EMERGENCE VIA CONSTRAINED OPTIMIZATION: ANALYSIS AND EXPERIMENTS WITH CONSTRAINT-DRIVEN FLOCKING

by

Logan E. Beaver

A dissertation submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Mechanical Engineering

Summer 2022

EMERGENCE VIA CONSTRAINED OPTIMIZATION: ANALYSIS AND EXPERIMENTS WITH CONSTRAINT-DRIVEN FLOCKING

by

Logan E. Beaver

Approved: _____

Ajay Prasad, Ph.D.
Chair of the Department of Mechanical Engineering

Approved: _____

Levi T. Thompson, Ph.D.
Dean of the College of Engineering

Approved: _____

Louis F. Rossi, Ph.D.
Vice Provost for Graduate and Professional Education and
Dean of the Graduate College

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Andreas A. Malikopoulos, Ph.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Herbert Tanner, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Sambeeta Das, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Rahul Mangharam, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Christopher M. Kroninger, M.S.
Member of dissertation committee

ACKNOWLEDGEMENTS

First, I would like to acknowledge and thank my advisor, Dr. Andreas Malikopoulos, for his support, energy, and enthusiasm over these past five years. Andreas's passion and drive for academia are contagious, and I am confident enough in the many lessons I've learned to strike out on my own. We both share a desire for disruptive ideas, and Andreas's vision has been an invaluable aid for shaping my research and perspective into its current form. I would also like to acknowledge the members of my dissertation committee. Dr. Bert Tanner's many suggestions have been invaluable for directing my research, and his technical questions have helped me to predict and work around technical issues. I appreciate the unique perspective of Dr. Sambeeta Das, and all her help and advice on planning my next career steps. I appreciate all the support of Dr. Rahul Mangharam at UPenn, visiting his research group has been a great experience, and these experiences have significantly affected the direction of my research. Finally, my meetings and summers with Chris Kroninger at ARL have been amazing experiences, and I have always been able to count on him to keep my research well-grounded.

Of course nothing is achieved in a bubble, and I would also like to acknowledge the many other wonderful people at the University of Delaware who have made this journey not only possible, but enjoyable as well. I am grateful to the department of mechanical engineering, and the Helwigs, for their support through the Helwig Fellowship. I also would like to thank the Graduate College for their support through the Graduate Scholar Award. I am especially grateful to Lisa Katzmire and Melissa Arenz for helping me with the thousands of logistical and procedural questions I've undoubtedly asked throughout my PhD.

I am grateful for the amazing friends I've made in the IDS Lab, and particularly those who broke ground with me in the first cohort: Adi Dave, Behdad Chalaki, Ben Remer, and Ishti Mahbub. I will never forget the (extremely) late nights working with Behdad and Ben in the scaled smart city, and how we would resolve hours of troubleshooting by changing the cars' batteries or re-calibrating VICON. It is unfortunate that Ben left us so soon (to go work in industry), and I'm sure he would have shared in our despair when we learned that Dunkin' Donuts lost their only night shift worker and started closing at 10pm! In particular, the winter of 2019 will be forever burned into my mind with equal parts pride and terror. Working around the clock with Behdad, Tommy, Michael, and Ray to triage our cars was possibly the hardest period of my entire life, and yet we persisted, and eventually turned that project into an Emmy-nominated news segment with NBC-10 Philadelphia. I also owe a debt of gratitude to Michael Dorothy, the "disruptive agent" at ARL. His guidance and technical discussions have become a cornerstone of my research over the past three years, and somehow, he always knows exactly what questions will completely break my models.

I would also like to thank all of my friends and family for their patience, support, and compassion during my time at UD. The pressures of graduate school are unique, and I will always be grateful for those who put up with my dynamic work and sleep schedules. I don't know what I did to deserve such a wonderful sister as Jade, but I will always be grateful to have her in my life. I also appreciate the efforts which all of my parents have made, they have supported me and sacrificed in their own ways. Although we have moved across the country, I have cherished my weekly Dungeons and Dragons nights with my old friends Tyler, Scott, Nate, and Mike. Adi helped me discover my love for board games, and I have spread that love like a virus to my dear friends Behdad and Shadi during our many weekend excursions. I am grateful to the MEGA board for all that they have done to improve the life of graduate students in our department and college; working with Nicole, Steve, Jack, Kleio, Kayla, and Kaleb was an absolute pleasure. Although I do not have time to name them all explicitly,

I am grateful to the many supportive friends and friendly acquaintances I have made throughout this journey. Completing a PhD is a marathon, and I have had countless good influences every step of the way.

Finally, and perhaps most significantly, I will always be grateful for the kindness and support of my girlfriend, Maryam Golbazi. In some ways it's incredible that we've both managed to survive our PhDs unscathed, and I am continually amazed at how much we've both grown these past few years. I have loved our many little trips around the east coast together, and I will never forget our micro-tour of Europe. Our daily work from home routine is the only thing that kept me sane during the pandemic, and I will always cherish the big adventures and small moments that we have shared.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

An automation revolution is looming, and the explosive growth of these sophisticated automated systems will only accelerate with the advent of industry 4.0, smart cities, and the internet of things. These systems are characterized as "complex," but this is a nebulous term. While there is no consensus on what exactly defines a complex system, it is generally understood that they share two characteristics: 1) complex systems consist of agents, which interact with each other and the environment using relatively simple rules, and 2) these interactions lead to *emergent* behavior, that is, patterns at length and time-scales larger than any of the individual agents. There are many examples of complexity and emergence in our everyday lives, such as flocks of geese, genetic networks, and economies. Given this context, the objective of this dissertation is twofold: to provide short-term benefits for the design and analysis of robotic multi-agent systems, and to enhance our general understanding of complex systems and their emergent behaviors.

This dissertation adopts the view of emergence proposed by English cybernetician W. Ross Ashby in 1962. Ashby argued that large-scale patterns emerge when a multi-agent system is driven into an equilibrium state. This definition of emergence is particularly useful, as it provides a means to rigorously interrogate complex systems using techniques from optimization and dynamical systems. Thus, this dissertation proposes a constraint-driven optimization approach to generate emergence in multi-agent systems, as well as a new technique to solve constrained optimal control problems quickly. **The first contribution** of this dissertation demonstrates that moving agents to predefined goals, e.g., for a drone light show, is an example of a "medium complexity" emergence problem—one that is challenging to solve by hand but is well

within the capabilities of a digital computer. This result is extended in **the second contribution**, a constraint-driven framework to bring groups of autonomous vehicles into energy-minimizing configuration; this includes highway platooning, aerial V formations, and predator avoidance. This constraint-driven approach has many benefits: 1) it is straightforward to interpret why an agent takes a particular action, 2) the system-level behavior can be guaranteed by examining pairwise agent states, 3) the approach is rigorous and data driven, and 4) agents can arbitrarily enter and leave the system, subject to safety constraints. Finally, **the third contribution** is a novel technique to generate optimal trajectories in real time. A naïve Matlab implementation of the proposed algorithm outperforms two open-source state of the art solvers (OpenOCL, which uses a C++ CasADi implementation, and ICLOCS2) by every metric.

In terms of broader impacts, this dissertation represents another union between complex systems and control theory. The open questions raised by Ashby can be interrogated directly through the engineering design process: emergent behavior is guaranteed when a multi-agent system reaches equilibrium, the "complexity" of a system corresponds to the number of equilibrium points, and the "goodness" of a particular emergent behavior corresponds to how well it achieves the desired outcome. It turns out that dynamical systems, stability, and optimization are a natural language for rigorous discussion of complex systems and emergent behavior.

Chapter 1

ENGINEERING COMPLEX SYSTEMS

> 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...
>
> Aristotle
> Metaphysics (350 BCE)

We are living in the most interconnected and complex society ever witnessed by humanity. Every aspect of our lives—finance, health, transportation, communication—is subject to a torrent of global influences. Even the natural environment, the largest complex system on earth, is being altered at unprecedented rates by human activity. Now, more than ever, it is critical that we develop tools to help us understand the behavior of these complex and interconnected systems. At the same time, an automation revolution is looming; the explosive growth of sophisticated automated systems will only accelerate with the advent of industry 4.0, smart cities, and autonomy as a service. A recent analysis suggests that, in the US, the robotics market has grown by over 10% every year, and has seen less than 10% penetration so far; this is particularly true in industrial settings, where collaborative robots have had a significant impact [1]. Furthermore, self-driving vehicles have been operating on our roads for over ten years, and new research continues to push the frontiers of automated ground and aerial vehicles for new modes of mobility. The objective of this dissertation, then, is to provide short-term benefits in the design and analysis of multi-robot systems, and long-term benefits by increasing our understanding of complex systems and their emergent behaviors.

Complex systems are an intriguing topic which have been the subject of study since at least the time of Aristotle. Yet, the term "complex system" is nebulous; there is no general consensus on what it means for a system to be complex. However, it is understood that complex systems share two characteristics. All complex systems consist of agents, which interact with each other and the environment using relatively simple rules. These interactions lead to *emergent* behavior in complex systems, that is, unexpected patterns at length and time-scales that are much larger than the agents and their local interactions. There are many examples of complexity and emergence in our everyday lives; flocks of geese, schools of fish, genetic regulatory networks, and financial systems.

Recent work on emergence in engineered systems uses a discussion of levels [2]. A complex system is said to exhibit emergent behavior if the large-scale patterns emerge over much greater length and time scales than the local interactions between agents. The 'high level' phenomena may also appear to follow a distinct set of rules compared to the 'low level' agent behaviors; computer memory is an illustrative example of this concept. In this case, the 'low level' is a collection of physical transistors, which operate under the laws of physics. The transistors may be assembled into a 'high level' memory chip; these chips operate on length and time scales that are orders of magnitude larger than the transistors. At a higher level still is a digital processor, where the collective states of many memory chips may be read from and written to thousands of times per second to execute an instruction set. Whether a digital computer counts as an emergent system, or if they are simply a result of increasing layers of abstraction, is a hotly debated question [3].

This dissertation adopts a cybernetics (Greek *kybernetes* 'to steer' + Latin *-ics* 'study') emergence paradigm, particularly the interpretation proposed by the English cybernetician W. Ross Ashby [4]. Cyberneticians in general, and Ashby in particular, are interested in the behavior of dynamical systems (or "machines") subject to feedback loops. Ashby proposes that any systems satisfying the following three conditions

will exhibit emergent behavior: 1) **the system consists of multiple interacting agents**, 2) **each agent has a local feedback control law**, and 3) **the system has equilibria**. In this case, a formal definition of emergence follows logically; wherever the agents are initialized, the system will naturally converge to the subset of stable equilibria—this is exactly the emergence of organization from low-level interactions! While our understanding of nonlinear dynamical systems has expanded significantly since Ashby's time [5], these same principles still apply if we restrict our analysis to attractive equilibria and their basins of attraction. This is also the approach taken the dynamical systems literature, where emergence has been modeled using the equilibrium states of chaotic attractors [6].

Ashby's definition of emergence is both pragmatic and elegant. First, convergence to an equilibrium point is straightforward to quantify, and the field of dynamical systems is rich with results [5]. The convergence perspective also agrees with the notion that many natural phenomena correspond to maxima and minima[1]; the free energy principle [7], the principle of least action, and Gibbs free energy are all examples of systems moving toward energy-minimizing states in the natural world. Ashby's definition of emergence also implies following properties:

- most emergent behaviors are "bad,"
- good emergent behavior must be sought out, and
- what is meant by "good" must be clearly defined in every case.

In this sense emergence is not "strong" or "weak," nor does it follow some set of indecipherable metaphysical laws [3]. Instead, the challenge with emergent systems is a matter of scale; some systems (such as the digital computer) have trivially small state and action spaces, while others (such as biological systems) have an incredibly

[1] Leonhard Euler claimed that "Nothing takes place in the world whose meaning is not that of some maximum or minimum."

rich subspace of equilibrium points at multiple scales. Thus, one benefit from viewing emergence as a problem of decentralized control, is the creation of "medium complexity" problems. In these instances, the size of the state space and the richness of the equilibrium points is too large to solve analytically, but they are still tractable to analyze with the use of a digital computer.

This dissertation proposes a constrained optimization approach to generating emergence in multi-agent systems which is rooted in Ashby's definition of emergence. Chapters 2 and 3 address problems in multi-agent formation control and flocking, which are the simplest emergent behaviors with practical applications in the controls community [8, 9]. In particular, Chapter 3 looks at energy-minimizing strategies for fleets of autonomous ground and aerial vehicles; thus, the emergent behavior of the system coincides with an energy-saving configuration of the agents. An optimization approach to emergence also agrees with the properties of complex systems laid out by Scott Page [10]; namely a system of agents which are *interdependent* (the agents are decentralized), *connected* (the agents communicate and interact), *adaptable* (the agents act using optimization), and *diverse* (Section 3.2 provides evidence that heterogeneity improves performance). The next section presents background material, literature gaps, and motivation for multi-agent flocking research. Much of the following material was previously published in a review article,

- **L.E. Beaver** and A.A. Malikopoulos, "An Overview on Optimal Flocking," in *Annual Reviews in Control*, vol. 51, pp. 88–99.

1.1 Literature Overview

The most common complex system which is analyzed in the swarm robotics literature is flocking. Generating emergent flocking behavior has been of particular interest since Reynolds proposed three heuristic rules for multi-agent flocking in computer animation [11]. Robotic flocking has been proposed in several applications including mobile sensing networks, coordinated delivery, reconnaissance, and surveillance [8].

With the significant advances in computational power in recent decades, the control of robotic swarm systems has attracted considerable attention due to their adaptability, scalability, and robustness to individual failure [12]. However, constructing a swarm with a large number of robots imposes significant cost constraints on each individual robot. Thus, any real robot swarm must consist of individual robots with limited sensing, communication, actuation, memory, and computational abilities. To achieve robotic flocking in a physical swarm, we must develop and employ energy-optimal approaches to flocking under these strict cost constraints.

One significant problem throughout the literature is the use of the term "flocking" to describe very different modes of aggregate motion. The biology literature emphasizes this point [13], where the distinction of line flocking (e.g., geese) and cluster flocking (e.g., starlings) is necessary. To this end, we believe it is helpful to partition the engineered flocking literature into cluster and line flocking. As with natural systems, these modes of flocking have vastly different applications and implementations; unlike biological systems, the behavior of engineering systems is limited only by the creativity of the designer. Our proposed flocking taxonomy is shown in Fig. 1.1. As a general rule, cluster flocking denotes behavior that emerges in service to a system-level objective, e.g., forming an α-lattice or tracking a moving target. On the other hand, line flocking occurs when agents exploit environmental interactions to minimize their energy consumption, e.g., aerial upwash, drafting, and underwater currents.

A vast amount of literature exists that seeks to achieve cluster flocking under Reynolds three flocking rules, (1) collision avoidance, (2) velocity matching, and (3) flock centering [11]. In almost all cluster flocking applications, each individual agent may only observe its direct neighbors at any particular instant in time, and therefore each agent may only make partial observations of the total system state. Thus, many of the approaches in the optimal cluster flocking literature rely on repeatedly simulating the system, evaluating the global cost, and updating the agents' control policies. This

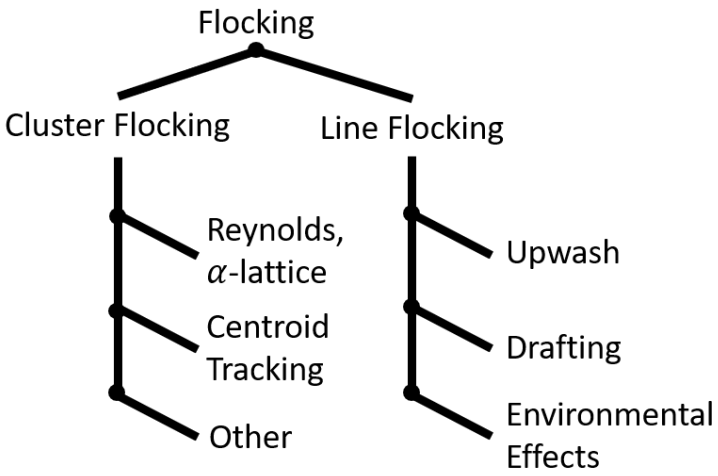

Figure 1.1: Our proposed flocking classification scheme for cluster and line flocking.

makes artificial potential fields a particularly attractive solution, as they can be parameterized in a finite number of variables and are known to guarantee stable flocking under very reasonable assumptions [8, 14]. However, it is well-known that potential fields have several major drawbacks [15], including steady oscillations and exacerbating deadlock in crowded environments. Despite these drawbacks, potential fields remain the most popular and well-developed solution; their performance was demonstrated in an outdoor experiment [16], where the potential fields were parameterized by 11 optimization variables that were selected using a genetic algorithm paired with a sophisticated simulation.

