5, -0.5)^\top$.

Fig. 7.4 illustrates the positions of the agents for the intervals $I = [0, 130, 200, 250, 350, 400, 550, 600]$ and the optimal trajectory of the leader for a system with $N = 7$ agents. The trajectories of the agents are represented in Fig. 7.5. In addition, Fig. 7.6 shows the distance between the leader and the trajectory d_{1t} and the distance between the flock center and the trajectory d_{ct}. According to Fig. 7.6, it is clear that the agents are far away from the trajectory at the beginning because of the initial positions. The evolution of d_{ct} shows that the group can track the trajectory well despite the agents' lack of information concerning the trajectory. However, the tracking behavior can be improved with a different parameter set. Moreover, it would be better for another trajectory with a greater radius. In addition, minimizing the step size in the simulation by reducing the number of control intervals might also provide a better tracking.

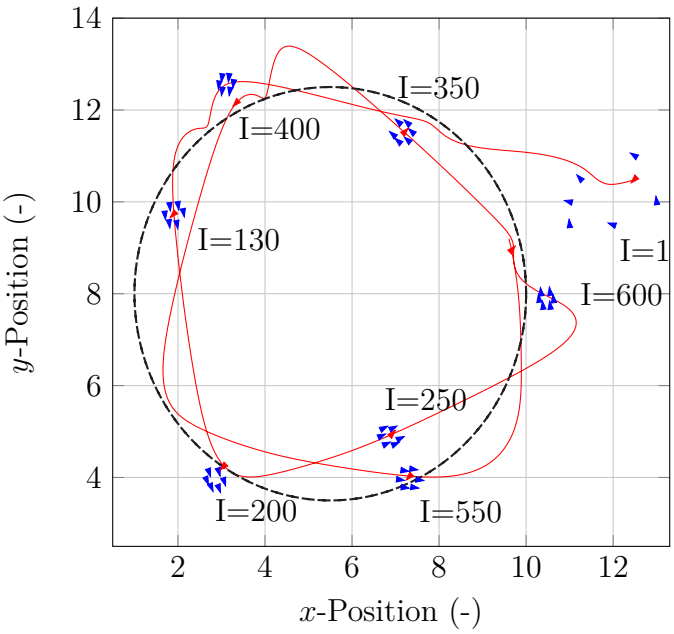


Figure 7.4: Interval snapshots of a group of $N = 7$ agents performing Task 2.

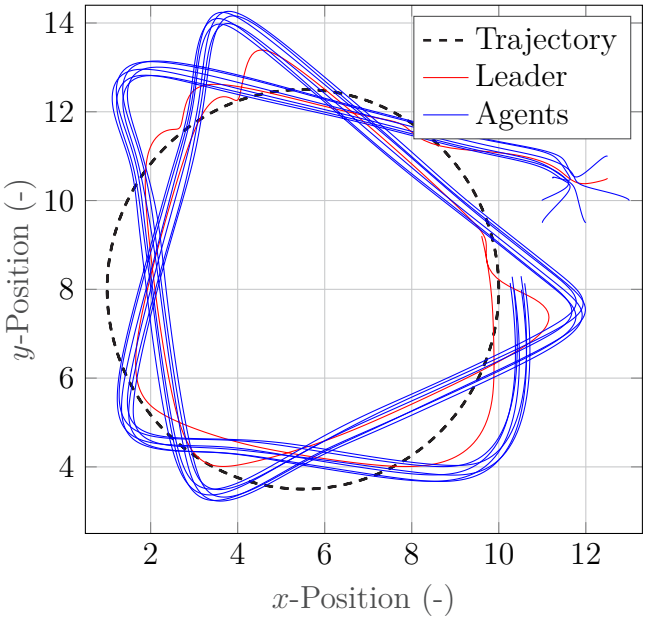


Figure 7.5: Trajectories of a group of $N = 7$ agents performing Task 2.

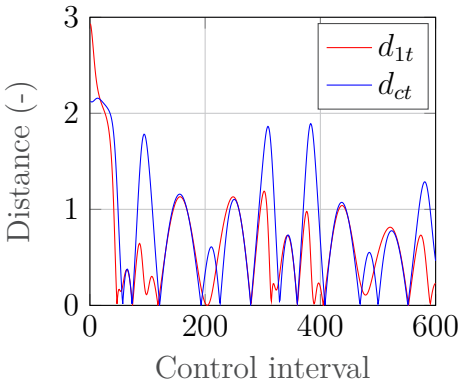


Figure 7.6: Distance d_{1t} and d_{ct} of a group of $N = 7$ agents during Task 2.

7.5.3 Remarks on the Proposed Approach

The results of this work have shown that a multi-agent system applying the flocking algorithm in [100] can fulfill a task without information about the objective and the environment if guidance is provided by a single optimally controlled leader-agent. It has been observed that the distances between the leader-agent and the rest of the group can be very small during trajectory tracking. The reason for this may be the influence of the leader's agile movements. This can be solved with an additional inequality constraint which should guarantee a minimal allowed distance between the leader-agent and the other agents. However, this would restrict the solution space of the OCP.

The direct multiple-shooting approach is prone to extreme nonlinearities in the cost function and in constraints. Hence, another direct approach such as direct collocation can be considered to obtain better results. In addition, the quality of the solution in this approach heavily depends on the initial guess. Trajectories, which are inside the solution set, obtained from previous optimizations with a fewer number of agents can serve as the initial guess instead of simple straight lines toward target position. As an alternative, using the A* algorithm [59] can also provide better initial guesses that are close enough to the solution set.

In order to analyze the optimization quality with respect to the size of the group, simulations with 3, 4, 5, 6 and 7 agents were performed for the Task 1. Fig. 7.7 shows the values of the cost functional at the end of the optimization obtained from 20 simulations for each group by randomly placing the agents in the above-mentioned rectangular area. Apparently, the values of the cost functional tend to increase as the

number of the agents increases. This is an expected result because the higher number of states increases the number of inequality and equality constraints. This narrows down the solution space that satisfies all problem constraints and the optimization problem becomes more complex.

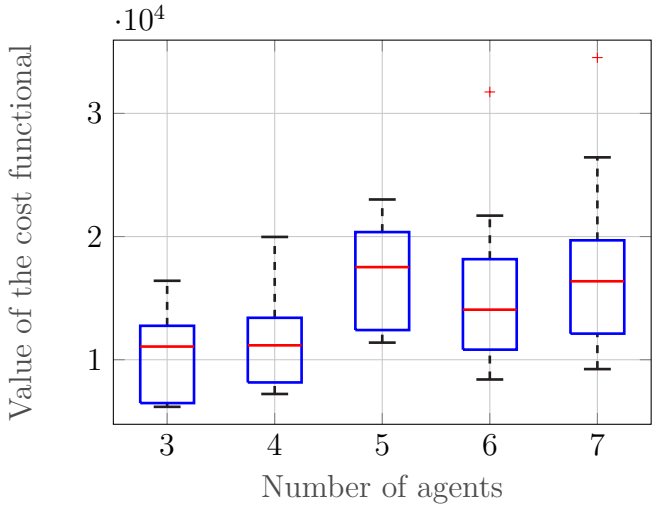


Figure 7.7: Box-plot for Task 1[4].