There are many other optimization-based approaches that employ Reynolds flocking rules. One early technique uses a learning-based approach to select motion primitives for individual agents [17]. Other work [18] formulates flocking as a dynamic program to generate optimal trajectories for a swarm of quadrotors, where the agents minimize their deviation from Reynolds flocking rules while tracking a global reference position. Metaheuristic algorithms, including Pigeon-inspired optimization [19] and particle swarm optimization [20], design systems that optimally follow Reynolds flocking rules. The effect of control input constraints for an optimal flocking controller

is studied in [21], and [22] considers both state and control constraints. An analysis by [23] presents a mechanism for flocking agents to estimate their neighbors' future trajectories, and this predictive device to achieves Reynolds flocking under a fully connected communication topology [24]. This work was extended to the decentralized information case [25] and validated experimentally with outdoor flight tests [26]. It has also been demonstrated that flocking behavior emerges when a system of agents optimally track a moving virtual reference point [27]. An article by Quintero et al. [28] applies dynamic programming to generate a swarm of aerial vehicles that circle over a moving ground target. This brief snapshot demonstrates the richness of the flocking literature, while also emphasizing that constraint-driven techniques have been completely neglected outside of one recent conference paper [29].

The problem of line flocking has received significantly less attention. To minimize energy consumption during flight, the most straightforward method to achieve line flocking may be to generate an optimal set of formation points based on the drag, wake, and upwash characteristics of each agent. This effectively transforms the line flocking problem into a formation reconfiguration problem, where each agent must assign itself to a unique goal and reach it within some fixed time, as is the case in [30]. However, a formation reconfiguration approach generally requires the formation to be computed offline and does not necessarily consider differences between individual agents (e.g., age, weight, size, and efficiency) or environmental disturbances. An alternative approach was recently taken to achieve the emergence of a peloton formation in a team of ground vehicles [31]. The authors apply CFD analysis to estimate the drag force imposed on each agent as a function of their separating distance; the agents then use this information to minimize their expended energy via gradient descent. The authors confirm the lack of research in this area, stating that *"Previous work on the application of cycling peloton behavior is nonexistent since this is the first attempt to do such."*

Line flocking with aerial vehicles has received some attention; an early approach

to capture line flocking behavior in a robotic system with model predictive control [32] uses a large multi-objective cost function. An analysis of the effect of upwash on energy consumption in fixed-wing UAVs was recently presented [33]. The authors demonstrate that the front and tail agents in a V formation have the highest rate of energy consumption in the flock, and thus the lead and tail agents are the limiting factor in total travel distance. The authors propose a load balancing algorithm based on a root-selection protocol, where the highest-energy agents replace the lead and tail agents periodically. The authors then demonstrate, in simulation, that periodic replacement of the lead and tail agents significantly increases the total distance travelled by the flock.

Limited research has also been done to explore the effect the environment, through air and water currents, on the energy consumption of flocks. This has primarily been addressed by analyzing how airplanes in the north Atlantic might rendezvous to save energy by flocking and exploiting atmospheric currents [34–36]. Energy-optimal flocking in the presence of strong background flows was investigated for underwater vehicles in the presence of ocean currents [37]. In this work, the authors assume that the background flow dominates the energy consumption of the agents, and therefore a tight cluster of agents closely approximates the energy-optimal trajectory traced out by the center of the flock. Thus, this problem can be reduced to finding the energy-optimal path for the center of the flock, then using another algorithm e.g., Reynolds flocking, to track the virtual reference point. Thus, while line flocking has great potential to increase the efficiency of multi-agent systems, there have been few results published on the topic—and to the best of our knowledge, Chapter 3 contains the first description of a constraint-driven line flocking system.

With the background material of this dissertation in place, and several literature gaps identified, the next section discusses the contributions of each chapter in detail.

1.2 Contributions

The work I have produced throughout my PhD can be roughly divided into four categories: 1) decentralized control in the context of formation generation and flocking, 2) the development of new techniques for constraint-driven control of multi-agent systems, 3) a new methods to generate optimal trajectories in real time, and 4) building and programming our 1:25 scale testbed and fleet of 35 connected and automated vehicles. My contributions to the first three categories make up Chapters 2–4 of this dissertation, and their summaries follow. I have not explicitly included my contributions to the scaled testbed in this dissertation, as that work is primarily concerned with robotic hardware and software architecture. For details on our scaled testbed, see the following articles:

- [38] **L.E. Beaver**, B. Chalaki, A.M. Mahbub, L. Zhao, R. Zayas, A.A. Malikopoulos, "Demonstration of a Time-Efficient Mobility System Using a Scaled Smart City" in *Vehicle System Dynamics*, vol. 58, issue 5, pp. 787–804, 2020.
- [39] B. Chalaki, **L.E. Beaver**, A.M. Mahbub, H. Bang, and A.A. Malikopoulos, "A Research and Educational Robotic Testbed for Real-Time Control of Emerging Mobility Systems: From Theory to Scaled Experiments," in *IEEE Control Systems magazine* (in press), 2022.

Goal Assignment as an Emergent Behavior

A common requirement for multi-agent systems is to move into a desired formation, which has become especially common with drone light shows in the entertainment industry. However, due to cost constraints imposed on individual agents in a swarm, e.g., limited computation capabilities, battery capacity, and sensing abilities, any efficient control approach must take into account energy consumption. The task of moving in a specified formation has been explored in the literature [12, 40, 41], however, achieving formations with minimum energy consumption during operation has not yet been thoroughly investigated. In Chapter 2, we consider the problem of moving a collection

of agents into a predefined formation with the following contributions: 1) an equivalence between the assignment problem and emergence as defined by Ashby, 2) a new approach to decentralized goal assignment, 3) a guarantee that all agents converge to a unique goal in finite time under reasonable assumptions about the initial conditions and goal states, and 4) an optimization-based approach to goal selection, which minimizes the expected energy consumption of the individual agents—particularly in the case of dynamic moving goals. The contributions of Chapter 2 were previously published in the following articles:

- [42] **L.E. Beaver** and A.A. Malikopoulos, "A Decentralized Control Framework for Energy-Optimal Goal Assignment and Trajectory Generation" in *58th Conference on Decision and Control*, pp. 879–884, 2019.
- [43] **L.E. Beaver** and A.A. Malikopoulos, "An Energy-Optimal Framework for Assignment and Trajectory Generation in Teams of Autonomous Agents" in *Systems & Control Letters*, vol. 138, April 2020.
- [44] H. Bang, **L.E. Beaver**, and A.A. Malikopoulos, "Energy-Optimal Goal Assignment of Multi-Agent Systems with Goal Trajectories in Polynomials" in *29th Mediterranean Conference on Control and Automation*, pp 1228–1233, 2021.

Constraint-Driven Control of Multi-Agent Systems

A recent push in constraint-driven control has brought the idea of long-duration autonomy to the forefront of multi-agent systems research [45]. For long-duration autonomy tasks, robots are left to interact with their environment on timescales significantly longer than what can be achieved in a laboratory setting. These approaches necessarily emphasize safe energy-minimizing control policies for the agents, whose behaviors are driven by interactions with the environment. Chapter 3 expands on the state of the art in constraint-driven control in four ways: 1) an original framework, and several examples, for constraint-driven control in multi-agent systems, 2) sufficient conditions on the local states of ground vehicles that guarantee the emergence of platooning behavior, 3) the first article, to the best of our knowledge, demonstrating

how the physics of fixed-wing UAVs can be exploited for emergent energy-saving V formations, and 4) a novel switching system architecture that preserves guarantees on agent behavior recursive feasibility. The contributions of Chapter 3 were published in the following articles:

- [46] **L.E. Beaver** and A.A. Malikopoulos, "Constraint-Driven Optimal Control of Multiagent Systems: A Highway Platooning Case Study" in *IEEE Control Systems Letters*, vol. 6, pp. 1754–1759, 2022.
- [47] **L.E. Beaver** and A.A. Malikopoulos, "Constraint-Driven Optimal Control for Emergent Swarming and Predator Avoidance" *(in review)*, arxiv 2203.11057.
- [48] **L.E. Beaver**, C. Kroninger, and A.A. Malikopoulos, "A Constraint-Driven Approach to Line Flocking: The V Formation as an Energy-Saving Strategy" in *(in preparation)*.

Real-Time Optimal Control

As cyber-physical systems achieve higher autonomy levels, they will be forced into complicated interactions with each other and the surrounding environment. These systems must be able to react quickly to these disturbances and rapidly re-plan efficient trajectories. For continuous-time systems, the optimality conditions take the form of an ordinary differential equation; unfortunately, these equations are generally unstable and take significant computational power and time to solve [49]. This challenge has been partially resolved for differentially flat systems [50], where a bijective mapping exists between the nonlinear system dynamics and an equivalent system with integrator dynamics in the so-called "flat" space. In the overwhelming majority of cases, a designer selects a set of basis functions [51], e.g., polynomials, Fourrier series, or splines, to generate their optimal trajectory in the flat space; this approach can quickly yield near-optimal trajectories for the system. In contrast, Chapter 4 presents an original technique that separates the optimality conditions into two differential equations, which can be solved independently using standard numerical methods. To the best of our knowledge, the only corpus of work taking a similar approach was F. Chaplais and

N. Petit [52, 53], who proved that the optimality conditions ought to be separable in a special case. However, the authors shifted their focus to solving constrained optimization problems with saturation functions [54] and kernels [55], and a general separation result has not been presented in the literature to date. Thus, the main contributions of Chapter 4 are as follows: 1) an differential equation that is independent of the costates (and thus numerically stable), which describes the optimal evolution of the state trajectory and can algoritmically generate "optimal motion primitives," 2) an equivalent set of optimality conditions that describes how the system reacts to constraint activations (i.e., transitioning between motion primitives), 3) a numerical example of energy-optimal obstacle avoidance that significantly outperforms two open source optimal control libraries by every metric, OpenOCL [56] (a Matlab package backed by the C++ CasADi library) and ICLOCS2 [57], and 4) experimental validation of an unconstrained motion primitive being used to coordinate a system of connected and automated vehicles at an unsignalized roundabout to eliminate stop-and-go driving. The contributions of Chapter 4 were published in the following peer-reviewed articles:

- [58] B. Chalaki, **L.E. Beaver** and A.A. Malikopoulos, "Experimental Validation of a Real-Time Optimal Controller for Coordination of CAVs in a Multi-Lane Roundabout" 2020 Intelligent Vehicles Symposium, pp. 775–780, 2020.
- [59] **L.E. Beaver**, M. Dorothy, C. Kroninger, and A.A. Malikopoulos, "Energy-Optimal Motion Planning for Agents: Barycentric Motion and Collision Avoidance Constraints," in *2021 American Control Conference*, pp. 1040–1045, 2021.
- [60] **L.E. Beaver** and A.A. Malikopoulos, "Optimal Control of Differentially Flat Systems is Surprisingly Simple" *(in review)*, arXiv 2013.03339.

Chapter 2

GOAL ASSIGNMENT AS AN EMERGENT BEHAVIOR

> Today we cannot see that the water flow equations contain such things as the barber pole structure of turbulence that one sees between rotating cylinders. We cannot see whether Schrödinger's equation contains frogs, musical composers, or morality——or whether it does not.
>
> Richard Feynman
> Lectures on Physics (1970)

In this chapter I present an original solution, based on [42–44], to the formation reconfiguration problem, i.e., how to move a collection of agents into a desired formation. This problem is an excellent candidate for Ashby's so-called "medium complexity" problems—systems that are too complex to easily solved analytically, but simple enough for a modern computer to simulate. The complexity comes from the assignment of N agents to $M \geq N$ goals. This system satisfies the requirements laid forth by Ashby [4] for a "medium complexity" emergence problem if one imagines a global $N \times M$assignment matrix that is constructed from the collective decisions of each agent:

1. **The system consists of multiple interacting agents.** Each row of the assignment matrix is updated by the corresponding agent.

2. **Each agent has a local feedback control law.** Agents update their assigned goal based on the local state of the system.

3. **The system has attractive equilibria.** There are ${}_NP_M$ possible assignments that assign each agent to a unique goal, and the system converges to one of them.

4. **The state and action space must be sufficiently large.** In this case, the assignment matrix may initially take ${}_NC_M$ possible values.

Thus, the assignment matrix precisely fits Ashby's definition of a non-trivial emergent system. Conveniently, the emergence of a feasible assignment in this system coincides with the convergence of agents to goals; thus, the emergent behavior is also useful from an engineering perspective. We can quantify the utility of any equilibrium by examining relevant notions of cost, e.g., energy consumption, arrival time, or maximum jerk. I present this analysis next, which has been edited from the original manuscript to fit within this dissertation.

2.1 An Energy-Optimal Framework for Assignment and Trajectory Generation in Teams of Autonomous Agents

Swarms are typical complex system which have attracted considerable attention in many applications, e.g., transportation, exploration, construction, surveillance, and manufacturing. As discussed by Oh et al. [40], swarms are especially attractive due to their natural parallelization and general adaptability. One of the typical multiagent applications is creating desired formations. However, due to cost constraints on any real swarm of autonomous agents, e.g., limited computation capabilities, battery capacity, and sensing capabilities, any efficient control approach needs to take into account the energy consumption of each agent. Moving agents into a desired formation has been explored previously; however, creating this formation while minimizing energy consumption is an open problem

Previous work [61] solves the assignment problem by imposing a priority order on the agents using a centralized planner, with higher priority agents acting first. In general, finding an optimal ordering is NP-Hard, and an optimal ordering is not

guaranteed to exist [62]. To reduce the complexity of ordering agents, much work has been done to decentralize the ordering problem, including applying discretized path-based heuristics [63] and reinforcement learning [64]. In contrast, the proposed approach introduces a decentralized method of dynamically ordering agents that is path independent and relies only on information directly observable by each agent.

Our approach only requires agents to make partial observations of the entire system. This may lead to performance degradation relative to a centralized controller with global knowledge, however, this is a fundamental feature of decentralized control problems [65]. Other efforts have attempted to circumvent this issue with information sharing, either directly [66, 67] or through decentralized auctioning [68]. However, these approaches tend to require knowledge of the global communication graph, require a significant number of communication rounds to reach a decision, or both. In contrast, our approach embraces the partial observability of the system and exploits it to reduce the computational load on each individual agent.

The main contributions of this work are: (1) a decentralized set of *interaction dynamics*, which impose a priority order on agents in a decentralized manner, (2) an assignment algorithm that exploits the unconstrained optimal trajectory of the agents, and (3) guarantees on the stability of the proposed control policy.

In Section 2.1.1, we formulate the decentralized optimal control problem, and we decompose it into the coupled assignment and trajectory generation subproblems. In Section 2.1.2, we present the conditions which guarantee system convergence along with the assignment problem. Then, in Section 2.1.3, we prove that these conditions are satisfied by our framework and solve the optimal trajectory generation problem. Finally, in Section 2.1.4, we present a series of MATLAB simulations to show the performance of the algorithm, and we contextualize the results within the dissertation in Section 2.2.

2.1.1 Problem Formulation

We consider a swarm of $N \in \mathbb{N}$ autonomous agents indexed by the set $\mathcal{A} = \{1, \ldots, N\}$. Our objective is to design a decentralized control framework to move the N agents into $M \in \mathbb{N}$ goal positions, indexed by the set $\mathcal{F} = \{1, \ldots M\}$. We consider the case where $N \leq M$, i.e., no redundant agents are brought to fill the formation. This requirement can be relaxed by defining a behavior for excess agents, such as idling [69]. Each agent $i \in \mathcal{A}$ follows double-integrator dynamics,

$$\dot{\mathbf{p}}_i(t) = \mathbf{v}_i(t), \tag{2.1}$$

$$\dot{\mathbf{v}}_i(t) = \mathbf{u}_i(t), \tag{2.2}$$

where $\mathbf{p}_i(t) \in \mathbb{R}^2$ and $\mathbf{v}_i(t) \in \mathbb{R}^2$ are the time-varying position and velocity vectors respectively, and $\mathbf{u}_i(t) \in \mathbb{R}^2$ is the control input (acceleration/deceleration) over time $t \in [t_i^0, t_i^f] \subset \mathbb{R}$, where t_i^0 and t_i^f are the initial and terminal time for agent i. Additionally, each agent's control input and velocity are bounded by

$$||\mathbf{v}_i(t)|| \leq v_{\max}, \tag{2.3}$$

$$||\mathbf{u}_i(t)|| \leq u_{\max}, \tag{2.4}$$

where $||\cdot||$ is the Euclidean norm. Thus, the state of each agent $i \in \mathcal{A}$ is given by the time-varying vector

$$\mathbf{x}_i(t) = \begin{bmatrix} \mathbf{p}_i(t) \\ \mathbf{v}_i(t) \end{bmatrix}, \tag{2.5}$$

and we denote the global (system) state as

$$\mathbf{x}(t) = \begin{bmatrix} \mathbf{x}_1(t) \\ \ldots \\ \mathbf{x}_N(t) \end{bmatrix}. \tag{2.6}$$

The energy consumption of any agent $i \in \mathcal{A}$ is given by

$$\dot{E}_i(t) = \frac{1}{2}||\mathbf{u}_i(t)||^2. \tag{2.7}$$

We select the L^2 norm of the control input as our energy model since, in general, acceleration/deceleration requires more energy than applying no control input. Therefore, we expect that minimizing the acceleration/deceleration of each agent will yield a proportional reduction in energy consumption. Our objective is to develop a decentralized framework for the N agents to optimally, in terms of energy, create any desired formation of M points while avoiding collisions between agents.

Definition 1. The *desired formation* is the set of time-varying vectors $\mathcal{G} = \{\mathbf{p}_j^*(t) \in \mathbb{R}^2 \mid j \in \mathcal{F}\}$. The set $\mathcal{G}$ can be prescribed offline, i.e., by a human designer, or online by a high-level planner.