7.6 Chapter Highlights

In this chapter, we introduced an optimally controlled leader-agent concept and its theoretical application by extending the flocking algorithm from [100] in the form of a hierarchical structure according to [146]. We simulated and evaluated the presented concept. This includes a leader-agent that knows the predetermined group objective beforehand and interacts with other agents in the group. Through optimal guidance of the leader, we can steer a group of agents with limited information toward an objective. Optimal control inputs for the leader were determined by using the IPOPT solver provided by the open-source tool CasADi. In this study, we defined two tasks for the group: Moving to a target position with obstacle avoidance and tracking a predefined trajectory.

With the presented concept, it is possible to find a high number of optimal solutions for both tasks. However, the optimal solution might fail depending on the initial states and system parameters because of the fact that the IPOPT may find a local optimum. Limited sensing capabilities are also a factor that can restrict the achievable performance. Moreover, for *larger* multi-agent systems, it has been observed that the agents move too close to each other. Nonetheless, the group reaches the target position without colliding

[4]The median of the data is indicated by a horizontal bar. Each box includes data values ranging from the first to the third quartile. The whiskers indicate the farthest data points that are within 1.5 times of the interquartile range. The outliers are shown with a plus symbol.

with obstacles. The theoretical concept proposed in this chapter can be extended to the coordination and navigation of multi-vehicle systems or warehouse logistics.

The supervised student thesis [163] has contributed to the development of this chapter's results.

8 Summary and Outlook

This thesis presents several approaches for distributed coordination, navigation and control for multi-agent systems. We have mainly addressed three different types of problems with respect to cooperative navigation: Collective local motion planning, coverage-oriented motion planning and global motion planning for a group of agents through an external leader. In this chapter, we now summarize the main contents, provide final remarks and motivations for future research.

8.1 Summary

This thesis is devoted to develop motion planning strategies for different problem sets. The focus is set on the theoretical development of motion planners that assist a group of interacting agents. Chapters 3, 4, 5 and 6 deal with online motion planning using distributed control methods. Moreover, Chapter 7 employs an offline motion planning strategy, where the motion planner computes the path before the start of motion. In addition, except for the particle system presented in Chapter 3, all approaches require only neighbor-to-neighbor information exchange. Since the thematic center point of the thesis is the motion planner, we consider only holonomic agents to preserve simplicity of the concepts. Furthermore, in all chapters, we deal with static obstacles.

In Chapter 3, we investigate the application potential of a Newton-type particle system, the CD model, which assumes all-to-all communication ability among particles. This means that each agent can exchange information with all other agents regardless of their locations. After analyses of inter-agent collision property of the CD model, we conclude that the system preserves its collision-free behavior despite an additional control input for navigation. In addition, inspired by the concept in [100], we propose an extended CD model with the aim to navigate it through a virtual leader without collision with different shaped convex obstacles.

In Chapter 4, we are concerned with another rendezvous problem combining a map-based schema with the flocking dynamics from [100]. The focus of this chapter is to recognize and eliminate local minima in the existence of concave obstacles. The proposed approach requires virtual map construction and neighbor-to-neighbor exchange of aggregated information about the unknown workspace. In this problem set, each agent has a communication mechanism with a limited communication range. In addition, each agent can localize itself and detect obstacle locations within its limited sensing range. The presented work in this chapter builds upon the flocking algorithm presented in [100]. In addition to potential field forces, we introduce an auxiliary input to generate tangential forces, which result in a rotational motion of the agent. However,

this does not solely allow the agent to escape from many concave obstacles. In order to eliminate local minima, each agent sets a virtual temporary target point by sharing and processing local information. In this way, it is possible to increase the success rate in avoiding local minima.

Chapter 5 also deals with the same problem as the previous chapter. In the strategy proposed in this chapter, the group's cohesive behavior and collective decision-making have a special priority. The difference between this chapter and Chapter 4 is that in this chapter, the agents share only critical points for motion planning instead of constructing a full map of the workspace by storing all utilized sensing points. This releases the memory of agents. However, in addition to the critical points, each agent shares and memorizes relevant information about its orientation and motion status, which gives hints about its current and sometimes past actions during navigation. Although the agents do not store all of this information during the whole operation, one drawback is the increased communication effort because of intense information exchange. Despite it, the sophisticated communication architecture raises the awareness of the agents for the unknown environment and enables them to find an appropriate path to a desired, joint target position in complex areas.

Chapter 6 is devoted to area coverage with multiple mobile agents. The abilities of agents regarding the way of communication and obstacle detection are identical to those in Chapter 4 and 5. However, for an efficient, coverage-oriented motion planning, agents have to preferably move away from each other instead of building flocks. Thus, the anti-flocking strategy is considered to accelerate the coverage. For this purpose, similar to the map creation in Chapter 4, the workspace is described by a finite set of cells. Using the virtual map of the workspace, communicated information and sensor data, each agent plans elaborated motions and identifies its environment at the same time. For the proposed algorithm for the complete coverage of an unknown workspace, the mobile agents do not require any information about the obstacles. Furthermore, they are able to virtually reconstruct the obstacles, identify inaccessible regions inside of them and finalize exploration task on their own.

Chapter 7 deals with the navigation of a group through a single intelligent agent. In contrast to the previous methods presented in this thesis, we employ an offline motion planning approach for rendezvous problem and for trajectory tracking. This requires a static environment formerly known to the intelligent agent, the so-called leader-agent. Through *optimally* planned motion, which is calculated in advance, the leader influences the behavior of the group of agents. Hereby, the challenging part of the problem set is that the regular agents do not know the group objective. Furthermore, they cannot detect obstacles and do not have information about the environment. For this task, we first formulate a constrained optimal control problem. The controller is implemented within the dynamics of the leader-agent. Then, the optimal control problem is converted into a nonlinear programming problem using a direct single shooting approach and solved by the interior-point method. For the calculation of derivations, we used algorithmic differentiation provided by the software framework CasADi.

8.2 Final Remarks

The results of the approaches presented in this thesis provide interesting insights into the cooperative motion planning of mobile multi-agent systems, especially in unknown or partially known operation areas. The theoretical investigation of self-driven active particles in Chapter 3 is significant to replicate some naturally observed flocking phenomena. However, the presented concept is not very suitable for technical applications because of two reasons. Firstly, the all-to-all communication assumed in this approach is a limiting factor due to the limited communication bandwidths of agents in real-world applications. Secondly, the distance between the agents is not easy to influence. Intuitive adjustability of distances between agents plays a crucial role to be able to scale the problem.