In this framework, the agents are cooperative and capable of communication within a neighborhood, which we define next.

Definition 2. The *neighborhood* of agent $i \in \mathcal{A}$ is the time-varying set

$$\mathcal{N}_i(t) = \Big\{ j \in \mathcal{A} \;\Big|\; \big|\big|\mathbf{p}_i(t) - \mathbf{p}_j(t)\big|\big| \leq h \Big\},$$

where $h \in \mathbb{R}$ is the sensing and communication horizon of each agent.

Agent $i \in \mathcal{A}$ is also able to measure the relative position of any neighboring agent $j \in \mathcal{N}_i(t)$, i.e., agent i makes partial observations of the global state. We denote the relative position between two agents i and j by the vector

$$\mathbf{s}_{ij}(t) = \mathbf{p}_j(t) - \mathbf{p}_i(t). \tag{2.8}$$

Each agent $i \in \mathcal{A}$ occupies a closed disk of radius R; hence, to guarantee safety for agent i we impose the following constraints for all agents $i \in \mathcal{A}, \quad j \in \mathcal{N}_i(t), \; j \neq i$,

$$||\mathbf{s}_{ij}(t)|| \geq 2R, \quad \forall t \in [t_i^0, t_i^f], \tag{2.9}$$

$$h >> 2R. \tag{2.10}$$

Condition (2.9) is our safety constraint, which ensures that no two agents collide. We also impose the strict form of (2.9) pairwise to all goals in the desired formation, $\mathcal{G}$. Equation (2.10) is a system-level constraint which ensures agents are able to detect each other prior to a collision.

In our modeling framework we impose the following assumptions:

Assumption 1. The state $\mathbf{x}_i(t)$ for each agent $i \in \mathcal{A}$ is perfectly observed and there is negligible communication delay between the agents.

Assumption 1 is required to evaluate the idealized performance of the generated optimal solution. In general, this assumption may be relaxed by formulating a stochastic optimal control problem to generate agent trajectories.

Assumption 2. The energy cost of communication is negligible; the only energy consumption is in the form of (2.7).

The strength of this assumption is application dependent. For cases with long-distance communications or high data rates, the trade-off between communication and motion costs can be controlled by varying the sensing and communicating radius, h, of the agents.

To solve the desired formation problem, we first relax the inter-agent collision avoidance constraint to decouple the agent trajectories. This decoupling reduces the problem from a single mixed-integer program to a coupled pair of binary and quadratic programs, which we solve sequentially. This decoupling is common in the literature [68, 70], and usually does not affect the outcome of the assignment problem.

Next, we present some preliminary results before decomposing the desired formation problem into the two subproblems, minimum-energy goal assignment and trajectory generation.

2.1.1.1 Preliminaries

First we consider that any agent $i \in \mathcal{A}$ obeys double-integrator dynamics, (2.1)–(2.2), and has an energy model with the form of (2.7).Then, we let the state and control trajectories of agent i be unconstrained, i.e., relax (2.3), (2.4), and (2.9). In this case, if i is traveling between two fixed states, the unconstrained minimum-energy trajectory is given by the following system of linear equations:

$$\mathbf{u}_i(t) = \mathbf{a}_i\ t + \mathbf{b}_i, \tag{2.11}$$

$$\mathbf{v}_i(t) = \frac{\mathbf{a}_i}{2}\ t^2 + \mathbf{b}_i\ t + \mathbf{c}_i, \tag{2.12}$$

$$\mathbf{p}_i(t) = \frac{\mathbf{a}_i}{6}\ t^3 + \frac{\mathbf{b}_i}{2}\ t^2 + \mathbf{c}_i\ t + \mathbf{d}_i, \tag{2.13}$$

where $\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i$, and $\mathbf{d}_i$ are constant vectors of integration. The derivation of (2.11)–(2.13) is straightforward, and it explicitly derived in Chapter 4.

Thus, the energy consumed for any unconstrained trajectory of agent $i \in \mathcal{A}$ at time t traveling towards the goal $j \in \mathcal{F}$ is given by

$$\begin{aligned} E_i^j(t) = \int_t^{t_i^f} ||\mathbf{u}_i(\tau)||^2\ d\tau &= (t_f^3 - t^3)\Big(\frac{a_{i,x}^2 + a_{i,y}^2}{3}\Big) \\ &+ (t_f^2 - t^2)\Big(a_{i,x}\ b_{i,x} + a_{i,y}\ b_{i,y}\Big) \\ &+ (t_f - t)\Big(\frac{b_{i,x}^2 + b_{i,y}^2}{2}\Big), \end{aligned} \tag{2.14}$$

where $t \in [t_i^0, t_i^f]$, and $\mathbf{a}_i = [a_{i,x}, a_{i,y}]^T$, $\mathbf{b}_i = [b_{i,x}, b_{i,y}]^T$ are the coefficients of (2.11).

Next, we present the interaction dynamics between agents. To resolve any conflict between agents, we consider the following objectively measurable constants: 1)

neighborhood size, 2) energy required to reach a goal, and 3) agent index, which may be arbitrarily assigned. Each of these quantities has an associated indicator function for comparing two agents $i, j \in \mathcal{A},\ j \neq i$,

$$\mathbb{1}_{ij}^{\mathcal{N}}(t) \coloneqq \begin{cases} 1 & |\mathcal{N}_i(t)| > |\mathcal{N}_j(t)|, \\ 0 & |\mathcal{N}_i(t)| \leq |\mathcal{N}_j(t)|, \end{cases} \tag{2.15}$$

$$\mathbb{1}_{ij}^{\mathcal{E}}(t) \coloneqq \begin{cases} 1 & E_i(t) > E_j(t), \\ 0 & E_i(t) \leq E_j(t), \end{cases} \tag{2.16}$$

$$\mathbb{1}_{ij}^{\mathcal{A}}(t) \coloneqq \begin{cases} 1 & i > j, \\ 0 & i < j. \end{cases} \tag{2.17}$$

Next, we define the interaction dynamics by combining (2.15) - (2.17) into a single indicator function.

Definition 3. We define the *interaction dynamics* between any agent $i \in \mathcal{A}$ and another agent $j \in \mathcal{N}_i(t), j \neq i$ as

$$\begin{aligned} \mathbb{1}_{ij}^{C}(t) = \mathbb{1}_{ij}^{\mathcal{N}}(t) + \big(1 - \mathbb{1}_{ij}^{\mathcal{N}}(t)\big)\big(1 - \mathbb{1}_{ji}^{\mathcal{N}}(t)\big) \\ \Big(\mathbb{1}_{ij}^{\mathcal{E}}(t) + \big(1 - \mathbb{1}_{ij}^{\mathcal{E}}(t)\big)\big(1 - \mathbb{1}_{ji}^{\mathcal{E}}(t)\big)\mathbb{1}_{ij}^{\mathcal{A}}(t)\Big), \end{aligned} \tag{2.18}$$

where $\mathbb{1}_{ij}^{C} = 1$ implies agent i has priority over agent j, and $\mathbb{1}_{ij}^{C} = 0$ implies that agent j has priority over agent i.

The interaction dynamics are instantaneously and noiselessly measured and communicated by each agent under Assumption 1. Whenever two agents have a conflict (i.e., share an assigned goal, or have overlapping assignments) (2.18) is used to impose an order on the agents such that higher priority agents act first.

Remark 1. For any pair of agents $i \in \mathcal{A}, j \in \mathcal{N}_i(t), j \neq i$, it is always true that $\mathbb{1}_{ij}^{C}(t) = 1 - \mathbb{1}_{ji}^{C}(t)$, i.e., the outcome of the interaction dynamics (2.18) is always

unambiguous, and therefore it imposes an order on any pair of agents.

Remark 1 can be proven by simply enumerating all cases of (2.15) - (2.17).

2.1.2 Optimal Goal Assignment

The optimal solution of the assignment problem must assign each agent to a goal such that the total unconstrained energy cost, given by (2.14), is minimized. In our framework, each agent $i \in \mathcal{A}$ only has information about the positions of its neighbors, $j \in \mathcal{N}_i(t)$, and the goal positions prescribed by $\mathcal{G}$. Agent i derives the goal assignment using a binary matrix $\mathbf{A}_i$(t), which we define next.

Definition 4. For each agent $i \in \mathcal{A}$, we define the *assignment matrix*, $\mathbf{A}_i(t)$, as an $|\mathcal{N}_i(t)| \times M$ matrix with binary elements. The elements of $\mathbf{A}_i(t)$ map each agent to exactly one goal, and each goal to no more than one agent.

The assignment matrix for agent $i \in \mathcal{A}$ assigns all agents in $\mathcal{N}_i(t)$ to goals by considering the cost (2.14). We discuss the details of the optimal assignment problem later in this section.

Next we define the prescribed goal, which determines how each agent $i \in \mathcal{A}$ assigns itself a goal.

Definition 5. We define the *prescribed goal* for agent $i \in \mathcal{A}$ as the goal assigned to agent i by the rule,

$$\mathbf{p}_i^a(t) \in \left\{ \mathbf{p}_k \in \mathcal{G} \mid a_{ik} = 1,\ a_{ik} \in \mathbf{A}_i(t), k \in \mathcal{F} \right\}, \tag{2.19}$$

where $\mathbf{A}_i(t)$ is the assignment matrix, and the right hand side is a singleton set, i.e., agent i is assigned to exactly one goal.

Next, we present the goal assignment algorithm in terms of some agent $i \in \mathcal{A}$. However, as this framework is cooperative, each step is performed by all individuals simultaneously.

In some cases, multiple agents may select the same prescribed goal. This may occur when two agents $i \in \mathcal{A}, j \in \mathcal{N}_i(t), j \neq i$ have different neighborhoods and use conflicting information to solve their local assignment problem. This motivates the introduction of competing agents, which we define next.

Definition 6. For agent $i \in \mathcal{A}$, we define the set of *competing agents* as

$$\mathcal{C}_i(t) = \Big\{ k \in \mathcal{N}_i(t) \mid \mathbf{p}_k^a(t) = \mathbf{p}_i^a(t) \Big\}.$$

When $\big|\mathcal{C}_i\big| > 1$ there are at least two agents, $i, j \in \mathcal{N}_i(t)$ which are assigned to the same goal. In this case, all but one agent in $\mathcal{C}_i(t)$ must be *permanently banned* from the goal $\mathbf{p}_i^a(t)$. Next, we define the banned goal set.

Definition 7. The *banned goal set* for agent $i \in \mathcal{A}$ is defined as

$$\begin{aligned} \mathcal{B}_i(t) = \Big\{ g \in \mathcal{F} \;\Big|\; & \mathbf{p}_i^a(\tau) = \mathbf{p}_g(\tau) \in \mathcal{G}, \\ & \Big(\prod_{j \in \mathcal{C}_i(\tau), j \neq i} \mathbb{1}_{ij}^c(\tau) \Big) = 0, \; \exists\, \tau \in [t_i^0, t] \Big\}, \end{aligned} \tag{2.20}$$

i.e., the set of all goals which agent $i \in \mathcal{A}$ had a conflict over and did not have priority per Definition 3.

Banning is achieved by applying (2.20) to all agents $j \in \mathcal{C}_i(t)$ whenever $|\mathcal{C}_i(t)| > 1$. After the banning step is completed, agent $i \in \mathcal{A}$ checks if the size of $\mathcal{B}_i(t)$ has increased. If so, agent i increases the value of t_i^f by

$$t_i^f = t + T, \tag{2.21}$$

where t is the current time, and $T \in \mathbb{R}_{>0}$ is a system parameter. This allows agent i a sufficient amount of time to reach its new goal.

Next, each agent broadcasts its new set of banned goals to all of its neighbors. Any agent who was banned from $\mathcal{C}_i(t)$ assigns itself to a new goal. However, this may

cause new agents to enter $\mathcal{C}_i(t)$ as they are banned from other goals. To ensure each agent $j \in \mathcal{N}_i(t)$ is assigned to a unique goal, the assignment and banning steps are iterated until the condition

$$\left|\mathcal{C}_j(t)\right| = 1, \quad \forall j \in \mathcal{N}_i(t), \tag{2.22}$$

is satisfied. For a given neighborhood $\mathcal{N}_i(t), i \in \mathcal{A}$, some number of agents will be assigned to the goal $g \in \mathcal{F}$. After the first banning step, all agents except the one which was assigned to goal g are permanently banned and may never be assigned to it again. If additional agents are assigned to g, then this process will repeat for at most $N-1$ iterations. Afterwards every goal g will have at most one agent from $\mathcal{N}_i(t)$ assigned to it. Thus, we will have $|\mathcal{C}_j(t)| = 1$ for all $j \in \mathcal{N}_i(t)$ for every $i \in \mathcal{A}$. We enforce the banned goals through a constraint on the assignment problem, which follows.

Problem 1 (Goal Assignment)**.** Each agent $i \in \mathcal{A}$ selects its prescribed goal (Definition 5) by solving the following binary program

$$\min_{a_{jk} \in \mathbf{A}_i} \left\{ \sum_{j \in \mathcal{N}_i(t)} \sum_{k \in \mathcal{F}} a_{jk} E_j^k(t) \right\} \tag{2.23}$$

subject to:

$$\sum_{j \in \mathcal{N}_i(t)} a_{jk} \leq 1, \quad k \in \mathcal{F}, \tag{2.24}$$

$$\sum_{k \in \mathcal{F}} a_{jk} = 1, \quad j \in \mathcal{N}_i(t), \tag{2.25}$$

$$a_{jk} = 0, \quad \forall\ k \in \mathcal{B}_j(t),\ j \in \mathcal{N}_i(t), \tag{2.26}$$

$$a_{jk} \in \{0, 1\}.$$

This process is repeated by each agent, $i \in \mathcal{A}$, until (2.22) is satisfied for all $j \in \mathcal{N}_i(t)$.

As the conflict condition in Problem 1 explicitly depends on the neighborhood of agent $i \in \mathcal{A}$, Problem 1 may need to be recalculated each time the neighborhood of agent i switches. The assignments generated by Problem 1 are guaranteed to bring each agent to a unique goal; we show this with the help of Assumption 3 and Lemma 1.

Assumption 3. For every agent $i \in \mathcal{A}$, for all $t \in [t_i^0, t_i^f]$, the inequality $\left|\left(\mathcal{F} \setminus \bigcup_{j \in \mathcal{N}_i(t)} \mathcal{B}_j(t)\right)\right| \geq |\mathcal{N}_i(t)|$ holds.

Assumption 3 is a condition that is sufficient but not necessary to prove convergence of our proposed optimal controller. Intuitively, Assumption 3 only requires that one agent does not ban many agents from many goals. Due to the minimum-energy nature of our framework, this scenario is unlikely; additionally, permanent banning may be relaxed to temporary banning in a way that Assumption 3 is always satisfied.

Lemma 1 (Solution Existence)**.** Under Assumption 3, the feasible region of Problem 1 is nonempty for agent i.

Proof. Let the set of goals available to all agents in the neighborhood of agent $i \in \mathcal{A}$ be denoted by the set

$$\mathcal{L}_i(t) = \{p \in \mathcal{F} \mid p \notin \mathcal{B}_j(t),\ \forall j \in \mathcal{N}_i(t)\}. \tag{2.27}$$

Let the injective function $w : \mathcal{A} \to \mathcal{F}$ map each agent to a goal. By Assumption 3, $|\mathcal{N}_i(t)| \leq |\mathcal{L}_i(t)|$, thus a function w exists. As w is injective, the imposed mapping must satisfy (2.24) and (2.25). Likewise, $\mathcal{L}_i(t) \bigcap \mathcal{B}_j(t) = \emptyset$ for all $j \in \mathcal{N}_i(t)$. Thus, w must satisfy (2.26). Therefore, the mapping imposed by the function w is a feasible solution to Problem 1. □

Next, we show that for a sufficiently large value of T the convergence of all agents to goals is guaranteed.

Theorem 1 (Assignment Convergence)**.** Under Assumption 3, for sufficiently large values of the initial t_i^f and T, and if the energy-optimal trajectories for agent $i \in \mathcal{A}$ never increase the unconstrained energy cost (2.14), then t_i^f must have an upper bound for all $i \in \mathcal{A}$.

Proof. Let $\{g_n\}_{n\in\mathbb{N}}$ be the sequence of goals assigned to agent $i \in \mathcal{A}$ by the solution of Problem 1. By Lemma 1, $\{g_n\}_{n\in\mathbb{N}}$ must not be empty, and the elements of this sequence are natural numbers bounded by $1 \leq g_n \leq M$. Thus, the range of this sequence is compact, and the sequence must be either (1) finite, or (2) convergent, or (3) periodic.

1) For a finite sequence there is nothing to prove, as t_i^f is upper bounded by $t_{i,\text{initial}}^f + MT$.

2) Under the discrete metric, an infinite convergent sequence requires that there exists $N \in \mathbb{N}_{>0}$ such that $g_n = p$ for all $n > N$ for some formation index $p \in \mathcal{F}$. This reduces to case 1, as t_i^f does not increase for repeated assignments to the same goal.

3) By the Bolzano-Weierstrass Theorem, an infinite non-convergent sequence $\{g_n\}_{n\in\mathbb{N}}$ must have a convergent subsequence, i.e., agent i is assigned to some subset of goals $\mathcal{I} \subseteq \mathcal{F}$ infinitely many times with some constant number of intermediate assignments, P_g, for each goal $g \in \mathcal{I}$. Necessarily, $\mathcal{I} \bigcap \mathcal{B}_i(t) = \emptyset$ for all $t \in [t_i^0, t_i^f]$ from the construction of the banned goal set. This implies that, by the update method of t_i^f, the position of all goals, $g \in \mathcal{I}$ must only be considered at time t_i^f.

This implies that the goals available to agent i, i.e., $\mathcal{I} = \mathcal{F} \setminus \mathcal{B}_i(t_i^f)$, must be shared between $n > 0$ other periodic agents. Hence, at some time t_1 a goal, $g \in \mathcal{I}$, must be an optimal assignment for agent i, a non optimal assignment at time $t_2 > t_1$ and an optimal assignment again at time t_3 which corresponds to the ${P_g}^{th}$ assignment.

As $t_3 > t_2 > t_1$, the energy required to move agent i to goal g satisfies

$$E_i^g(t_1) \leq E_i^k(t_1), \tag{2.28}$$

$$E_i^k(t_2) \leq E_i^g(t_2), \tag{2.29}$$

$$E_i^g(t_3) \leq E_i^k(t_3), \tag{2.30}$$

for any other goal $k \in \mathcal{I}, k \neq g$. Therefore, for agent i to follow an energy optimal trajectory under our premise, it must never increase the energy required to reach is assigned goal, which implies

$$E_i^g(t_1) \geq E_i^g(t_2), \tag{2.31}$$

$$E_i^k(t_2) \geq E_i^k(t_3), \tag{2.32}$$

this implies

$$E_i^k(t_1) \geq E_i^k(t_3), \tag{2.33}$$

which is only possible if agent i simultaneously approaches all goals $k \in \mathcal{I}$. This implies that goals g and k are arbitrarily close, which violates the minimum spacing requirements of the goals; therefore no such periodic behavior may exist. □

Note that Theorem 1 bounds the arrival time of agent $i \in \mathcal{A}$ to any goal $g \in \mathcal{F}$. A similar bound may be found for the total energy consumed, i.e.,

$$E_i^g(t) \leq \frac{1}{2}(t_{i,\text{initial}}^f + MT) \cdot \max\left\{|u_{\min}|, |u_{\max}|\right\}^2.$$

Next, we formulate the optimal trajectory generation problem for each agent and prove that the resulting trajectories always satisfy the premise of Theorem 1.

2.1.3 Optimal Trajectory Generation

After the goal assignment is complete, each agent must generate a collision-free trajectory to their assigned goal. The trajectories must minimize the agent's total energy cost subject to dynamic, boundary, and collision constraints. The initial and final state constraints for each agent $i \in \mathcal{A}$ are given by

$$\mathbf{p}_i(t_i^0) = \mathbf{p}_i^0, \qquad \mathbf{v}_i(t_i^0) = \mathbf{v}_i^0, \tag{2.34}$$

$$\mathbf{p}_i(t_i^f) = \mathbf{p}_i^a(t_i^f), \qquad \mathbf{v}_i(t_i^f) = \dot{\mathbf{p}}_i^a(t_i^f), \tag{2.35}$$

where the conditions at t_i^f come from the solution of Problem 1.

Whenever an agent must steer to avoid collisions, we will apply the agent interaction dynamics (Definition 3) to impose an order on the agents such that lower priority agents must steer to avoid the higher priority ones. Thus, we may simplify the collision avoidance constraint for agent $i \in \mathcal{A}$ to

$$\begin{aligned} ||\mathbf{s}_{ij}(t)|| \geq 2R, \quad \forall\, j \in \{k \in \mathcal{A} \mid \mathbb{1}_{ik}(t) = 0\}, \\ \forall\, t \in [t_i^0, t_i^f], \end{aligned} \tag{2.36}$$

which will always guarantee safety for all agents.

We may then formulate the decentralized optimal trajectory generation problem.

Problem 2. For each agent $i \in \mathcal{A}$, find the optimal control input, $\mathbf{u}_i(t)$, which minimizes the energy consumption of agent i and satisfies its boundary conditions and safety constraints.

$$\min_{\mathbf{u}_i(t)} \frac{1}{2} \int_{t_i^0}^{t_i^f} ||\mathbf{u}_i(t)||^2 \, dt \tag{2.37}$$

subject to: (2.1), (2.2), (2.34), (2.35), and (2.36).

By imposing an order on the agents, we can show that the solution of Problem 2 will always satisfy the premise of Theorem 1. First, Lemma 2 shows that an unconstrained trajectory must never increase the energy required to reach a goal.

Lemma 2. For any agent $i \in \mathcal{A}$, following the unconstrained trajectory, the energy cost (2.14) required to reach a fixed goal $g \in \mathcal{F}$ is not increasing.

Proof. We may write the derivative of (2.14) along an unconstrained trajectory as

$$\begin{aligned}\frac{dE_i^g(t)}{dt} &= \lim_{\delta \to 0} \frac{1}{\delta} \left(\int_{t+\delta}^{t_i^f} ||\mathbf{u}_i(\tau)||^2 d\tau - \int_t^{t_i^f} ||\mathbf{u}_i(\tau)||^2 d\tau \right) \\ &= -\lim_{\delta \to 0} \frac{1}{\delta} \int_t^{t+\delta} ||\mathbf{u}_i(\tau)||^2 d\tau, \end{aligned} \tag{2.38}$$

which is never positive. Therefore, (2.14) is never increasing. □

Next, we introduce Theorem 2, which proves the premise of Theorem 1 is always satisfied by any trajectory which is a feasible solution to Problem 2.

Theorem 2. If a solution to Problem 2 exists for all agents, then Theorem 1 is satisfied as long as Assumption 3 holds.

Proof. The case when any agent $i \in \mathcal{A}$ is moving with an unconstrained trajectory is covered by Lemma 2, so we focus on the case when any of the safety constraints are active.

Let $\mathcal{K} \subseteq \mathcal{A}$ be a group of agents which all have their safety constraint active over some interval $t \in [t_1, t_2]$. By Definition 3, there exists some $i \in \mathcal{K}$ such that $\mathbb{1}_{ij}^C(t) = 1$ for all $j \in \mathcal{K},\ j \neq i$. Therefore, agent i satisfies Lemma 2 and always moves toward its assigned goal by Theorem 1.

Next, consider agent $j \in \mathcal{V} \setminus \{i\}$ such that $\mathbb{1}_{jk}^c(t) = 1$ for all $k \in \mathcal{K} \setminus \{i\}$. As agent j may never be assigned to the same goal as i, there must exist some time $t_j < \min\{t_i^f, t_j^f\}$ such that $|\mathbf{s}_{ij}(t_j)| > 2R$ by the goal spacing rules. Thus, agent j

will move with an unconstrained trajectory for all $t \in [t_j, t_j^f]$. The above steps can be recursively applied until only a single agent remains, which follows an unconstrained trajectory for some finite time interval. This satisfies the premise of Theorem 1. □

The precise approach to generating optimal trajectories for the agents is omitted for brevity; it is discussed in Chapter 4 of this dissertation. Next, we present simulation results that validate our proposed control algorithm.

2.1.4 Simulation Results

To provide insight into the behavior of the agents, a series of simulations were performed with $M = N = 10$ agents and a time parameter of $T = 10$ s. The simulations were run for $t = 20$ s or until all agents reach their assigned goal, whichever occurred later. The center of the formation moved with a velocity of $\mathbf{v}_{cg} = [0.15,\ 0.35]$ m/s; the leftmost and rightmost three goals each move with an additional periodic velocity of $[0.125 \cos 0.75t,\ 0]$ m/s relative to the formation. Videos of the simulation results can be found at https://sites.google.com/view/ud-ids-lab/omas.

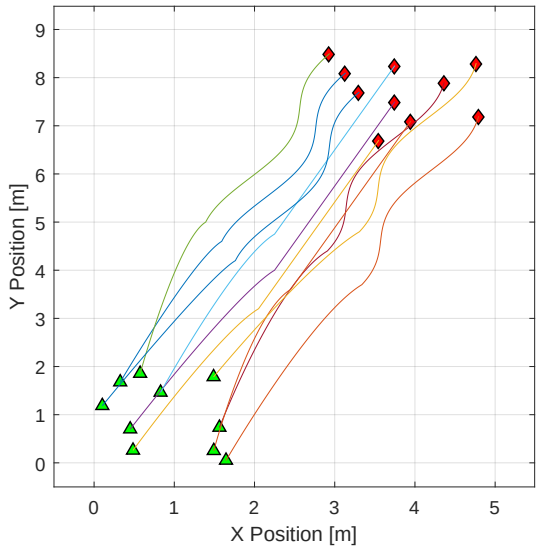

Figure 2.1: Simulation result for the centralized case. Goals which minimize the unconstrained trajectories are assigned to the agents once at t_i^0.

The minimum separating distance between agents, total energy consumed, and maximum velocity for the unconstrained solutions to Problem 2 are all given as a function of the horizon in Table 2.1. The energy consumption only considers the energy required to reach the goal, which, in this case, was significantly lower than the energy required to maintain the formation. The trajectory of each agent over time is given in Figures 2.1–2.3 for varying sensing horizon values. Although the trajectories may appear to cross in Figures 2.1–2.3, they are only crossing in space and not in time.

Table 2.1: Numerical results for N=10 agents and M=10 goals for various sensing distances.

h [m]	min. separation [cm]	energy [J/kg]	t_f [s]	Total Bans
∞	25.25	0.85	20	0
1.60	1.64	1.10	20	4
1.50	1.60	1.17	20	24
1.40	2.01	1.96	23.3	31
1.30	0.33	1670	26.05	36
1.20	0.65	866	25.35	34
1.10	1.05	5370	26.85	40
1.00	1.96	7609	30.65	35
0.95	3.12	3149	25.05	27
0.75	1.37	6.87	20	35
0.50	0.27	692.0	26.65	35

The performance of our algorithm is strongly affected by how much information is available to each agent. This is a function of the sensing horizon, initial states of the agents, and the desired formation shape. Generally, better overall performance requires the agents to have more information. However, it is not apparent what information is necessary; in fact, the results in Table 2.1 generally show no correlation between energy consumption and sensing horizon.

The trade-off for more information is in the computational and sensing load imposed on each agent. As an agent observes more of the system (via sensing, communication, or memory), the computational burden to solve the assignment and trajectory

generation problems also increases. However, this computational cost does not necessarily result in improved system performance, as demonstrated in Table 2.1.

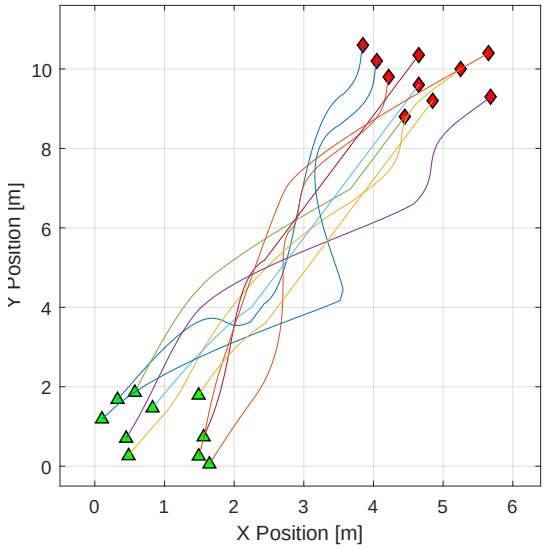

Figure 2.2: Simulation result for $h = 1.30$ m. The agents do not start with a globally unique assignment, and several agents must re-route partway through the simulation. Although the trajectories cross in space they do not cross in time.

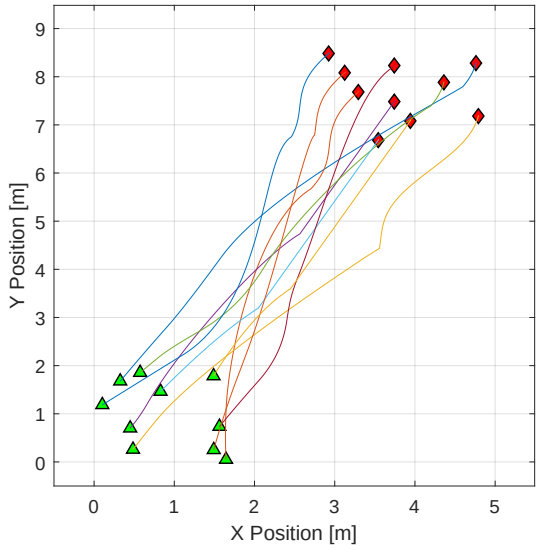

Figure 2.3: Simulation result for $h = 0.75$ m. Although the horizon for this case is smaller than in Figure 2.2, the system dynamics happen to result in more efficient trajectories overall.

2.2 Concluding Remarks

This chapter framed goal assignment as a "medium-complexity" problem of emergence. The decentralized nature of the problem stems from the fact that the agents only require global information about the goal positions. While the system is guaranteed to converge under Assumption 3, this in general too restrictive; there is no way to verify that the assumption holds without simulating the evolution of the system! As a remedy, we proposed an event-driven assignment mechanism in [44], which implies Assumption 3.

Theorem 3. Let each agent $i \in \mathcal{A}$ be assigned to an initial goal $j \in \mathcal{F}$. If, after the initial assignment, agent i only updates its assignment when $j \in \mathcal{B}_i$, i.e., when i becomes banned from j, then Problem 1 always has a solution.

Proof. When an agent i is banned from a goal $k \in \mathcal{F}$, then some agent in $\mathcal{A}$ must be assigned to goal k for all future time. Let $C = |\bigcup_{i \in \mathcal{A}} \mathcal{B}_i|$. Then, one feasible solution

to Problem 1 is to assign, at most, $N - C$ remaining agents to the $M - C$ unbanned goals. □

Thus, once the system is initialized, Theorem 1 guarantees that all agents will converge to a unique goal position in finite time. Viewing emergence through the lens of an engineering design problem also enables us to precisely describe the "goodness" of an equilbrium point using engineering design principles; in this case, the energy required for each agent to reach its assigned goal. This raises further questions: given the nonmonotonic results in Table 2.1, what alternative set of interaction rules might yield improved system performance? Do the agents require more information to make optimal decisions when the sensing horizon is smaller, or do these results show that the agents may be receiving unnecessary information? Note that this non-monotonicity property is not resolved using the event-driven assignment [44]. The idea of energy-minimizing agents is further addressed in Chapter 3, where the agent interactions are specifically designed to drive the system into an energy-minimizing configuration.

Chapter 3

CONSTRAINT-DRIVEN CONTROL FOR MULTI-AGENT SYSTEMS

> You insist that there is something a machine cannot do. If you tell me precisely what it is a machine cannot do, then I can always make a machine which will do just that.
>
> John von Neumann
> Public Lecture (1948)

This chapter builds off the results of Chapter 2, which demonstrate how emergent behavior can be achieved by driving agents to equilibrium points. This chapter achieves this using constraint-driven control, a *declarative control policy* wherein the system is constrained to a feasible domain and a measure of performance is designed to optimally select a control action at each time instant [71]. This is in contrast to a *procedural policy*, such as PID or LQR, where a feedback control law is explicitly defined by a designer a priori. In particular, I present material from two articles [47, 48] that push agents to energy-minimizing equilibrium points; the third article [47] expands this approach to minimize the locomotive energy of the agents subject to a greater number of constraints.

A significant recent trend in the constraint-driven control literature is the concept of ecologically-inspired robotics [45]. In ecologically-inspired robotics, each individual is presumed to interact with their environment on timescales significantly longer than what can be achieved in a laboratory setting. These approaches necessarily emphasize safe energy-minimizing control policies for the agents, where agents minimize their control effort subject to the environment, task constraints, and safety.

The ecologically-inspired robotics paradigm has been applied to many multi-agent control problems in simulations and experiments [29, 59, 72], and shows great promise for long-duration autonomy tasks.

As an extension of ecologically-inspired robotics, the first two articles in this chapter push a collection of agents into an energy-minimizing configuration by considering their aerodynamic interactions. The first case considers a fleet of connected and automated vehicles (CAVs) traveling on a highway [47], where agents may unexpectedly enter or exit the system at prescribed on and off ramps. In particular, each CAV can save energy by exploiting the low-pressure aerodynamic wake of the preceding CAV; the resulting analysis also yields sufficient conditions for when a CAV will join a platoon. In the second article [48], a fleet of uncrewed aerial vehicles (UAVs) maximize the aerodynamic benefit received from their neighbors' upwash. This leads to the emergence of stable V formations, which reduces the total energy cost of the entire system. We also prove that V formations are unstable if each UAV greedily minimizes its experienced drag force.

In addition to the energy-saving aspect, it is also necessary to demonstrate recursive feasibility, i.e., that each agent has a non-empty set of feasible actions at each time instant. The third article in this chapter [48] proposes a new approach to guaranteeing recursive feasibility for constraint-driven problems. Most existing work [29, 45, 72] introduces slack variables on constraints that are not critical to safety; this ensures that a safe actions can always be selected with a corresponding penalty in the objective function. However, such approaches are not fundamentally different than simply moving the constraints back into the objective function, which defeats the purpose of using constraint-driven techniques. Other articles introduce additional "compatibility constraints " to guarantee solution existence [73], however, we expect this approach to grow prohibitively conservative as the number of constraints increases. Still more results look for "minimum violating" solutions [74], whereas the constraint relaxation technique always prioritizes safety. The constraint relaxation approach is

fundamentally similar to an article on multi-agent control barrier functions [75], which proposes a transition to a "safe mode" of operation when no feasible control inputs exist; the results in this chapter take this idea to its logical conclusion. We demonstrate that this approach yields an equivalent switching system, which is easy to interpret while still admitting strong guarantees on the system-level behavior and the safety of individual agents.