All navigation schemes after Chapter 3 require neighbor-to-neighbor information exchange, which allows their application using agents with relative short communication ranges. In addition, local communication makes the system easily adaptable to the change of agent number and also helps to minimize power consumption because the agents do not have to be necessarily connected to all other agents during the whole operation.

The dynamics of mobile agents are described using the point-mass model, which is the simplest model that is frequently used for motion planning. The proposed methods in this thesis are particularly designed to coordinate holonomic agents. Hence, point-mass models are legitimate to represent the robots' dynamics. In addition, through point-mass models, motion planning complexity is considerably reduced for holonomic agents.

The presented algorithms are also suitable to plan trajectories for nonholonomic agents. However, these agents should be operated at low speed to have a relative *small* trajectory tracking error. The results of the application of a flocking algorithm to a nonholonomic MAS can be found in the student thesis [164]. For mobile agents subject to nonholonomic constraints in practice, an example workflow with the presented approaches would be as follows:

1. Transformation of nonholonomic agent state vector $(x_i, y_i, v_i, \theta_i)$ to the states of point-mass model $(\mathbf{p}_i, \mathbf{v}_i)$. Note that x_i and y_i describe the position of the agent, v_i is the speed and θ_i denotes the heading angle with respect to the global coordinate system.
2. Each agent applies the point-mass-model-based navigation approach and plans its motion in discrete time for the next time step $k+1 \;\rightarrow\; (\mathbf{p}_i(k+1), \mathbf{v}_i(k+1))$.
3. The desired agent state calculated by the navigation algorithm $(\mathbf{p}_i(k+1), \mathbf{v}_i(k+1))$ is transformed back to the agent coordinates $(x_i(k+1), y_i(k+1), v_i(k+1), \theta_i(k+1))$.
4. This is inserted as a reference signal into the position and velocity controller (e.g., a closed-loop linear–quadratic regulator) of the agent and the agent is operated.

In Chapter 7, we solve an optimal control problem. Thereby, initial guess plays a significant role not only in the convergence rate, but also influences whether the algorithm convergences to an optimal solution at all. Hence, providing a good initial guess, which is close enough to the solution, is of great importance for future investigation.

Although motion planning approaches proposed in this thesis neglect measurement noise, they can be extended by DKF to obtain a good performance in the existence of measurement noise. Using DKF for a flocking system, the student thesis [167] shows an acceptable compensation of the noisy measurement of the target position.

8.3 Future Research Directions

Based on the experience gained through this thesis and considering the results and remarks provided above, we can make several suggestions for future research. There are still challenges to be dealt with in the coordination and control of multi-agent systems.

In this thesis, we address only problems on the Euclidean space. However, there are problems in which agents are operated in some nonlinear spaces, e.g., on a sphere surface. By extending the methods in the present work, some cooperative navigation tasks with multiple agents in earth and space science can be performed in an optimal way. Moreover, generalizing the algorithms for motion planning inside three-dimensional spaces is also of interest to broaden the application fields of the control schemes.

The control strategies developed in this thesis are restricted to environments with static obstacles. However, there are many operation areas with moving obstacles. Although many studies on the avoidance of dynamic obstacles have been conducted using potential fields, they usually address dot-like obstacles, which are easy to integrate into our motion planning approaches. Since the obstacles are not so simple and have their own dynamics in the real world, agents require more intelligence and better equipment to estimate the dynamics of unknown obstacles. Furthermore, moving obstacles inside narrow passages may lead to very complex problems in motion planning with multiple agents, which deserve further and deep research that can be an independent study.

The proposed approaches can be implemented on holonomic robots equipped with laser scanner and wireless communication devices. A similar experimental setup to that in [28] can be used for demonstrations in future work. In this way, communication-related practical issues, such as package loss and time delays in communication can be observed and investigated separately in a future work.

It has recently been shown that the topology of the interaction network is important for controlling mobile multi-agent systems, especially for the tracking of dynamic targets. Study [85] demonstrated that the agents are better at following a dynamic target that slowly changes its direction of motion if they are in connection with more agents. However, agents with a fewer number of neighbors can respond better to fast directional changes of the target. We can also confirm that it is difficult for a group to reach consensus in the tracking of a sinusoidal trajectory with a high frequency. Thus,

alternative network topologies for the communication can be investigated for collective tracking tasks.

Regarding the optimal control scheme proposed in Chapter 7, an analysis of different optimization methods and solvers concerning their real-time applicability for the considered problem can be a part of future work. Moreover, it is also of interest to adapt the parameters of the OCP and the system parameters in order to find optimal solutions for larger multi-agent systems in an easier way. Integrating a proper start value generator into the OCP will increase the chances to find the optimum in optimization and allow the handling of different shaped obstacles other than circular ones. In addition, extending the proposed concept to MPC in combination with machine learning to learn some model parameters online is also an interesting direction for future work to deal with dynamic environments including moving obstacles.

Appendix A

Algorithms: Collective Navigation Framework

Algorithm A.1 : Status 0: Motion toward the desired goal position

Data : $status_i$, Memory buffer

Result : $status_i(t_{k+1})$

```
1  while status_i(t_k) == 0 do
2      if N_i^β == 0 and Orientation information is available then
3          status_i(t_{k+1}) = 5;
4      else if N_i^β == 1 and Eq. (5.22)== 0 then
5          if Eq. (5.23)== 1 then
6              status_i(t_{k+1}) = 4;
7          else if Eq. (5.23)== 0 then
8              status_i(t_{k+1}) = 1;
9      else if N_i^β == 2 then
10         status_i(t_{k+1}) = 3;
11     else
12         status_i(t_{k+1}) = 0;
```

Algorithm A.2 : Status 1: Obstacle detection and tangential navigation

Data : $status_i$, Memory buffer

Result : $status_i(t_{k+1})$

while $status_i(t_k) == 1$ **do**
  if $N_i^\beta == 0$ **then**
    if *Eq. (5.11)*$== 1$ **then**
      $status_i(t_{k+1}) = 0$;
    else if *Eq. (5.11)*$== 0$ **then**
      $status_i(t_{k+1}) = 2$;
  else if $N_i^\beta == 1$ ***and*** *Eq. (5.22)*$== 0$ ***and*** *Eq. (5.23)*$== 1$ **then**
    $status_i(t_{k+1}) = 4$;
  else if $N_i^\beta == 2$ **then**
    $status_i(t_{k+1}) = 3$;
  else if *Relevant information about a corner is available (see Eq. (5.21))* **then**
    $status_i(t_{k+1}) = 3$;
  else
    $status_i(t_{k+1}) = 1$;