Finally, each of the articles in this chapter demonstrates how constraint-driven control is a rigorous data-driven technique. Since the behavior of each agent is encoded using constraints, each agent must use information about the state of its neighborhood and the local environment to determine the set of feasible actions. This results in an easy to interpret solution, where a designer must only examine the switching system and active constraints to understand the the behavior of each agent. The following sections are composed of three articles [46–48], which have been lightly edited to fit within this dissertation.

3.1 Constraint-Driven Optimal Control of Multi-Agent Systems: A Highway Platooning Case Study

3.1.1 Introduction

In this section, we propose a constraint-driven approach to generate emergent platooning behavior in a fleet of connected and automated vehicles operating in highway conditions. Platooning behavior has been of particular interest due to the high potential for energy savings over long distances [76–78], thus we believe that platoon formation for long-distance highway travel is a natural fit for constraint-driven control. There are several approaches to optimal platoon formation in the literature. In one example, the authors sought to optimally position differential drive robots in an echelon formation such that energy lost to drag was minimized [31]. Reynolds' flocking rules were applied to highway vehicles in [79], which sought to minimize energy consumption while maintaining a desired speed. Energy-efficient flocking was also proposed for a

system of flying robots in $\mathbb{R}^2$ [32]. Previous approaches either construct a large multi-objective optimization problem to determine the next control action, or they apply sub-optimal consensus algorithms to reach a drafting configuration. A recent review of these techniques is presented in [9].

Our approach, in contrast to existing work, is constraint-driven. In our framework, agents seek to expend as little energy as possible subject to a set of task and safety constraints. This set-theoretic approach to control is interpretable, i.e., the cause of an agent's action can be deduced by examining which constraints become active during operation. By examining the conditions that lead to an empty feasible space, our framework also addresses when a vehicle should break away to form a new platoon or overtake the preceding vehicle. Our approach is totally decentralized, and thus it is well-suited to "open systems," where agents may suddenly enter or leave. We allow vehicles to arbitrarily enter or exit the system as long as their initial state is feasible and no other vehicles' safety constraint is violated. This also allows vehicles to keep their final destination and arrival time private, which has the secondary benefit of guaranteeing privacy for all vehicles and their passengers.

The remainder of the section is organized as follows. In Section 3.1.2, we formulate the platoon formation problem, and in Section 3.1.3, we present our decentralized constraint-driven control algorithm. Finally, in Section 3.1.4, we validate our results by simulating 60 vehicles, where vehicles randomly enter and leave the road network while the total number of vehicles is not known a priori.

3.1.2 Problem Formulation

We consider a set of CAVs traveling in a single-lane roadway. In particular, we consider an open transportation system that contains $N(t) \in \mathbb{N}$ CAVs indexed by the set $\mathcal{N}(t) = \{0, 1, 2, \ldots, N(t) - 1\}$, where $t \in \mathbb{R}$ is time and vehicle $i \in \mathcal{N}(t) \setminus \{0\}$ is in the aerodynamic wake of vehicle $i - 1$. We denote the state of each CAV $i \in \mathcal{N}(t)$ by $\mathbf{x}_i(t) = \Big[p_i(t), v_i(t)\Big]^T$, where $p_i(t), v_i(t) \in \mathbb{R}$ are the longitudinal position and speed of

vehicle i on its current path respectively. Each vehicle obeys the second-order dynamics

$$\begin{aligned}\dot{p}_i(t) &= v_i(t),\\ \dot{v}_i(t) &= a_i(t) = u_i(t) - F_i\big(v_i(t), \hat{p}_i(t)\big),\end{aligned} \tag{3.1}$$

where $a_i(t)$ is acceleration, $u_i(t)$ is forward force imparted through the tires, $F_i\big(v_i(t), \hat{p}_i(t)\big)$ is the aerodynamic drag force acting on the vehicle, and $\hat{p}_i(t)$ is the relative position of CAV i, which we formally define later. The objective of each vehicle is to minimize the effect of the external drag force, i.e.,

$$J_i(v_i(t), \hat{p}_i(t)) = \frac{1}{2} F_i\big(v_i(t), \hat{p}_i(t)\big)^2. \tag{3.2}$$

By minimizing the external drag force of each vehicle, we have direct benefits in energy consumption. Each vehicle i is subject to state and control constraints, i.e.,

$$0 < v_{\min} \leq v_i(t) \leq v_{\max}, \tag{3.3}$$

$$a_{\min} \leq a_i(t) \leq a_{\max}, \tag{3.4}$$

where (3.3) is the lower and upper speed limit and (3.4) is the maximum safe deceleration and acceleration.

We index the vehicles in descending order, i.e., $p_i(t) < p_j(t)$ for all $i > j,\ i, j \in \mathcal{N}(t)$. Note, when a vehicle enters or exits the system, the CAVs can communicate to re-sequence themselves. To simplify our notation, we introduce the *relative state* coordinates.

Definition 8. For each vehicle $i \in \mathcal{N}(t)$, the relative states and control action are,

$$\hat{p}_i(t) = \begin{cases} p_i(t) & \text{if } i = 0, \\ p_i(t) - p_{i-1}(t) & \text{if } i > 0, \end{cases} \tag{3.5}$$

$$\hat{v}_i(t) = \begin{cases} v_i(t) & \text{if } i = 0, \\ v_i(t) - v_{i-1}(t) & \text{if } i > 0, \end{cases} \tag{3.6}$$

$$\hat{a}_i(t) = \begin{cases} \dot{v}_i(t) & \text{if } i = 0, \\ \dot{v}_i(t) - \dot{v}_{i-1}(t) & \text{if } i > 0. \end{cases} \tag{3.7}$$

Note that in this coordinate system $\hat{p}_i(t) < 0$ for $i > 0$. While our approach does not impose a reference frame, it may be practical for a physical vehicle to measure (3.5) - (3.6) directly, i.e., by using a proximity sensor. In that case, it may be advantageous for each vehicle to consider its current state as the center of a moving reference frame.

To guarantee safety we impose the following safety constraint,

$$\hat{p}_i(t) + \delta \leq 0, \quad i \in \mathcal{N} \setminus \{0\}, \tag{3.8}$$

where $\delta \in \mathbb{R}_{>0}$ is the minimum safe bumper-to-bumper following distance. However, naïvely satisfying (3.8) may still lead to unsafe scenarios and collisions. Consider the case when $\hat{v}_i(t)$ is very large and vehicle $i-1$ applies $a_{i-1}(t) = a_{\min}$. To guarantee vehicle i never ends up in an unsafe scenario, we impose an augmented safety constraint that guarantees sufficient stopping distance,

$$g_i^s(v_i, \hat{p}_i, \hat{v}_i) = \begin{cases} \hat{p}_i(t) + \delta & \text{if } \hat{v}_i(t) \leq 0, \\ \hat{p}_i(t) + \delta + \hat{v}_i(t) \cdot \left(\frac{v_{\min} - v_i(t)}{a_{\min}}\right) & \\ \qquad + \frac{\hat{v}_i(t)^2}{2a_{\min}} & \text{if } \hat{v}_i(t) > 0, \end{cases} \tag{3.9}$$

for $i \in \mathcal{N} \setminus \{0\}$. The case when $\hat{v}_i(t) > 0$ in (3.9) is derived for CAV i by assuming $i-1$

applies the maximum braking force until $v_{i-1}(t) = v_{\min}$ at some time t_1 and cruises with $a_{i-1}(t) = 0$, for $t \geq t_1$. Then, (3.9) allows CAV i sufficient stopping distance to brake at $a_{\min}$ and maintain $\hat{p}_i(t) + \delta = 0$. Note that the quadratic $\hat{v}_i(t)$ term is zero when $\hat{v}_i(t) = 0$ and increases up to a maximum at $\hat{v}_i(t) = v_{\max} - v_{\min}$. Thus, satisfaction of (3.9) always implies (3.8).

Finally, each vehicle $i \in \mathcal{N}(t)$ has a terminal time t_i^f, which corresponds to the time that vehicle i will exit the system, e.g., take an exit off the highway. The value of t_i^f is known only to vehicle i and is not shared with any other vehicle. This also ensures the privacy of vehicle i's destination. To ensure vehicle i reaches its destination by time t_i^f, we impose an arrival deadline constraint,

$$\big(S_i - p_i(t)\big) - \big(t_i^f - t\big)\, v_i(t) \leq 0, \tag{3.10}$$

where S_i is the position that i will exit the system, e.g., via an off-ramp. The arrival deadline constraint (3.10) ensures that vehicle i can reach its final destination by cruising at a constant speed.

Our objective in this section is the formation of platoons for long-duration autonomy, e.g., long-distance highway conditions. Therefore, once vehicle i joins a platoon, i.e., $\hat{v}_i = 0$ and $\hat{p}_i + \delta = 0$, other techniques, such as control barrier functions [80] and consensus approaches [81], can be used to maintain the platoon. To minimize (3.2) for our long-duration autonomy task, we impose the following assumptions.

Assumption 4. We neglect the effects of wind, and assume the air has constant properties. For vehicle $i \in \mathcal{N} \setminus \{0\}$ the drag force is zero at $F_i\big(0, \hat{p}_i(t)\big)$, increasing in $v_i(t)$, and decreasing in $\hat{p}_i(t)$. For vehicle $i = 0$ the drag force is zero at $F_i\big(0, \hat{p}_i(t)\big)$, increasing in $v_i(t)$, and constant in $\hat{p}_i(t)$.

Assumption 4 is the crux of our analysis, as it determines the signs of the derivatives of the cost function. This assumption is not restrictive, and it can be relaxed if the partial derivatives of F_i can be calculated or measured. Different forms of F_i

will result in different vehicle behavior, and this can be interpreted as a data-driven forcing function. For physical systems containing wind, eddies, and other turbulent effects, F_i may be thought of as an average or filtered drag force; sensing the average aerodynamic forces between vehicles in real time is an active area of ongoing research [82].

Assumption 5. The drag acting on vehicle i is only a function of the states of vehicles i and $i-1$ for $i \in \mathcal{N} \setminus \{0\}$, and there are no external noise or disturbances.

Assumption 6. Communication between CAVs occurs instantaneously and noiselessly.

Assumptions 5 and 6 idealize the environment in which the vehicles are operating to simplify the analysis. Assumption 5 may be relaxed by expanding the drag model to include a time-varying component and additional interaction forces. Likewise, Assumption 6 can be relaxed by including delays, noise, and packet loss in a communication model. If the disturbances and delays are bounded, then Assumption 6 can be relaxed by shrinking the set of feasible actions using standard techniques, e.g., control barrier functions and differential inclusions [83]. However, we believe this adds significant analytical complexity without changing the fundamental results of our analysis.

Assumption 7. Each vehicle $i \in \mathcal{N}(t)$ is equipped with a low-level controller that can track the desired acceleration, $a_i(t)$, by controlling the forward force applied to the CAV through $u_i(t)$.

Assumption 7 allows us to derive the kinematic motion of each CAV without directly considering the applied drag force. This enables us to generate an analytic closed-form optimal trajectory for each vehicle without the numerical challenges associated with boundary-layer fluid dynamics. This assumption can be relaxed by considering robust tracking, e.g., control barrier functions, or online learning to estimate and compensate for the aerodynamic interactions.

3.1.3 Optimal Control with Gradient Flow

We employ *gradient flow* to generate the control input for each vehicle. This a gradient-based optimization technique, wherein each vehicle's control action is a gradient descent step. This technique has been used successfully to control multi-agent constraint-driven systems [75, 84]. Our motivation for gradient flow is twofold: first, planning a trajectory through a fluid boundary layer in real time requires significantly more computational power than what is available to a CAV. Second, the exit time of the preceding vehicle is an unknown quantity, and so each vehicle cannot quantify the trade-off between accelerating to draft the preceding vehicle versus the energy savings of drafting. Thus, we take a conservative approach where no vehicle will increase its energy consumption while traveling on the highway. This approach yields conditions for *when* platooning is an appropriate strategy in addition to *how* the platoon should be formed.

As a first step, we define the set of *safe control inputs* and show that it satisfies recursive feasibility [85]. For the remainder of the analysis, we omit the explicit dependence of state variables on t when no ambiguity arises.

Definition 9. For each vehicle $i \in \mathcal{N} \setminus \{0\}$, the set of safe control inputs is

$$\begin{aligned} \mathcal{A}_i^s(v_i, \hat{p}_i, \hat{v}_i) = \Big\{ a \in \mathbb{R} \ : \ & a_{\min} \leq a \leq a_{\max}, \\ & v_i = v_{\max} \implies a \leq 0 \\ & v_i = v_{\min} \implies a \geq 0 \\ & g_i^s = 0 \implies \frac{d}{dt} g_i^s \leq 0 \Big\}, \end{aligned} \tag{3.11}$$

where g_i^s is the rear-end safety constraint (3.9), and $\frac{d}{dt} g_i^s \leq 0$ can be achieved through the control action, $a_i(t)$. The safe set ensures the state, control, and safety constraints of vehicle i are always satisfied.

Theorem 4. (Recursive Feasibility) For any vehicle $i \in \mathcal{N} \setminus \{0\}$, if the variables

$\hat{p}_i(t), \hat{v}_i(t), v_i(t)$ satisfy (3.3) and (3.9) at time $t_1 \in \mathbb{R}$, then the set $\mathcal{A}_i^s$ is non-empty for all $t \geq t_1$.

Proof. To prove Theorem 4, we show that a feasible control input always exists in the worst case scenario for vehicle i. Let $\hat{v}_i(t_0) \geq 0$ and $a_{i-1}(t) = a_{\min}$ for $t \in [t_0, t_1)$ such that $v_{i-1}(t_1) = v_{\min}$ and $a_{i-1}(t) = 0$ for $t \geq t_1$. We take the time derivative of (3.9), which yields

$$\dot{\hat{v}}_i(t) + \hat{v}_i(t) \cdot \Big(- \frac{a_i(t)}{a_{\min}} \Big) + \hat{a}_i(t) \Big(\frac{v_{\min} - v_i(t)}{a_{\min}} \Big) + \Big(\frac{\hat{v}_i(t)\, \hat{a}_i(t)}{a_{\min}} \Big). \tag{3.12}$$

Over the interval $[t_0, t_1)$, $v_i(t) > v_{\min}$, and thus $a_i(t) = a_{\min}$ is a feasible control action. This implies that $\hat{a}_i(t) = 0$, and evaluating (3.12) implies

$$\dot{\hat{v}}_i(t) + \hat{v}_i(t) \Big(- 1 \Big) = 0, \tag{3.13}$$

i.e., (3.12) is identically zero, which implies that (3.9) is constant. Next, consider the interval $[t_1, t_2)$ such that $a_i(t) = a_{\min}$ for $t \in [t_1, t_2)$ and $v_i(t) = v_{\min}$ for $t \geq t_2$. Thus, $a_i(t) = a_{\min}$ is a feasible control action, and evaluating (3.12) implies

$$\begin{aligned} \dot{\hat{v}}_i(t) - \hat{v}_i(t) + v_{\min} - v_i(t) + \hat{v}_i(t) &= 2\hat{v}_i(t) - 2\hat{v}_i(t) \\ &= 0, \end{aligned} \tag{3.14}$$

which is identically zero over the entire interval. This implies that (3.9) is constant. Finally, for $t > t_2$, $a_i(t) = 0$ is a feasible control action, which implies that $\hat{a}_i(t) = 0$, $\hat{v}_i(t) = 0$, and (3.9) is constant. Therefore, (3.9) is constant for all $t > t_1$ in the worst-case scenario and $\mathcal{A}_i^s \neq \emptyset$ for all $t \geq t_1$. □

Next, before deriving our energy-minimizing constraint, we present the unique equilibrium point that minimizes the energy consumption of each CAV $i \in \mathcal{N}(t)$. For

the lead vehicle, i.e., $i = 0$, the drag force is minimized at $v_0 = 0$ and increasing in v_0 by Assumption 4. It is trivial to show that the lead vehicle's energy consumption is minimized at $v_0(t) = v_{\min}$. For a following vehicle, i.e., $i > 0$, the Karush-Kuhn-Tucker (KKT) conditions yield

$$L = F^2 + \mu^v(v_{\min} - v_i) + \mu^p(\hat{p}_i + \delta), \tag{3.15}$$

$$\frac{\partial L}{\partial v_i} = 2FF_v - \mu^v = 0, \tag{3.16}$$

$$\frac{\partial L}{\partial \hat{p}_i} = 2FF_{\hat{p}} + \mu^p = 0, \tag{3.17}$$

$$\frac{\partial L}{\partial \mu^v} = v_{\min} - v_i = 0, \tag{3.18}$$

$$\frac{\partial L}{\partial \mu^p} = \hat{p}_i + \delta = 0, \tag{3.19}$$

where the subscripts v, $\hat{p}$ refer to the partial derivative of F with respect to $v_i(t)$ and $\hat{p}_i(t)$, respectively, and $\hat{p}_i(t) + \delta = 0$ is implied by (3.9) when $v_i(t) = v_{\min}$. Given Assumption 4, we can determine the signs of the partial derivatives, which implies

$$v_i = v_{\min}, \qquad \hat{p}_i = -\delta, \tag{3.20}$$

$$\mu^v = 2F\frac{\partial F}{\partial v_i} > 0, \qquad \mu^p = -2F\frac{\partial F}{\partial \hat{p}_i} > 0. \tag{3.21}$$

Thus, CAV $i > 0$ minimizes its energy consumption by following CAV $i - 1$ at speed $v_{\min}$ and distance $\hat{p}_i(t) = \delta$. Note that, as F_i is strictly increasing in v_i and strictly decreasing in $\hat{p}_i$, thus the platooning formation corresponds to the unique minimum-energy configuration of the N CAVs.

Finally, to minimize the drag force imposed on each vehicle, we implement gradient flow by requiring the time derivative of the cost functional (3.2) to be negative semidefinite for each vehicle $i \in \mathcal{N}(t)$. For vehicle $i = 0$ this implies

$$\dot{J}_i = \frac{\partial F\big(v_i(t), \hat{p}_i(t)\big)}{\partial v_i(t)} a_i(t) \leq 0, \tag{3.22}$$

which, by Assumption 4, implies that

$$a_i(t) \leq 0 \quad \text{for } i = 0. \tag{3.23}$$

For vehicle $i > 0$, Assumption 4 implies

$$\dot{J}_i = \begin{bmatrix} (F\ F_v) \\ (F\ F_{\hat{p}}) \end{bmatrix} \cdot \begin{bmatrix} a_i(t) \\ \hat{v}_i(t) \end{bmatrix} \leq 0. \tag{3.24}$$

Expanding (3.24) yields

$$F\, F_v\, a_i(t) + F\, F_{\hat{p}}\, \hat{v}_i(t) \leq 0, \tag{3.25}$$

which can be solved for $a_i(t)$ using the signs of the partial derivatives imposed by Assumption 4,

$$a_i(t) \leq \frac{|F_{\hat{p}}|}{F_v} \hat{v}_i(t). \tag{3.26}$$

Thus, in order for CAV i to form a platoon with $i-1$, we must have $\hat{v}_i(t) > 0$. Intuitively this makes sense, if $\hat{v}_i(t) < 0$ then $\hat{p}_i(t)$ is decreasing (increasing the drag force) and i must decelerate to achieve an equivalent decrease in the drag force. Likewise, $\hat{v}_i(t) > 0$ implies that $\hat{p}_i(t)$ is increasing (decreasing the drag force) and i may accelerate without increasing the overall drag force.

Note that, consistent with multi-agent control barrier functions [75], it is possible that imposing (3.26) and the set of safe control inputs (Definition 9) on each vehicle admits no feasible solutions. In particular, this occurs when

$$v_i(t) = v_{\min} \text{ and } \frac{|F_{\hat{p}}|}{F_v}\Big(v_{\min} - v_{i-1}(t)\Big) < 0, \tag{3.27}$$

$$a_i(t) \leq \frac{|F_{\hat{p}}|}{F_v} \hat{v}_i(t) < a_{\min}. \tag{3.28}$$

Similar conditions arise when imposing the deadline constraint on CAV i. Taking the time derivative of (3.10) yields,

$$-v_i(t) + v_i(t) - a_i(t) \leq 0, \tag{3.29}$$

which implies i must apply $a_i(t) \geq 0$ when (3.10) is active. It is possible that vehicle i cannot jointly satisfy the deadline, safety, and drag force constraints. In particular, if either of

$$S_i - p_i(t) - (t_i^f - t)\, v_i(t) = 0 \text{ and } \frac{|F_{\hat{p}}|}{F_v}\hat{v}_i(t) < 0, \tag{3.30}$$

$$S_i - p_i(t) - (t_i^f - t)\, v_i(t) = 0 \text{ and } g_i^s = 0, \hat{v}_i(t) > 0, \tag{3.31}$$

is satisfied, then no control action can guarantee drag minimization, safety, and arrival time simultaneously. Thus, if CAV $i > 0$ satisfies (3.27) - (3.30), it must fall back and become the lead CAV of its own platoon. CAV i will re-initialize itself as index 0 of a new platoon, and all following CAVs $j > i$ will be re-initialized as $j - i$. This platoon will operate independently as long as any of (3.27) - (3.30) are satisfied for the vehicle physically ahead of this CAV on the road. The same test may be applied to determine when two platoons ought to merge into a single platoon. Similarly, if (3.31) is satisfied, then CAV i is unable to achieve its deadline without violating rear-end safety. This affords at least 2 possibilities for CAV i, 1) move into a passing lane to overtake the preceding vehicle, or 2) relax the deadline constraint until i becomes the lead CAV of a platoon. Resolving this conflict depends on the geometry of the roadway and application of interest and is beyond the scope of this paper. Thus, (3.27) - (3.31) determine whether platooning is an appropriate strategy for CAV i.

In addition to the above challenges that arise from the task constraint, selecting an energy-minimizing control law that satisfies (3.11) and (3.26) is, in general, insufficient to generate emergent platooning behavior. This fact is demonstrated in [86], which shows that only minimizing energy consumption is not a stable configuration for

selfish energy-minimizing agents. As an illustrative example, consider the case where the initial states of the vehicles are randomly selected from the set of feasible states such that each CAV $i \in \mathcal{N} \setminus \{0\}$ is in the wake of vehicle $i-1$. This implies a transient period for i, where $v_{i-1}(t) > v_{\min}$. To generate a platoon, we would like to have CAV i achieve and maintain $\hat{v}_i(t) > 0$. We can consider two cases, for the first case let $\hat{v}_i(t_i^0) > 0$, then i can maximize its energy savings by selecting $a_i(t) = a_{\min}$. However, this may lead to a situation where $\hat{v}_i(t) = 0$ and $\hat{p}_i(t) + \delta < 0$, i.e., the vehicles do not form a platoon. Thus, vehicle i ought to apply a small, feasible deceleration such that $\hat{v}_i(t) > 0$ is maintained. In the second case let $\hat{v}_i(t_i^0) \leq 0$, then i ought to decelerate as little as possible, i.e., (3.26) should be a strict equality. Then, if CAV $i-1$ applies a large deceleration, it is possible that $\hat{v}_i(t) > 0$ in the future, and i will be able to join the platoon. The solution of the following optimization problem can accomplish this behavior.

Problem 3. For each CAV $i \in \mathcal{N} \setminus \{0\}$, such that (3.27) and (3.28) are not satisfied, generate the control action that solves

$$\min_{a_i(t)} \ \frac{1}{2} a_i(t)^2$$

subject to:

$$\begin{aligned}
&a_i(t) \in \mathcal{A}_i^s\big(v_i(t), \hat{p}_i(t), \hat{v}_i(t)\big), \\
&a_i(t) \leq \frac{|F_{\hat{p}}|}{F_v} \hat{v}_i(t), \\
&\big(S_i - p_i(t)\big) - v_i(t)\,(t_i^f - t) = 0 \implies a_i(t) \geq 0.
\end{aligned}$$

Note that each vehicle must solve Problem 3 to determine its control input at each time step. In this case, the feasible region is compact, and the solution can be derived offline by determining the upper and lower bound on the feasible space of Problem 3. The optimal solution is the feasible value closest to 0. Next, we present our main results that characterize sufficient conditions for platoon formation.

Lemma 3. For any vehicle $i \in \mathcal{N}(t)$ at any time $t \in \mathbb{R}$, the control action that solves Problem 3 is upper bounded by 0.

Proof. For vehicle $i = 0$, (3.23) implies $a_i(t) \leq 0$.

For vehicle $i > 0$, let $\alpha_1 = \sup\left\{\mathcal{A}_i^s\right\}$, let $\alpha_2 = \frac{|F_p|}{F_v}\hat{v}_i(t)$, and let $\alpha = \min\{\alpha_1, \alpha_2\}$, i.e., α is the smallest upper bound of Problem 3's feasible space. For the case when $\alpha \leq 0$, the solution of Problem 3 is upper bounded by 0. For the case when $\alpha > 0$ the lower bound of Problem 3 is

$$\beta = \begin{cases} a_{\min} & \text{if } v_i(t) \neq v_{\min}, \\ 0 & \text{if } v_i(t) = v_{\min}, \end{cases} \tag{3.32}$$

thus $\beta \leq 0 < \alpha$. This implies that any control action $a_i(t) > 0$ incurs a higher cost than $a_i(t) = 0$, which is a feasible action in this case. Thus, the solution of Problem 3 is always upper bounded by zero. □

Theorem 5. For two CAVs $i, i-1 \in \mathcal{N}(t)$ the initial condition $v_i(t_i^0) > v_{i-1}(t_i^0)$ guarantees that i and $i-1$ will form a platoon as long as t_i^f and t_{i-1}^f are sufficiently large and the deadline constraint for i does not become active.

Proof. First, consider the case when the rear-end safety constraint does not become active. Assume $\hat{v}_i(t_1) < 0$ at some $t_1 > t_i^0$. Continuity of $\hat{v}_i(t)$ implies that there is at least one non-zero interval of time $[t_0, t_1]$ such that $\hat{a}_i(t) < 0$ and $\hat{v}_i(t) \geq 0$ for $t \in [t_0, t_1]$. Over any such interval, $\hat{v}_i(t) \geq 0$ implies that $a_i(t) = 0$ is a feasible control action. Furthermore, Lemma 3 implies $a_{i-1}(t) \leq 0$, which implies that $\hat{a}_i(t) \geq 0$. This contradicts $\hat{a}_i(t) < 0$, therefore no such interval can exist and $\hat{v}_i(t) > 0$ for all $t > t_i^0$ as long as the safety constraint does not become active.

Next, consider the case when only the rear-end safety constraint is active, i.e., (3.9) is strictly equal to zero. In this case, solving (3.9) for $\hat{v}_i(t)$ yields

$$\hat{v}_i(t) = v_i(t) - v_{\min} - \Big[\big(v_i(t) - v_{\min}\big)^2 + 2|a_{\min}|\big(\hat{p}_i(t) + \delta\big)\Big]^{\frac{1}{2}}, \tag{3.33}$$

where $\hat{p}_i(t) + \delta \leq 0$, and thus (3.33) implies $\hat{v}_i(t) \geq 0$ when $\hat{p}_i(t) + \delta < 0$. Thus, $\hat{v}_i(t)$ is positive and decreasing and only reaches zero when $\hat{p}_i(t) + \delta = 0$, i.e., platoon formation occurs. □

Theorem 5 is a sufficient condition for platooning, and can be recursively applied at any time t_0 to guarantee the convergence of any sequence of vehicles satisfying $v_i(t_0) < v_{i+1}(t_0) < \cdots < v_{i+k}(t_0)$ for $k \in \mathbb{N}$. We also note that platooning may occur when $v_i(t_0) > v_{i+1}(t_0)$, in particular if vehicle i decelerates sufficiently fast such that $v_i(t_1) < v_{i+1}(t_1)$ for some $t_1 > t_0$. In this case, Theorem 4 can be applied at $t = t_1$ to guarantee platoon formation.

Finally, the behavior of the front CAV $i = 0$ depends on the context of the platooning problem. The lead CAV may select any trajectory satisfying $a_i(t) \leq 0$ and $v_i(t) \geq v_{\min}$ under our framework. For example, following $u_i(t) = 0$ could minimize transient energy operation while the drag force slows the vehicle down to the minimum speed. Alternatively, to facilitate platoon formation, it may be practical to select $a_i(t) = a_{\min}$ to reach the minimum speed as fast as possible. We apply the latter approach in the next sections to demonstrate emergent platoon formation in a simulated and physical experiment.

3.1.4 Simulation Results

To validate our proposed control approach, we simulated a road 1750 m long with 3 on and off ramps. The on-ramps were located at 100, 600, and 1100 m, and the off-ramps were at 500, 1000, and 1500 m. We simulated the flow of traffic over 140 seconds,

and we introduced vehicles to the system with a random delay $T \sim \mathcal{U}\big(0.5, 1.5\big)$ seconds. For each CAV i, we selected its initial and exit positions from a uniform distribution over the four possible locations, i.e., the three on-ramps and an initial position of of $p_i(t_i^0) = 0$. Similarly, i may exit the highway at a distance of $p_i(t_i^f) = 1750$ or at any off-ramp beyond $p_i(t_i^0)$. After selecting its initial position, we discarded any CAV that could not simultaneously satisfy (3.3), (3.4), and (3.9) for itself and the vehicle behind it. This approach resulted in $N = 136$ vehicles entering the highway over 140 seconds, yielding an average inflow of 3500 vehicles per hour.

We selected the arrival time for each vehicle after determining its feasible initial state. For each CAV i, we drew the arrival time t_i^f from the uniform distribution,

$$t_i^f \sim \mathcal{U}\Big(\frac{S_i - p_i(t_i^0)}{v_i(t_i^0)}, \frac{S_i - p_i(t_i^0)}{v_{\min}}\Big), \tag{3.34}$$

which guaranteed satisfaction of the deadline constraint (3.10) at t_i^0. In the case that CAV i later was unable to achieve its deadline, i.e., (3.30) or (3.31) became active, we relaxed the deadline constraint. In particular, when i satisfied (3.30) or (3.31) we removed the deadline constraint from Problem 3 for i. If i later became the leader of a platoon, we relaxed the drag minimization constraint (3.26) and required i to accelerate until the deadline constraint (3.10) was satisfied. This achieved a balance between energy-minimization and deadline satisfaction while guaranteeing safety, and it circumvented the additional challenges of overtaking in a multi-lane highway environment.

The vehicle trajectories are presented in Figs. 3.1 and 3.2, which show the dynamic formation and break-up of platoons as vehicles enter and exit the system over two 60 second windows of the simulation. Figs. 3.3 show a zoomed in region of Fig. 3.1 where vehicles entering at the 100 m on-ramp form a platoon at approximately 175 m. For further supplemental diagrams and videos see: https://sites.google.com/view/ud-ids-lab/cdp.

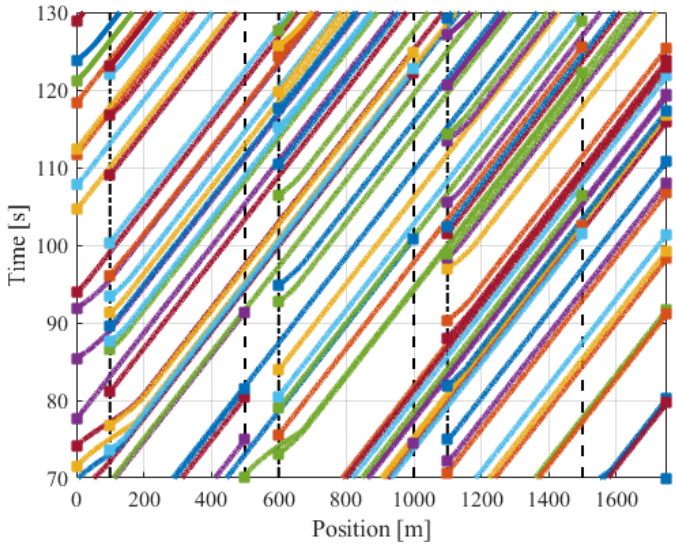

Figure 3.1: Position vs time plot for the $N = 136$ CAVs over a 60 second window of steady operation. Squares correspond to vehicles entering and exiting the roadway; dash-dot lines correspond to on-ramps and dotted lines correspond to off-ramps.

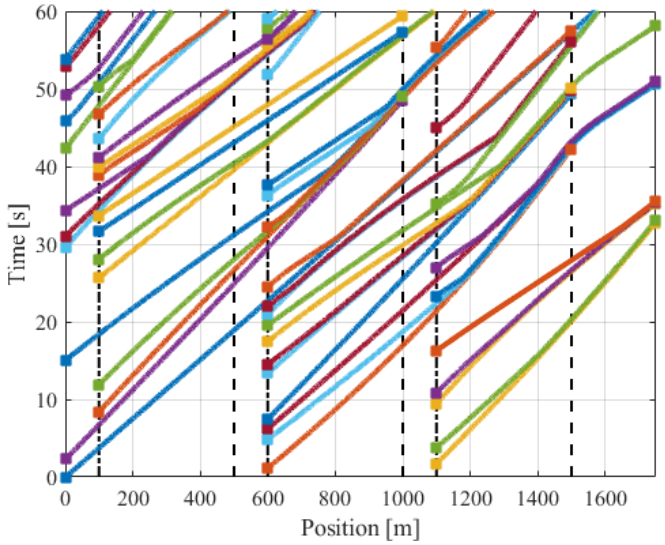

Figure 3.2: Position vs time plot for the $N = 136$ CAVs over the initial 60 second transient. Squares correspond to vehicles entering and exiting the roadway; dash-dot lines correspond to on-ramps and dotted lines correspond to off-ramps.