Algorithm A.3 : Status 2: Motion at the endpoint of an obstacle

Data : $status_i$, Memory buffer

Result : $status_i(t_{k+1})$

while $status_i(t_k) == 2$ **do**
  if $N_i^\beta == 0$ ***and*** *Eq. (5.11)*$== 0$ **then**
    $status_i(t_{k+1}) = 0$;
  else if $N_i^\beta == 1$ ***and*** *Eq. (5.22)*$== 0$ **then**
    if *Eq. (5.23)*$== 1$ **then**
      $status_i(t_{k+1}) = 4$;
    else if *Eq. (5.23)*$== 0$ **then**
      $status_i(t_{k+1}) = 1$;
  else if $N_i^\beta == 2$ **then**
    $status_i(t_{k+1}) = 3$;
  else if $c_{reset} == 1$ **then**
    $status_i(t_{k+1}) = 0$;
  else
    $status_i(t_{k+1}) = 2$;

Algorithm A.4 : Status 3: Corner avoidance maneuver

Data : $status_i$, Memory buffer

Result : $status_i(t_{k+1})$

```
while status_i(t_k) == 3 do
    if N_i^β == 0 then
        status_i(t_{k+1}) = 0;
    else if N_i^β == 1 and Eq. (5.22)== 0 then
        if Eq. (5.23)== 1 then
            status_i(t_{k+1}) = 4;
        else if Eq. (5.23)== 0 then
            status_i(t_{k+1}) = 1;
    else if N_i^β == 2 then
        status_i(t_{k+1}) = 3;
    else
        status_i(t_{k+1}) = 3;
```

Algorithm A.5 : Status 4: Orientation phase

Data : $status_i$, Memory buffer

Result : $status_i(t_{k+1})$

```
while status_i(t_k) == 4 do
    if N_i^β == 0 then
        if Relevant orientation information is available then
            status_i(t_{k+1}) = 5;
        else if No relevant orientation information is available then
            status_i(t_{k+1}) = 0;
    else if N_i^β == 1 and Eq. (5.23)== 0 then
        status_i(t_{k+1}) = 1;
    else
        status_i(t_{k+1}) = 4;
```

Algorithm A.6 : Status 5: Tangential navigation based on received information

Data : $status_i$, Memory buffer

Result : $status_i(t_{k+1})$

```
while status_i(t_k) == 5 do
    if N_i^β == 0 then
        if Eq. (5.11)== 1 then
            status_i(t_{k+1}) = 0;
        else if No relevant information is available then
            status_i(t_{k+1}) = 0;
        else if Relevant information about an endpoint is available and Eq.
          (5.11)== 0 and Eq. (5.28)== 1 then
            status_i(t_{k+1}) = 2;
            Apply Eq. (5.33);
        else if Relevant information about an endpoint is available (Eq. 5.21) and
          Eq. (5.28)== 0 then
            status_i(t_{k+1}) = 5;
            Apply Eq. (5.30);
    else if N_i^β == 1 then
        if Eq. (5.22)== 0 and Eq. (5.23)== 1 then
            status_i(t_{k+1}) = 4;
        else if A virtual corner is determined (see Fig. 5.9) and Eq. (5.26== 1)
          then
            status_i(t_{k+1}) = 3;
        else if A virtual corner is determined (Fig. 5.9) and Eq. (5.26)== 0 then
            status_i(t_{k+1}) = 6;
        else
            status_i(t_{k+1}) = 1;
    else if N_i^β == 2 then
        status_i(t_{k+1}) = 3;
    else
        status_i(t_{k+1}) = 5;
```

Algorithm A.7 : Status 6: Waiting mode

Data : $status_i$, Memory buffer

Result : $status_i(t_{k+1})$

```
while status_i(t_k) == 6 do
    if Reorientation information (status = 3 or status = 4) is available then
        status_i(t_{k+1}) = 4;
        Apply the corresponding θ^c;
    else if Relevant information about an endpoint is available (Eq. (5.21)) then
        status_i(t_{k+1}) = 4;
        Adopt the direction of rotation of the circular motion for the next time step;
    else if Information with status = 1 is available then
        status_i(t_{k+1}) = 4;
        Apply the corresponding θ^c;
    else
        status_i(t_{k+1}) = 6;
```

List of Figures

List of Tables

Bibliography

[1] E. U. Acar and H. Choset. Multi-agent coverage control design with dynamic sensing regions. *The International Journal of Robotics*, 21(4):245–366, Apr. 2002.

[2] J. Adamy. *Nichtlineare Systeme und Regelungen*. Springer, 2018.

[3] A. A. Adepegba, S. Miah, and D. Spinello. Multi-agent area coverage control using reinforcement learning. In *The Twenty-Ninth International Flairs Conference*, 2016.

[4] G. Albi, M. Bongini, E. Cristiani, and D. Kalise. Invisible sparse control of self-organizing agents leaving unknown environments. *SIAM J. Appl. Math*, 2015.

[5] J. Alonso-Mora, T. Naegeli, R. Siegwart, and P. Beardsley. Collision avoidance for aerial vehicles in multi-agent scenarios. *Autonomous Robots*, 39(1):101–121, 2015.

[6] I. Amundson and X. D. Koutsoukos. A survey on localization for mobile wireless sensor networks. In *International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, pages 235–254. Springer, 2009.

[7] J. Andersson. *October 2013. A General-Purpose Software Framework for Dynamic Optimization*. PhD thesis, PhD thesis, Arenberg Doctoral School, KU Leuven, Department of Electrical Engineering.

[8] J. Andersson, J. Åkesson, and M. Diehl. Casadi: A symbolic package for automatic differentiation and optimal control. In *Recent advances in algorithmic differentiation*, pages 297–307. Springer, 2012.

[9] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl. Casadi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.

[10] L. T. Aschepkov, D. V. Dolgy, T. Kim, and R. P. Agarwal. *Optimal control*. Springer, 2016.

[11] M. Aureli and M. Porfiri. Coordination of self-propelled particles through external leadership. *EPL (Europhysics Letters)*, 92(4):40004, 2010.

[12] M. Aureli, F. Fiorilli, and M. Porfiri. Interactions between fish and robots: an experimental study. In *ASME 2010 Dynamic Systems and Control Conference*, pages 923–930. American Society of Mechanical Engineers Digital Collection, 2010.

[13] A. M. Ayala, S. B. Andersson, and C. Belta. Temporal logic motion planning in unknown environments. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5279–5284. IEEE, 2013.

[14] E. Ayvali, H. Salman, and H. Choset. Ergodic coverage in constrained environments using stochastic trajectory optimization. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept. 2017.

[15] P. Bak. *How nature works: the science of self-organized criticality*. Springer Science & Business Media, 1996.

[16] L. D. Beal, D. C. Hill, R. A. Martin, and J. D. Hedengren. Gekko optimization suite. *Processes*, 6(8):106, 2018.

[17] V. M. Becerra. Solving complex optimal control problems at no cost with psopt. In *2010 IEEE International Symposium on Computer-Aided Control System Design*, pages 1391–1396. IEEE, 2010.

[18] J. T. Betts. *Practical methods for optimal control and estimation using nonlinear programming*, volume 19. SIAM, 2010.

[19] J. Bisschop and A. Meeraus. On the development of a general algebraic modeling system in a strategic planning environment. In *Applications*, pages 1–29. Springer, 1982.

[20] J. A. Bondy, U. S. R. Murty, et al. *Graph theory with applications*, volume 290. Macmillan London, 1976.

[21] M. Bongini. *Sparse Optimal Control of Multiagent Systems*. Dissertation, Technical University of Munich, Munich, July 2016. URL `https://mediatum.ub.tum.de/doc/1303123/1303123.pdf`.

[22] M. Bongini and M. Fornasier. Sparse stabilization of dynamical systems driven by attraction and avoidance forces. *Networks & Heterogeneous Media*, 9:1–31, 2014. doi: 10.3934/nhm.2014.9.1.

[23] M. Bongini, M. Fornasier, F. Frcöhlich, and L. Haghverdi. Sparse control of force field dynamics. In *2014 7th International Conference on NETwork Games, COntrol and OPtimization (NetGCoop)*, pages 195–200. IEEE, 2014.

[24] J. Borenstein, Y. Koren, et al. The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288, 1991.

[25] A. Borzi and S. Wongkaew. Modeling and control through leadership of a refined flocking system. *Mathematical Models and Methods in Applied Sciences*, 25(2): 255–282, 2015.

[26] A. Borzi and S. Wongkaew. Modeling and control through leadership of a refined flocking system. *Mathematical Models and Methods in Applied Sciences*, 25(02): 255–282, 2015.

[27] A. S. Brandao, M. Sarcinelli-Filho, and R. Carelli. An analytical approach to avoid obstacles in mobile robot navigation. *International Journal of Advanced Robotic Systems*, 10(6), June 2013.

[28] A. S. Brandão, M. Sarcinelli-Filho, and R. Carelli. An analytical approach to avoid obstacles in mobile robot navigation. *International Journal of Advanced Robotic Systems*, 10(6):278, 2013.

[29] H. Broer, F. Takens, and B. Hasselblatt. *Handbook of dynamical systems*. Elsevier, 2010.

[30] S. Camazine, J.-L. Deneubourg, N. R. Franks, J. Sneyd, E. Bonabeau, and G. Theraula. *Self-organization in biological systems*, volume 7. Princeton University press, 2003.

[31] M. Caponigro, M. Fornasier, B. Piccoli, and E. Trélat. Sparse stabilization and control of the cucker-smale model. 2015. URL `https://hal.archives-ouvertes.fr/hal-00743792/document`.

[32] J. A. Castellanos and J. D. Tardos. *Mobile robot localization and map building: A multisensor fusion approach*. Springer Science & Business Media, 2012.

[33] H. Chaté, F. Ginelli, G. Grégoire, F. Peruani, and F. Raynaud. Modeling collective motion: variations on the vicsek model. *The European Physical Journal*, 64: 451—456, 2008.

[34] H. M. Choset, S. Hutchinson, K. M. Lynch, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun. *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[35] J.-M. Coron. *Control and nonlinearity*. Number 136. American Mathematical Soc., 2007.

[36] F. Cucker and J. G. Dong. Avoiding collisions in flocks. *IEEE Transactions on Automatic Control*, 55(5), May 2010.

[37] F. Cucker and J.-G. Dong. A conditional, collision-avoiding, model for swarming. *Discrete and Continuous Dynamical Systems*, 34(3):1009–1020, 2014.

[38] F. Cucker and S. Smale. On the mathematics of emergence. *Japanese Journal of Mathematics*, 2(1):197–227, 2007.

[39] F. Cucker and S. Smale. Emergent behavior in flocks. *IEEE Transactions on Automatic Control*, 52(5):852–862, May 2007.

[40] A. Dang, H. M. La, T. Nguyen, and J. Horn. Distributed formation control for autonomous robots in dynamic environments. May 2017. URL `https://arxiv.org/abs/1705.02017`.

[41] M. Defoort, T. Floquet, A. Kökösy, and W. Perruquetti. Sliding-mode formation control for cooperative autonomous mobile robots. *IEEE Transactions on Industrial Electronics*, 55(11), Nov. 2008.

[42] L. Doitsidis, S. Weiss, A. Renzaglia, M. W. Achtelik, E. Kosmatopoulos, R. Siegwart, and D. Scaramuzza. Optimal surveillance coverage for teams of micro aerial vehicles in gps-denied environments using onboard vision. *Autonomous Robots*, 33(1-2):173–188, 2012.

[43] A. Dorri, S. S. Kanhere, and R. Jurdak. Multi-agent systems: A survey. *IEEE Access*, 6:28573–28593, 2018.

[44] M. Elhoseny and A. E. Hassanien. Dynamic wireless sensor networks. In *Studies in Systems, Decision and Control*, volume 165. Springer, 2019.

[45] J. A. Fax and R. M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, 2004.

[46] O. Feinerman, I. Pinkoviezky, A. Gelblum, E. Fonio, and N. S. Gov. The physics of cooperative transport in groups of ants. *Nature Physics*, 14(7):683–693, 2018.

[47] R. Fourer, D. M. Gay, and B. W. Kernighan. A modeling language for mathematical programming. *Management Science*, 36(5):519–554, 1990.

[48] H. Fukushima, K. Kon, and F. Matsuno. Model predictive formation control using branch-and-bound compatible with collision avoidance problems. *IEEE Transactions on Robotics*, 29(5):1308–1317, 2013.

[49] M. Futterlieb, V. Cadenat, and T. Sentenac. A navigational framework combining visual servoing and spiral obstacle avoidance techniques. In *2014 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, volume 2, pages 57–64. IEEE, 2014.

[50] N. Ganganath, C.-T. Cheng, and K. T. Chi. Distributed anti-flocking control for mobile surveillance systems. In *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1726–1729. IEEE, 2015.

[51] N. Ganganath, C. Cheng, and C. K. Tse. Distributed antiflocking algorithms for dynamic coverage of mobile sensor networks. *IEEE Transactions on Industrial Informatics*, 12(15):1795–1805, May 2016.

[52] N. Ganganath, W. Yuan, , C. Cheng, T. Fernando, and H. H. C. Iu. Territorial marking for improved area coverage in anti-flocking-controlled mobile sensor networks. *IEEE International Symposium on Circuits and Systems (ISCAS)*, Jan. 2018.

[53] N. Ganganath, W. Yuan, C.-T. Cheng, T. Fernando, and H. H. Iu. Territorial marking for improved area coverage in anti-flocking-controlled mobile sensor networks. In *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4. IEEE, 2018.

[54] M. Guerra, D. Efimov, G. Zheng, and W. Perruquetti. Avoiding local minima in the potential field method using input-to-state stability. *Control Engineering Practice*, 55:174–184, 2016.

[55] S.-Y. Ha, T. Ha, and J.-H. Kim. Emergent behavior of a cucker-smale type particle model with nonlinear velocity couplings. *IEEE Transactions on Automatic Control*, 55(7):1679–1683, 2010.

[56] H. Haken. *Information and self-organization: A macroscopic approach to complex systems*. Springer Science & Business Media, 2006.