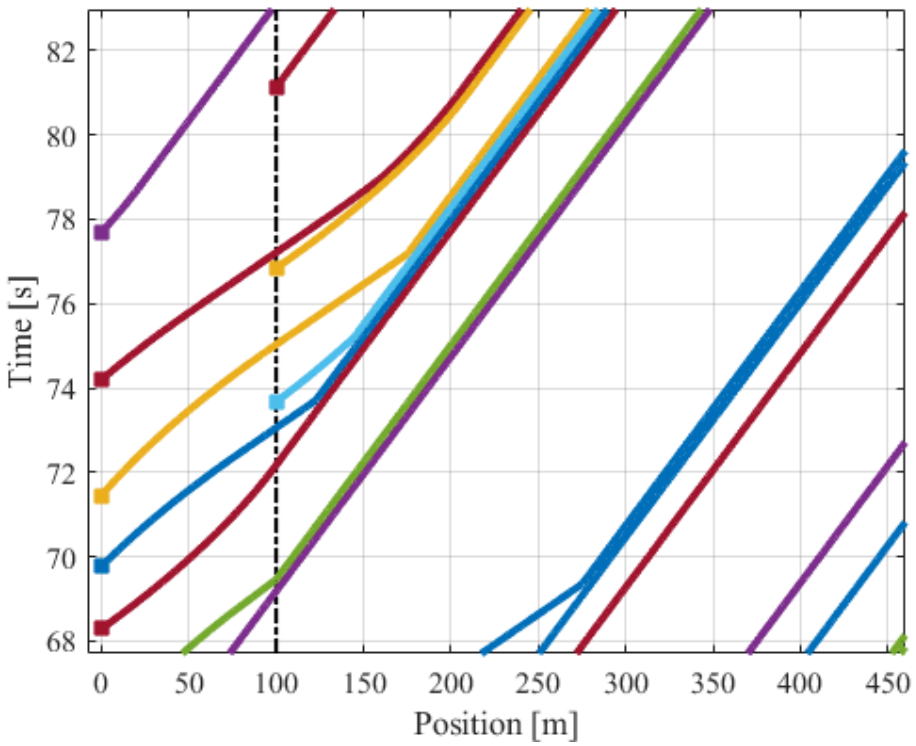

Figure 3.3: A close up where 4 vehicles form a platoon near the on-ramp at 100 m.

3.2 A Constraint-Driven Approach to Line Flocking: The V Formation as an Energy-Saving Strategy

3.2.1 Introduction

Generating emergent flocking behavior has been of particular interest since Reynolds proposed three heuristic rules for multi-agent flocking in computer animation [11]. In aerial systems, the main energy savings comes from upwash, i.e., trailing regions of upward momentum in the slipstream, which followers exploit to reduce induced drag and energy consumption. Flocking to minimize energy consumption is known as line flocking in the engineering literature [9], and it is named on the biological behavior of geese, pelicans, etc [13].

The simplest method to achieve a V formation may be to generate an optimal set of formation points based on the aerodynamic characteristics of each agent. This effectively transforms the line flocking problem into a formation reconfiguration problem, where each agent must assign itself to a unique goal and reach it within some fixed time, as is the case in [30]. The physical effects of V formation flight were explored in a recent article [33], where the authors demonstrate that the leading and trailing agents consume energy at a significantly higher rate. This implies that these agents are the limiting factor in the total distance traveled, and the authors propose a formation-reconfiguration scheme based on a load-balancing protocol. However, a formation reconfiguration approach generally requires the formation to be computed offline, and while some articles consider agent heterogeneity (e.g., age, weight, size, and efficiency) [87], this has not yet been explicitly considered in a flocking problem. Furthermore, the formation points must be recalculated online if an agent enters or leaves the system, or if there are significant changes in the ambient environment.

An alternative approach, line flocking is a data-driven control technique that involves measuring (or modeling) the aerodynamic and hydrodynamic interactions between agents so that they may dynamically position themselves to save energy. This is

achieved for aerial vehicles in $\mathbb{R}^2$ using a model predictive control approach [32]. The authors construct a multi-objective optimization problem that minimizes speed differences, maximizes upwash benefit, and minimizes the field of view occlusion between agents. This work demonstrates that solving this multi-objective optimization problem yields emergent V formations, even when the agents are initialized randomly. A recent review of related optimal flocking techniques is presented in [9].

Our approach, in contrast to existing work, is constraint-driven. In our framework, agents seek to travel as efficiently as possible subject to a set of task and safety constraints. This set-theoretic approach to control is interpretable, i.e., the cause of an agent's action can be deduced by examining which constraints become active during operation. By examining the conditions that lead to an empty feasible space, our framework also addresses the problem of constraint compatibility, i.e., how each agent ought to behave when the intersection of all its constraints is the empty set. Our approach is totally decentralized, and thus it is well-suited to "open systems," where agents may suddenly enter, leave, or experience failure.

In this work we describe *anseroids* (anserine-oid, meaning 'goose-like') that generate dynamic echelon and V formations without any knowledge of the total number of agents in the system, and which are not given any information about the desired formation shape. To the best of our knowledge, the only result similar to ours is [32], which uses particle swarm optimization combined with model predictive control to solve an optimal control problem. However, their approach depends on a multi-objective optimization problem with four components, and they provide no guarantees on the emergence of flocking behavior. This work was also explored in a reinforcement learning context in [88], which provides conditions for V formations not to occur.

Our contributions are as follows: (1) The first (to the best of our knowledge) optimal control algorithm that guarantees emergent V formations as a means of energy savings, (2) a physics-based flocking model where agent decisions are driven by the environment, (3) an interpretable switching system architecture that describes the

optimal behavior of each UAV, and (4) compelling that shows energy savings is an increasing function of flock diversity.

The remainder of the section is organized as follows. In Section 3.2.2, we discuss our notation and present the dynamics of our problem. Next, we present our optimal control problem and guarantees on its behavior in Section 3.2.3. Finally, in Section 3.2.4, we validate our results by simulating 4 UAVs in a 2D plane, where the vehicles are initialized to form a vertical line.

3.2.2 Problem Formulation

3.2.2.1 Note on Notation

Most references on optimal control, e.g., [89, 90], consider centralized problems. Thus, directly adopting their notation may lead to ambiguities about the state space of a decentralized problem. To relieve this tension, we take the following approach for an agent with index i: endogenous variables, e.g., the position of agent i, are written without an explicit dependence on time, while exogenous variables, e.g., the position of agent j as measured by agent i, are written with an explicit dependence on time. This notation is common in the applied mathematics literature [91], and makes it explicitly clear how functions evolve with respect to the state (e.g., state dynamics) and how they evolve with respect to time (e.g., external signals measured by the agent).

3.2.2.2 System Dynamics

We consider a fleet of $N \in \mathbb{N}$ fixed-wing uncrewed aerial vehicles (UAVs) indexed by the set $\mathcal{A} = \{1, 2, \ldots, N\}$. We denote the state of each UAV $i \in \mathcal{A}$ by

$$\boldsymbol{x}_i \coloneqq \begin{bmatrix} \boldsymbol{p}_i \\ \theta_i \end{bmatrix}, \tag{3.35}$$

where $\boldsymbol{p}_i \in \mathbb{R}^2$ is the UAV's position and $\theta_i \in \mathbb{R}$ is the UAV's velocity and heading angle. Each UAV obeys unicycle dynamics,

$$\begin{aligned} \dot{\boldsymbol{p}}_i &= \begin{bmatrix} v_i \cos\theta \\ v_i \sin\theta \end{bmatrix}, \\ \dot{\theta}_i &= \omega_i, \end{aligned} \tag{3.36}$$

where $v_i \in \mathbb{R}_{>0}$ and $\omega_i \in \mathbb{R}$ are the linear and angular speed of UAV i. We impose the actuation constraints,

$$\begin{aligned} |\omega_i| &\leq \omega_{\max}, \\ 0 < v_{\min} \leq v_i &\leq v_{\max}, \end{aligned} \tag{3.37}$$

where $\omega_{\max}$ is the maximum turning rate and $v_{\min} < v_{\max}$ correspond to the minimum and maximum air speed.

Finally, the total drag force acting on UAV i has the form,

$$F_i(\boldsymbol{x}_i, t) = C_1 v_i^2 + \frac{C_2}{v_i^2} - \frac{L}{v_i} W(\boldsymbol{p}_i, t) \tag{3.38}$$

where $C_1, C_2 \in \mathbb{R}_{>0}$ capture the profile and self-induced drag that include the drag coefficient, air density, and wing area. The function $W : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}$ describes the scalar upwash field, which we formally define in Section 3.2.3. We impose the following assumptions on our system.

Assumption 8. Each UAV is equipped with a low-level flight controller that can track the sequence of control actions.

Assumption 9. The UAVs are operating in still air with constant aerodynamic properties.

We employ Assumptions 8 and 9 to determine the behavior of the system in idealized conditions, and they are not restrictive on our analysis. In particular, applying adaptive and robust control techniques, such as control barrier functions [75] or

Gaussian Processes [92], can be used to overcome the resulting model mismatch.

Assumption 10. Collision avoidance between UAVs in $\mathcal{A}$ can be neglected.

Generally, vee formations have significant space between individuals without opportunities for collisions between agents [33, 87]. Thus, we impose Assumption 10 to focus the scope of our work on the emergence of the vee formation. Furthermore, aerodynamic effects disincentive UAVs from approaching too closely, and collision avoidance can always be guaranteed by introducing pairwise collision avoidance constraints [59].

Assumption 11. There exists a global heading angle θ_g and a small tolerance $\epsilon \in \mathbb{R}_{>0}$ such that $|\theta_i - \theta_g| < \epsilon$ for all UAVs $i \in \mathcal{A}$.

We impose Assumption 11 to simplify our analysis of the aerodynamics. First, it allows us to consider spanwise cuts of the domain, which reduces our analysis from 2D to 1D. Second, it allows us to model the wake as a scalar field centered on each UAV instead of modeling the wake evolution numerically, i.e., using computational fluid dynamics. This assumption is common in the multi-UAV literature [32, 33], although it is usually not stated explicitly. We impose this assumption as a constraint in our final control algorithm, and it can be interpreted as the "migratory urge" proposed by Reynolds [11]; the direction θ_g could also be computed using consensus, and some agents could simply separate themselves from the flock if Assumption 11 becomes too restrictive.

3.2.2.3 Wake Model

Under Assumption 11, we model the wake of each UAV $i \in \mathcal{A}$, as a scalar field centered at $\boldsymbol{p}_i$ and aligned with θ_i. Physically, the upwash field is a consequence of the pressure difference between the top and bottom of the wing [93]. This induces a vortex at the wing tips, which generates an upward velocity (i.e., upwash) far from the wing and downward velocity (i.e., downwash) behind the wing. Classically the wingtip vortices have been modeled using irrotational flow [33, 93]. However, this model is

known to cause nonphysical behavior at the wing tips, where the vertical air speed approaches infinity. In this work we augment the irrotational vortex model with a rotational core, which drives the velocity to zero at the wing tips. Namely, each vortex induces the upwash velocity,

$$u_i(r) = \begin{cases} \frac{\Gamma}{2\pi r} & \text{if } r \geq r^*, \\ \Omega r & \text{if } r \leq r^*, \end{cases} \tag{3.39}$$

where u_i is the vertical airspeed, r is the distance to the vortex center, Γ is the circulation of the irrotational vortex, Ω is the angular rotation speed of the rotational core, and

$$r^* = \left(\frac{\Gamma}{2\pi\Omega}\right)^{\frac{1}{2}}. \tag{3.40}$$

Note that under Assumption 11, the induced velocity field has the form,

$$f(y) = u_i(y - b) - u_i(y + b), \tag{3.41}$$

where y is a relative spanwise position and $2b$ is the wingspan. The upwash distribution is shown in Fig. 3.4.

In the streamwise, i.e., longitudinal direction, the wing interactions quickly coalesce into the two strong wingtip vortices. As the vortices move aft from the wing, their influence approximately doubles. Modeling the streamwise behavior of the vortices is a challenging problem; we adopt the approach of [32] using a gaussian function,

$$g(x) = 2\exp -\frac{(x-\mu)^2}{2\sigma^2}, \tag{3.42}$$

where μ determines the location of maximum upwash benefit, σ determines the wake of the length, x is the relative streamwise position,. Finally, we define the relative distance vector,

$$\boldsymbol{s}_{ij}(\boldsymbol{p}_i, t) = \boldsymbol{p}_i - \boldsymbol{p}_j(t), \tag{3.43}$$

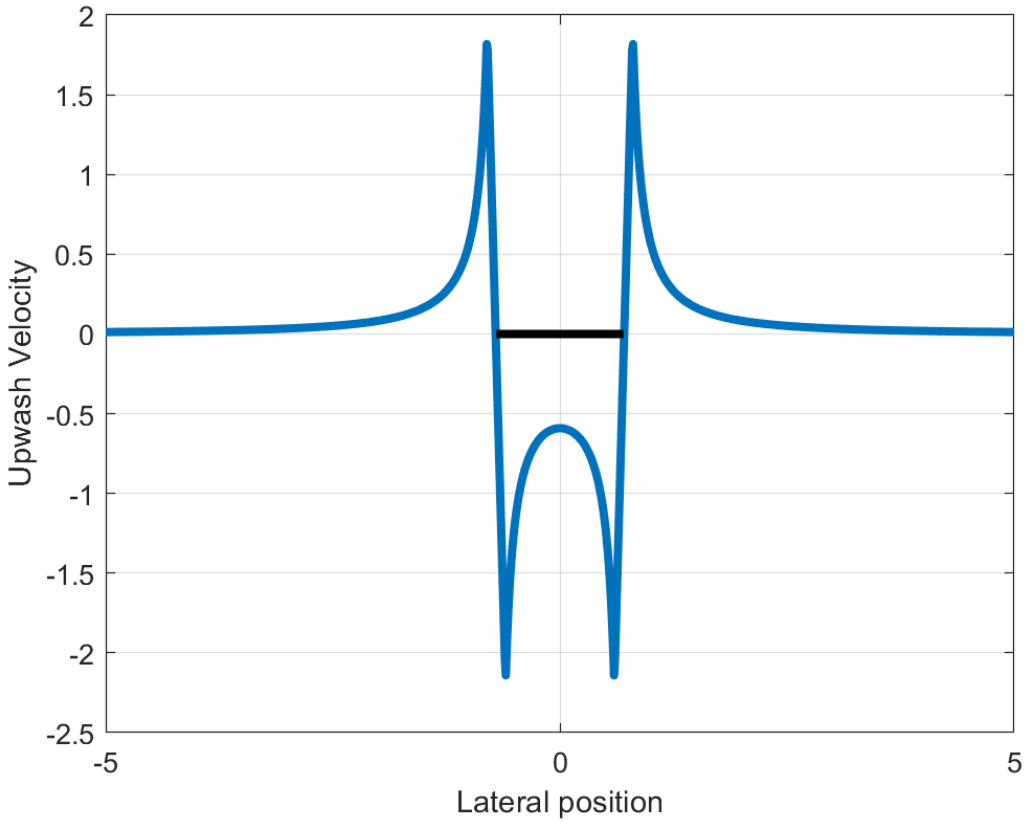

Figure 3.4: Upwash velocity induced in the spanwise direction due to the wing tip vortices.

and combine (3.41) and (3.42) yields an expression for the magnitude of UAV j's upwash on i,

$$w_i^j(\boldsymbol{p}_i, t) \coloneqq f\Big(\boldsymbol{s}_{ij}(\boldsymbol{p}_i, t) \cdot \hat{y}\Big) \cdot g\Big(\boldsymbol{s}_{ij}(\boldsymbol{p}_i, t) \cdot \hat{x}\Big), \tag{3.44}$$

where $\hat{x}, \hat{y}$ are unit vectors aligned with and perpendicular to θ_g, respectively.

3.2.3 Optimal Feedback Controller

We employ *gradient flow* to generate the control input for each vehicle. This a gradient-based optimization technique, wherein each vehicle's control action is a gradient descent step, and this technique has been used successfully to control multi-agent constraint-driven systems [46, 75, 84]. Our motivation for gradient flow is twofold:

first, it enables the UAVs to immediately react to their surroundings without the computational and communication costs associated with decentralized trajectory planning [9, 47]. Second, it allows the UAVs to operate in an open system, i.e., it allows UAVs to be arbitrarily added and removed to the domain without a priori knowledge. We impose the following pairwise assumption to simplify our analysis, and derive rigorous guarantees on the UAVs behavior.

Assumption 12. For each UAV $i \in \mathcal{A}$, there is at most one UAV $j \neq i$ such that the upwash force $w_i^j(\boldsymbol{p}_i, t)$ is not negligible.

Intuitively, Assumption 12 requires the UAVs to be sufficiently 'close' to a V or echelon formation. This makes our analysis tractable, as each UAV must only consider the influence of its immediate leader. We extend the results of this section to the general case in Section 3.2.4, where the UAVs are simulated in 2D space, where we do not force 12 to hold.

First, we seek the control input v_i that minimizes the drag force on UAV $i \in \mathcal{A}$, i.e.,

$$\frac{\partial F_i}{\partial v_i} = 2C_1 v_i - 2\frac{C_2}{v_i^3} + \frac{L}{v_i^2} W_i(\boldsymbol{p}_i, t) = 0. \tag{3.45}$$

Rearranging terms and multiplying by v_i^3 yields,

$$v_i^4 + \frac{L}{2C_1} W_i(\boldsymbol{p}_i, t) v_i - \frac{C_2}{C_1} = 0. \tag{3.46}$$

Remark 2. Note that (3.46) minimizes the drag experienced by UAV $i \in \mathcal{A}$, which generally maximizes the distance traveled by the UAV per unit of energy expended. Alternatively, one could minimize the power lost to drag by considering the product of (3.38) and v_i; this minimizes the power lost to drag, which generally maximizes the flight time of the UAV. The following analysis holds for both cases.