[57] J. Halloy, G. Sempo, G. Caprari, C. Rivault, M. Asadpour, F. Tâche, I. Saïd, V. Durier, S. Canonge, J. M. Amé, et al. Social integration of robots into groups of cockroaches to control self-organized choices. *Science*, 318(5853):1155–1158, 2007.

[58] J. Han and L. Wang. Nondestructive intervention to multi-agent systems through an intelligent agent. *PloS one*, 8(5), 2013.

[59] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[60] W. E. Hart, C. D. Laird, J.-P. Watson, D. L. Woodruff, G. A. Hackebeil, B. L. Nicholson, and J. D. Siirola. *Pyomo-optimization modeling in python*, volume 67. Springer, 2017.

[61] R. Hegselmann, U. Krause, et al. Opinion dynamics and bounded confidence models, analysis, and simulation. *Journal of Artificial Societies and Social Simulation*, 5(3), 2002.

[62] B. Houska, H. J. Ferreau, and M. Diehl. Acado toolkit—an open-source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods*, 32(3):298–312, 2011.

[63] G. Isaza, M. H. Mejia, L. F. Castillo, A. Morales, and N. Duque. Network management using multi-agents system. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, 1(3):49–54, 2012.

[64] S. Ivić, B. Crnković, and I. Mezić. Ergodicity-based cooperative multiagent area coverage via a potential field. *IEEE Transactions on Cybernetics*, 47(8):1983–1993, 2016.

[65] M. Ji, A. Muhammad, and M. Egerstedt. Leader-based multi-agent coordination: Controllability and optimal control. In *2006 American Control Conference*, pages 6–pp. IEEE, 2006.

[66] J. Kennedy, A. Chapman, and P. M. Dower. Generalized coverage control for time-varying density functions. *18th European Control Conference (ECC)*, June 2019.

[67] J. Kennedy, A. Chapman, and P. M. Dower. Generalized coverage control for time-varying density functions. In *2019 18th European Control Conference (ECC)*, pages 71–76. IEEE, 2019.

[68] H. K. Khalil and J. W. Grizzle. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, NJ, 2002.

[69] S. Knorn, Z. Chen, and R. H. Middleton. Overview: Collective control of multiagent systems. *IEEE Transactions on Control of Network Systems*, 3(4):334–347, 2015.

[70] A. Koval, S. S. Mansouri, and G. Nikolakopoulos. Online multi-agent based cooperative exploration and coverage in complex environment. *18th European Control Conference (ECC)*, June 2019.

[71] A. Krnjak, I. Draganjac, S. Bogdan, T. Petrović, D. Miklić, and Z. Kovačić. Decentralized control of free ranging agvs in warehouse environments. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2034–2041. IEEE, 2015.

[72] K. Kułakowski, P. Gawroński, and P. Gronek. The heider balance: A continuous approach. *International Journal of Modern Physics C*, 16(05):707–716, 2005.

[73] J. Lee, Y. Nam, S. Hong, and W. Cho. New potential functions with random force algorithms using potential field method. *Journal of Intelligent & Robotic Systems Science*, 66(3):302–319, 2012.

[74] Z. Lendek, R. Babuška, and B. De Schutter. Distributed kalman filtering for multiagent systems. In *2007 European Control Conference (ECC)*, pages 2193–2200. IEEE, 2007.

[75] N. E. Leonard and E. Fiorelli. Virtual leaders, artificial potentials and coordinated control of groups. In *Proceedings of the 40th IEEE Conference on Decision and Control*, volume 3, pages 2968–2973. IEEE, 2001.

[76] Z. Li, N. Hovakimyan, and D. Stipanović. Distributed multi-agent tracking and estimation with uncertain agent dynamics. In *Proceedings of the 2011 American Control Conference*, pages 2204–2209. IEEE, 2011.

[77] Y. Liu, H. Liu, Y. Tian, and C. Sun. Reinforcement learning based two-level control framework of uav swarm for cooperative persistent surveillance in an unknown urban area. *Aerospace Science and Technology*, page 105671, 2020.

[78] H. Long, Z. Qu, X. Fan, and S. Liu. Distributed extended kalman filter based on consensus filter for wireless sensor network. In *Proceedings of the 10th World Congress on Intelligent Control and Automation*, pages 4315–4319. IEEE, 2012.

[79] J. Lü, F. Chen, and G. Chen. Nonsmooth leader-following formation control of nonidentical multi-agent systems with directed communication topologies. *Automatica*, 64:112–120, 2016.

[80] C. Lum, J. Vagners, M. Vavrina, and J. Vian. Formation flight of swarms of autonomous vehicles in obstructed environments using vector field navigation. In *Proceedings of the International Conference on Unmanned Aircraft Systems*, 2012.

[81] J. Lunze. *Regelungstechnik 2: Mehrgrößensysteme, Digitale Regelung*. Springer-Verlag, 2010.

[82] J. Lunze. *Control theory of digitally networked dynamic systems*, volume 1. Springer, 2014.

[83] L. Ma, F. He, L. Wang, and Y. Yao. Multi-agent coverage control design with dynamic sensing regions. *Advances in Practical Multi-Agent Systems*, 325(3): 161–172, Aug. 2010.

[84] A. Macwan, J. Vilela, G. Nejat, and B. Benhabib. A multirobot path-planning strategy for autonomous wilderness search and rescue. *IEEE Transactions on Cybernetics*, 45(9):1784–1797, 2014.

[85] D. Mateo, N. Horsevad, V. Hassani, M. Chamanbaz, and R. Bouffanais. Optimal network topology for responsive collective behavior. *Science Advances*, 5(4): eaau0999, 2019.

[86] L. D. Mech. Alpha status, dominance, and division of labor in wolf packs. *Canadian Journal of Zoology*, 77:1196–1203, 1999.

[87] D. Mellinger, A. Kushleyev, and V. Kumar. Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams. In *2012 IEEE International Conference on Robotics and Automation*, pages 477–483. IEEE, 2012.

[88] T. Mercy, R. Van Parys, and G. Pipeleers. Spline-based motion planning for autonomous guided vehicles in a dynamic environment. *IEEE Transactions on Control Systems Technology*, 26(6):2182–2189, 2017.

[89] M. Mesbahi and M. Egerstedt. *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010.

[90] S. Miah, B. Nguyen, A. Bourque, and D. Spinello. Non-autonomous area coverage and coordination of a multi-agent system for harbor protection applications. *Proceedings of 8th International Conference on Simulation and Modeling Methodologies, Technologies and Applications*, 1(15):486–492, 2018.

[91] Y.-Q. Miao, A. Khamis, and M. S. Kamel. Applying anti-flocking model in mobile surveillance systems. *International Conference on Autonomous and Intelligent Systems (AIS)*, jun 2010.

[92] M. Milford and G. Wyeth. Hybrid robot control and slam for persistent navigation and mapping. *Robotics and Autonomous Systems*, 58(9):1096–1104, 2010.

[93] E. Natalizio and V. Loscrí. Controlled mobility in mobile sensor networks: advantages, issues and challenges. *Telecommunication Systems*, 52(4):2411–2418, 2013.

[94] M. T. Nguyen, L. Rodrigues, C. S. Maniu, and S. Olaru. Discretized optimal control approach for dynamic multi-agent decentralized coverage. In *2016 IEEE International Symposium on Intelligent Control (ISIC)*, pages 1–6. IEEE, 2016.

[95] Y. Nie, O. Faqir, and E. C. Kerrigan. Iclocs2: try this optimal control problem solver before you try the rest. In *2018 UKACC 12th International Conference on Control (CONTROL)*, pages 336–336. IEEE, 2018.

[96] N. Nigam, S. Bieniawski, I. Kroo, and J. Vian. Control of multiple uavs for persistent surveillance: algorithm and flight test results. *IEEE Transactions on Control Systems Technology*, 20(5):1236–1251, 2011.

[97] H. Nijmeijer and A. Van der Schaft. *Nonlinear dynamical control systems*, volume 175. Springer, 1990.

[98] M. Okamoto and M. Akella. Novel potential-function-based control scheme for non-holonomic multi-agent systems to prevent the local minimum problem. *International Journal of Systems Science*, 46(12):2150–2164, Sept. 2013.

[99] R. Olfati-Saber. Flocking with obstacle avoidance. 2003. Technical Report CIT-CDS 03-006.

[100] R. Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3), Mar. 2006.

[101] R. Olfati-Saber. Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401 – 420, Mar. 2006.

[102] R. Olfati-Saber. Distributed tracking for mobile sensor networks with information-driven mobility. In *2007 American Control Conference*, pages 4606–4612. IEEE, 2007.

[103] R. Olfati-Saber. Kalman-consensus filter: Optimality, stability, and performance. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 7036–7042. IEEE, 2009.

[104] R. Olfati-Saber and P. Jalalkamali. Collaborative target tracking using distributed kalman filtering on mobile sensor networks. In *Proceedings of the 2011 American Control Conference*, pages 1100–1105. IEEE, 2011.

[105] R. Olfati-Saber and R. M. Murray. Flocking with obstacle avoidance: Cooperation with limited communication in mobile networks. In *42nd IEEE International Conference on Decision and Control*, volume 2, pages 2022–2028. IEEE, 2003.

[106] B. G. Pachpatte. *Inequalities for differential and integral equations*, volume 197. Academic Press, San Diego, 1997.

[107] M. Papageorgiou, M. Leibold, and M. Buss. *Optimierung*, volume 4. Springer, 2015.

[108] S. Papatheodorou, A. Tzes, K. Giannousakis, and Y. Stergiopoulos. Distributed area coverage control with imprecise robot localization: Simulation and experimental studies. *International Journal of Advanced Robotic System*, 15(5), Sept. 2018. doi: 10.1177/1729881418797494.

[109] S. Papatheodorou, A. Tzes, K. Giannousakis, and Y. Stergiopoulos. Distributed area coverage control with imprecise robot localization: Simulation and experimental studies. *International Journal of Advanced Robotic Systems*, 15(5): 1729881418797494, 2018.

[110] A. A. Paranjape, S.-J. Chung, K. Kim, and D. H. Shim. Robotic herding of a flock of birds using an unmanned aerial vehicle. *IEEE Transactions on Robotics*, 34(4):901–915, 2018.

[111] M. A. Patterson and A. V. Rao. Gpops-ii: A matlab software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming. *ACM Transactions on Mathematical Software (TOMS)*, 41(1):1–37, 2014.

[112] L. Perea, P. Elosegui, and G. Gómez. Extension of the cucker-smale control law to space flight formations. *Journal of Guidance, Control, and Dynamics*, 32(2): 527–537, 2009.

[113] T. H. Pham, Y. Bestaoui, and S. Mammar. Aerial robot coverage path planning approach with concave obstacles in precision agriculture. In *2017 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*, pages 43–48. IEEE, 2017.

[114] M. Pohjola and H. Koivo. Measurement delay estimation for kalman filter in networked control systems. *IFAC Proceedings Volumes*, 41(2):4192–4197, 2008.

[115] A. V. Proskurnikov and A. L. Fradkov. Problems and methods of network control. *Automation and Remote Control*, 77(10):1711–1740, 2016.

[116] A. Prusiewicz. A multi-agent system for computer network security monitoring. In *KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications*, pages 842–849. Springer, 2008.

[117] J. Qin, C. Yu, and B. D. Anderson. On leaderless and leader-following consensus for interacting clusters of second-order multi-agent systems. *Automatica*, 74:214–221, 2016.

[118] Z. Qu. *Cooperative control of dynamical systems: applications to autonomous vehicles*. Springer Science & Business Media, 2009.

[119] V. Ramaswamy and J. R. Marden. A sensor coverage game with improved efficiency guarantees. In *2016 American Control Conference (ACC)*, pages 6399–6404. IEEE, 2016.

[120] A. V. Rao. A survey of numerical methods for optimal control. *Advances in the Astronautical Sciences*, 135(1):497–528, 2009.

[121] R. Reinhardt, A. Hoffmann, and T. Gerlach. *Nichtlineare Optimierung: Theorie, Numerik und Experimente*. Springer-Verlag, 2012.

[122] W. Ren, R. W. Beard, and E. M. Atkins. Information consensus in multivehicle cooperative control. *IEEE Control Systems Magazine*, 27(2):71–82, 2007.

[123] C. W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, pages 25–34, 1987.

[124] I. Ross. A matlab application package for solving optimal control problems. *Naval Postgraduate School, Monterey, CA, Tech. Rep. TR-711*, 2004.

[125] I. Saha, R. Ramaithitima, V. Kumar, G. J. Pappas, and S. A. Seshia. Automated composition of motion primitives for multi-robot systems from safe ltl specifications. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1525–1532. IEEE, 2014.

[126] H. Salman, E. Ayvali, and H. Choest. Multi-agent ergodic coverage with obstacle avoidance. *Proceedings of the 27th International Conference on Automated Planning and Scheduling (ICAPS)*, 2017.

[127] H. Salman, E. Ayvali, and H. Choset. Multi-agent ergodic coverage with obstacle avoidance. In *Twenty-Seventh International Conference on Automated Planning and Scheduling*, 2017.

[128] S. H. Semani and O. A. Basir. Semi-flocking algorithm for motion control of mobile sensors in large-scale surveillance systems. *IEEE Transactions on Cybernetics*, 45 (1), jan 2015.

[129] S. H. Semnani and O. A. Basir. Semi-flocking algorithm for motion control of mobile sensors in large-scale surveillance system. *IEEE Transactions on Cybernetics*, 45(1):129–137, Jan. 2015.

[130] J. Shen. Cucker–smale flocking under hierarchical leadership. *SIAM Journal on Applied Mathematics*, 68(3):694–719, 2007.

[131] S. L. Smith, M. Schwager, and D. Rus. Persistent robotic tasks: Monitoring and sweeping in changing environments. *IEEE Transactions on Robotics*, 28(2): 410–426, 2011.

[132] D. P. Stormont. Autonomous rescue robot swarms for first responders. *CIHSPS 2005. Proceedings of the 2005 IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety*, pages 151–157, March 2005.

[133] H. Su, X. Wang, and Z. Lin. Flocking of multi-agents with a virtual leader part i: with a minority of informed agents. In *2007 46th IEEE Conference on Decision and Control*, pages 2937–2942. IEEE, 2007.

[134] H. Su, X. Wang, and Z. Lin. Flocking of multi-agents with a virtual leader part ii: With a virtual leader of varying velocity. In *2007 46th IEEE Conference on Decision and Control*, pages 1429–1434. IEEE, 2007.

[135] Z. Sun. *Cooperative coordination and formation control for multi-agent systems*. Springer, 2018.

[136] P. N. Tu. *Dynamical systems: an introduction with applications in economics and biology*. Springer Science & Business Media, 2012.

[137] I. Ulrich and J. Borenstein. Vfh+: Reliable obstacle avoidance for fast mobile robots. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation*, volume 2, pages 1572–1577. IEEE, 1998.

[138] T. Vicsek and A. Zafeiris. Collective motion. *Physics reports*, 517(3-4):71–140, 2012.

[139] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet. Novel type of phase transition in a system of self-driven particles. *Phys. Rev. Lett.*, 75(6):1226–1229, Aug 1995. doi: 10.1103/PhysRevLett.75.1226.

[140] J. Vilca, L. Adouane, and Y. Mezouar. Reactive navigation of a mobile robot using elliptic trajectories and effective online obstacle detection. *Gyroscopy and Navigation*, 4(1):14–25, 2013.

[141] A. Wächter. *An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering*. PhD thesis, Carnegie Mellon University, 2002.

[142] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

[143] Z. Wang and D. Gu. Cooperative target tracking control of multiple robots. *IEEE Transactions on Industrial Electronics*, 59(8):3232–3240, 2011.

[144] K. Whitehead, C. H. Hugenholtz, S. Myshak, O. Brown, A. LeClair, A. Tamminga, T. E. Barchyn, B. Moorman, and B. Eaton. Remote sensing of the environment with small unmanned aircraft systems (uass), part 2: Scientific and commercial applications. *Journal of Unmanned Vehicle Systems*, pages 86–102, 2014.

[145] S. Wiggins. *Introduction to applied nonlinear dynamical systems and chaos*, volume 2. Springer Science & Business Media, 2003.

[146] S. Wongkaew. *On the control through leadership of multi-agent systems*. PhD thesis, Julius-Maximilians-Universität Würzburg, 2015.

[147] W. Yuan, N. Ganganath, C. Cheng, G. Qing, and F. C. M. Lau. Semi-flocking-controlled mobile sensor networks for dynamic area coverage and multiple target tracking. *IEEE Sensors Journal*, 18(21), Nov. 2018.

[148] W. Yuan, N. Ganganath, C.-T. Cheng, G. Qing, and F. C. Lau. Path planning for semi-flocking-controlled mobile sensor networks on mobility maps. In *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE, 2018.

[149] W. Yuan, N. Ganganath, C.-T. Cheng, S. Valaee, Q. Guo, and F. C. Lau. Energy-efficient semi-flocking control of mobile sensor networks on rough terrains. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 66(4):622–626, 2018.

[150] H.-T. Zhang, Z. Cheng, G. Chen, and C. Li. Model predictive flocking control for second-order multi-agent systems with input constraints. *IEEE Transactions on Circuits and Systems*, 62(6), June 2015.

[151] C. Zhu, L. Shu, T. Hara, L. Wang, S. Nishio, and L. T. Yang. A survey on communication and data management issues in mobile sensor networks. *Wireless Communications and Mobile Computing*, 14(1):19–36, 2014.

Publications by the Author

[152] E. Olcay and B. Lohmann. Extension of the cucker-dong flocking with a virtual leader and a reactive control law. In *2019 18th European Control Conference (ECC)*, pages 101–106. IEEE, 2019.

[153] E. Olcay, C. Dengler, and B. Lohmann. Data-driven system identification of an innovation community model. *16th IFAC Symposium on Information Control Problems in Manufacturing (INCOM)*, 51(11):1269–1274, 2018.

[154] E. Olcay, K. Gabrich, and B. Lohmann. Optimal control of a swarming multi-agent system through guidance of a leader-agent. In *8th IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys 2019)*, 2019.

[155] E. Olcay, B. Lohmann, and M. R. Akella. An information-driven algorithm in flocking systems for an improved obstacle avoidance. In *45th Annual Conference of the IEEE Industrial Electronics Society (IECON)*, volume 1, pages 298–304. IEEE, 2019.

[156] E. Olcay, C. Schöttl, A. Khalid, M. Zaggl, and B. Lohmann. An agent-based model of an online collaboration community by using fuzzy logic. In *9th IFAC Conference on Manufacturing Modelling, Management and Control (MIM)*, 2019.

[157] E. Olcay, J. Bodeit, and B. Lohmann. Sensor-based exploration of an unknown area with multiple mobile agents. In *21st IFAC World Congress*, 2020.

[158] E. Olcay, F. Schuhmann, and B. Lohmann. Collective navigation of a multi-robot system in an unknown environment. *Robotics and Autonomous Systems*, Vol. 132, 2020.

Supervised Student Theses

[159] S. Akdogan. Modeling and data-driven identification of community behaviors. Master's thesis, Technical University of Munich, 2017.

[160] S. Beger. Schwarmbildung von dynamischen Multiagentensystemen. Bachelor's thesis, Technical University of Munich, 2018.

[161] J. Bodeit. Kartengenerierung und Hindernisidentifikation durch mobile Sensornetzwerke. Bachelor's thesis, Technical University of Munich, 2019.

[162] K. Friedl. Navigation of swarming agents in an unknown area. Bachelor's thesis, Technical University of Munich, 2019.

[163] K. Gabrich. Optimale Steuerung von schwarmbildenden Multiagentensystemen durch Führung. Master's thesis, Technical University of Munich, 2018.

[164] M. Harder. Simulative Untersuchung eines Schwarmalgorithmus angewandt auf ein Multi-Roboter-System aus mobilen Kleinrobotern. Term paper, Technical University of Munich, 2018.

[165] T. Jukic. Modellierung und Regelung von selbst-organisierenden Agenten. Bachelor's thesis, Technical University of Munich, 2017.

[166] F. J. Schuhmann. Sensorbasierte Navigation von dynamischen Multiagentensystemen. Term paper, Technical University of Munich, 2019.

[167] E. van Halsema. Distributed tracking for a flocking system. Internship project, Eindhoven University of Technology, 2018.