Lemma 4. There is a unique real positive speed that minimizes the drag experienced by each UAV $i \in \mathcal{A}$.

Proof. The optimal airspeed for UAV i is the solution to (3.46), a quadratic function of v_i with the discriminant

$$\Delta_4 = -\left(256\frac{C_2}{C_1} + 27\Big(-\frac{L}{2C_1}W_i(\boldsymbol{p}_i,t)\Big)^4\right) > 0, \tag{3.47}$$

which implies that (3.46) has two complex conjugate roots and two real roots. The imaginary roots satisfy,

$$v_i^2 + bv_i + c = 0, \quad b^2 - 4c < 0, \tag{3.48}$$

and thus $c > 0$. Next, polynomial long division of (3.46) on (3.48) yields a quadratic form for the real roots and additional conditions on a and b, i.e.,

$$v_i^2 - bv + b^2 - c = 0, \tag{3.49}$$

$$2bc + \frac{L}{2C_1}W_i(\boldsymbol{p}_i,t) - b^3 = 0, \tag{3.50}$$

$$c^2 - b^2c - \frac{C_2}{C_1} = 0. \tag{3.51}$$

Condition (3.51) implies,

$$c(c-b^2) = \frac{C_2}{C_1} > 0, \tag{3.52}$$

which implies $c > b^2$. Finally, applying the quadratic equation to (3.49) yields,

$$v_i = \frac{b \pm \sqrt{-3b^2 + 4c}}{2}. \tag{3.53}$$

Multiplying the two real roots yields,

$$\frac{1}{4}\Big(4b^2 - 4c\Big) = b^2 - c < 0. \tag{3.54}$$

Thus, the two real solutions to v_i have opposite signs, and (3.46) has exactly one real positive solution. □

Note that following the proof of Lemma 4 it is possible to derive the optimal airspeed analytically, but that is beyond the scope of this paper. Our next result characterizes how the upwash benefit affects the optimal airspeed of each UAV.

Lemma 5. The optimal airspeed of UAV i decreases when gaining an upwash benefit and increases when experiencing an upwash cost.

Proof. Consider a UAV $i \in \mathcal{A}$ flying in isolation. In this case $W(\boldsymbol{p}_i, t) = 0$, and the optimal airspeed arises when (3.45) is satisfied, i.e.,

$$v_i^* = \left(\frac{C_2}{C_1}\right)^{1/4}. \tag{3.55}$$

Substituting this into (3.45) and rearranging terms implies,

$$v_i^4 - (v_i^*)^4 = -\frac{L}{2C_1} v_i W_i(\boldsymbol{p}_i, t). \tag{3.56}$$

Thus, as $v_i > 0$ from Lemma 4, $W_i < 0$ (upwash cost) implies $v_i > v_i^*$ and $W_i > 0$ (upwash benefit) implies $v_i < v_i^*$. □

Lemma 6. Under Assumptions 11 and 12, a flock of $N > 2$ agents flying at their optimal airspeed is unstable if their upwash benefit does not change sign; after finite time, the upwash benefit of each UAV approaches zero.

Proof. Let every UAV $i \in \mathcal{A}$ fly at their optimal airspeed v_i^*, i.e., the speed that minimizes (3.38). Without loss of generality, consider some UAV i satisfying $W(\boldsymbol{p}_i, t) > 0$. Under Assumption 12 there exists exactly one UAV j such that $w_i^j(\boldsymbol{p}_i, t) > 0$. By definition, $w_i^j(\boldsymbol{p}_i, t) << w_j^i(\boldsymbol{p}_i, t)$ due to the asymmetry of the upwash field. Thus, either $W(\boldsymbol{p}_i, t) > W(\boldsymbol{p}_j, t)$, or there exists some UAV $k \in \mathcal{A}$ such that $w_j^k(\boldsymbol{p}_k, t) = w_i^j(\boldsymbol{p}_i, t)$. Repeating the above process for a finite swarm size of $N > 2$ eventually yields UAV $f \in \mathcal{A}$, the front agent that recieves a reduced upwash benefit. Thus, $v_f^* > v_i^*$, where v_f^* is increasing, by Lemma 5; thus UAV f will satisfy $W(\boldsymbol{p}_f, t) = 0$ in finite time, and this result holds for all UAVs in the system. □

Theorem 6. Under Assumption 11, flying at the optimal air speed never leads to an energy-saving emergent V formation for $N > 2$ UAVs.

Proof. Theorem 6 follows from Lemma 6 in the case that UAVs cannot change the sign of their upwash benefit. The UAVs could maintain a V formation if they maintain a lateral spacing of b, however, this configuration is unstable by Lemma 6. Thus, under Assumption 12, the UAVs can only maintain this configuration by chattering around the unstable equilibrium point—yielding no benefits in terms of energy savings. □

Theorem 6 demonstrates that simply flying at the energy-optimal airspeed can never lead to emergent line flocking! This highlights a significant shortcoming within applying the robot ecology approach [94] to line flocking. Specifically, if each UAV $i \in \mathcal{A}$ minimizes its acceleration subject to a constraint that matches v_i^* as closely as possible, then an energy-saving V formation can not occur. The same result holds if i is constrained to maximize its upwash benefit [86] instead. Therefore, rather than minimizing the "energy" spent to actuate by minimizing acceleration, we propose that each agent ought to minimize its "locomotive power" expended through the cost function,

$$J(\boldsymbol{x}_i) = \left(\frac{v_i - v_i^*}{v_{\max} - v_{\min}}\right)^2 + \left(\frac{\omega_i}{\omega_{\max}}\right)^2, \tag{3.57}$$

where v_i^* is the unique optimal airspeed (Lemma 4) and both cost components are normalized. Employing the cost (**??**) is a subtle change, but it ends up playing a critical role in the generation and stabilization of emergent V formations.

Next, we estimate the wake interaction of UAV j on $i \in \mathcal{A}$ using our simplified aerodynamic model. To simplify our notation, we use the scalars x_i and y_i denote the relative position of i with respect to j in the streamwise and spanwise directions, respectively. First, we estimate how UAV i tends to roll due to the local flow field by

evaluating the integral,

$$m_i(x_i, y_i, t) = g(x_i) \int_{y_i - b}^{y_i + b} (\xi - y_i) f(\xi) d\xi. \tag{3.58}$$

Similarly, we can estimate the lift induced on the wing through momentum transfer,

$$w_i(x_i, y_i, t) = g(x_i) \int_{y_i - b}^{y_i + b} f(\xi) d\xi. \tag{3.59}$$

Note that both (3.58) and (3.59) both have analytical closed-form solutions. The irrotational flow far from the wing-tip is integrable, and it transitions to an affine function near the wing tip. Taking the sum of (3.59) over all UAVs determines the aggregate upwash effect on i,

$$\begin{aligned} W_i(\boldsymbol{p}_i, t) &= \sum_{k \in \mathcal{A} \setminus \{i\}} w_k(\boldsymbol{p}_i, t), \\ M_i(\boldsymbol{p}_i, t) &= \sum_{k \in \mathcal{A} \setminus \{i\}} m_k(\boldsymbol{p}_i, t), \end{aligned} \tag{3.60}$$

where $\boldsymbol{p}_i$ must be projected onto the streamwise and spanwise components of UAV k to yield x_i and y_i. Note that (3.60) should be multiplied by a constant to compute the upwash force in the correct units. However, as we only consider the sign of the derivative of each term, any positive scaling factor is irrelevant to our analysis.

Next, we estimate the cost required for UAV i to maintain a constant altitude and orientation in the presence of the upwash field. In particular, the UAV must expend energy to counter-roll against a non-zero moment, and it must pitch upward to counteract a negative upwash. This leads to an intuitive physics-based description of the cost to flock,

$$E_i(p_i, t) = \kappa |M_i(p_i, t)| - F_i(p_i, t), \tag{3.61}$$

where κ is a system parameter that captures the tradeoff between the cost to roll and

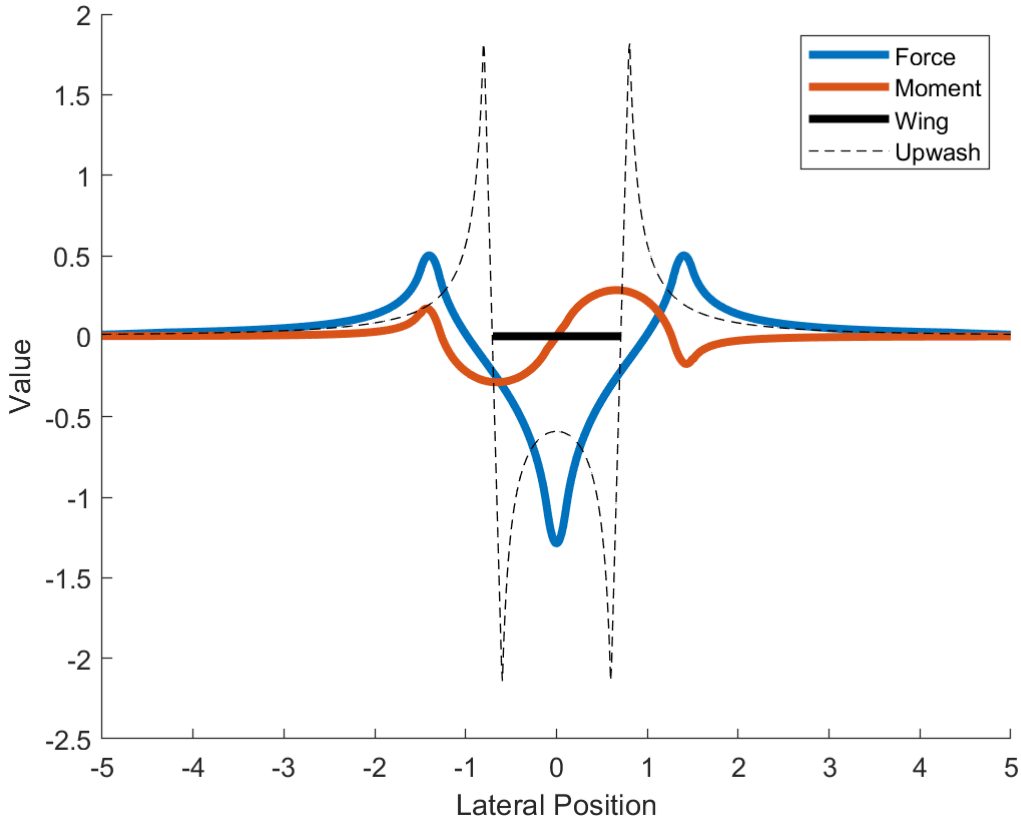

Figure 3.5: Upwash force and moment curves calculated by integrating the upwash velocity field along the wingspan at each point in the domain.

the cost to pitch upward. Finally, we require each UAV to satisfy,

$$\dot{E}_i(\boldsymbol{p}_i, t) \leq 0, \tag{3.62}$$

where $E_i(\boldsymbol{p}_i, t)$ has a finite lower bound. Thus, each UAV is driven toward an equilibrium point where the energy lost through wake interactions is minimized, or equivalently, the energy saved by flocking is maximized. Intuitively, the equilibrium points occur where the upwash benefit dominates over the correcting moment. By inspection of Fig. 3.5, this occurs slightly past the wingtips, which agrees with existing literature [33, 87]. Explicitly finding the equilibrium points, and determining the values of κ

that enable emergent flocking behavior, is the subject of an article in preparation and is beyond the scope of this dissertation. Supported by our prior analysis, we propose Problem 4, which each UAV solves to determine its control input at each time instant.

Problem 4. Each UAV $i \in \mathcal{A}$ takes the optimal control input that optimizes,

$$\min_{v_i,\omega_i} \left\{ \left(\frac{v_i - v_i^*}{v_{\max} - v_{\min}} \right)^2 + \left(\frac{\omega_i}{\omega_{\max}} \right)^2 \right\}$$

$$\text{subject to: } (3.36), (3.37),$$

$$\dot{E}(\boldsymbol{p}_i, t) \leq 0$$

$$|\theta_i - \theta_g| \leq \epsilon,$$

where θ_g is the global heading angle (Assumption 11), which is selected a priori by a designer.

It is possible for the feasible region of Problem 4 to become empty. This is intuitive, for example, if a UAV overshoots the upwash peak; it cannot continue forward and decrease $\dot{E}(\boldsymbol{p}_i, t)$, but it cannot travel slower than $v_{\min}$. This is known as the constraint compatibility problem, and it is well-studied in the set-theoretic control community [29, 47, 73, 95]. Generally, the problem of constraint incompatibility has been solved in the ecologically-inspired robotics literature by introducing slack variables [29, 45]. However, this is not fundamentally different from moving the constraint into the objective function to make it 'soft.' A foundational paper in multi-agent control barrier functions proposed operating the system in two modes [95]: a nominal mode where the agents solve the optimal control problem, and a 'safe mode' where the agents come to a stop when the feasible space is empty. We take this approach to its logical conclusion - when the feasible space of Problem 4 is empty, the controller switches modes and solves a relaxed version of the problem. Note that determining the conditions of constraint incompatibility for this system is ongoing work; the constraint incompatibility problem and a derivation of an equivalent switching system is further

discussed in the final section of this chapter, and the switching system for this problem is presented in Fig. 3.6.

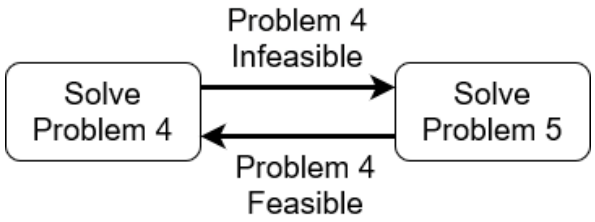

Figure 3.6: The behavior of each UAV visualized as a switching system. The feasible space of Problem 4 determines when the UAV should solve the original or the relaxed optimal control problem.

When Problem 4 has no feasible solution, it is unreasonable to relax the constraints corresponding to the dynamics or control constraints; are options are to either relax $\dot{E}_i < 0$ or $|\theta_i - \theta_g| \leq \epsilon$. We propose that the former should be relaxed to maintain Assumption 11; we present the relaxed optimal control problem with Problem 5, followed by a result that shows such a relaxation only lasts for a finite time interval.

Problem 5. Each UAV $i \in \mathcal{A}$ takes the optimal control input that optimizes,

$$\min_{v_i, \omega_i} \left\{ \left(\frac{v_i - v_i^*}{v_{\max} - v_{\min}} \right)^2 + \left(\frac{\omega_i}{\omega_{\max}} \right)^2 \right\}$$

$$\text{subject to: } (3.36), (3.37),$$

$$|\theta_i - \theta_g| \leq \epsilon,$$

where θ_g is the global heading angle (Assumption 11).

If UAV i has a feasible initial state, it is trivial to show that the control action

$$\begin{aligned} v_i &= \max\left\{ \min\left\{v^*, v_{\max}\right\}, v_{\min}\right\}, \\ \omega_i &= 0, \end{aligned} \tag{3.63}$$

is the optimal solution to Problem 4. Thus, Problem 5 is a 'safe state' that retains some structure of the original problem, i.e., it doesn't require the UAVs to come to a complete stop or travel at $v_{\min}$. Next, we prove that each UAV will only need to solve Problem 5 for a finite amount of time before the constraint can be un-relaxed.

Corollary 1. Any UAV will only solve Problem 5 for a finite interval of time before switching back to Problem 4.

Proof. Corollary 1 follows trivially from Lemma 6; any UAV i will satisfy $W_i = 0$ after a finite amount of time such that $\dot{E} = 0$ for all control actions. □

3.2.3.1 Implementation

We foresee two major issues when solving Problem 4 on a real system. The first is a sensing issue, namely determining the value of M_i and W_i for each UAV $i \in \mathcal{A}$. Our proposed solution is to rely on Assumption 8, i.e., that the UAV is equipped with a low-level tracking controller. At the tracking level, the upwash force and induced roll act as disturbances on UAV. By monitoring the roll and pitch signals, it is possible to infer the upwash force and induced moment. Thus, we propose that the flocking cost E_i and the optimal airspeed v_i^* can be inferred by sampling the low-level control signals, as demonstrated in Fig. 3.7.

The second issue is information-theoretic, namely calculating the time derivative of E_i for each UAV $i \in \mathcal{A}$. The gradient can be implied from periodic measurements, but it also has a time-varying component, i.e.,

$$\dot{E} = \frac{\partial E}{\partial t} + \frac{\partial E}{\partial \boldsymbol{p}_i}\dot{\boldsymbol{p}}_i, \tag{3.64}$$

where the non-smooth points in $\frac{dE}{d\mathbf{p}}$ can be handled with differential inclusions [96].

Each UAV i can estimate $\frac{\partial E}{\partial \boldsymbol{p}_i}$, and $\dot{\boldsymbol{p}}_i$ is known, but i must also have knowledge of v_j to compute $\frac{\partial E_i}{\partial t}$. Similarly, UAV j must have knowledge of v_i to compute $\frac{\partial E_j}{\partial t}$;

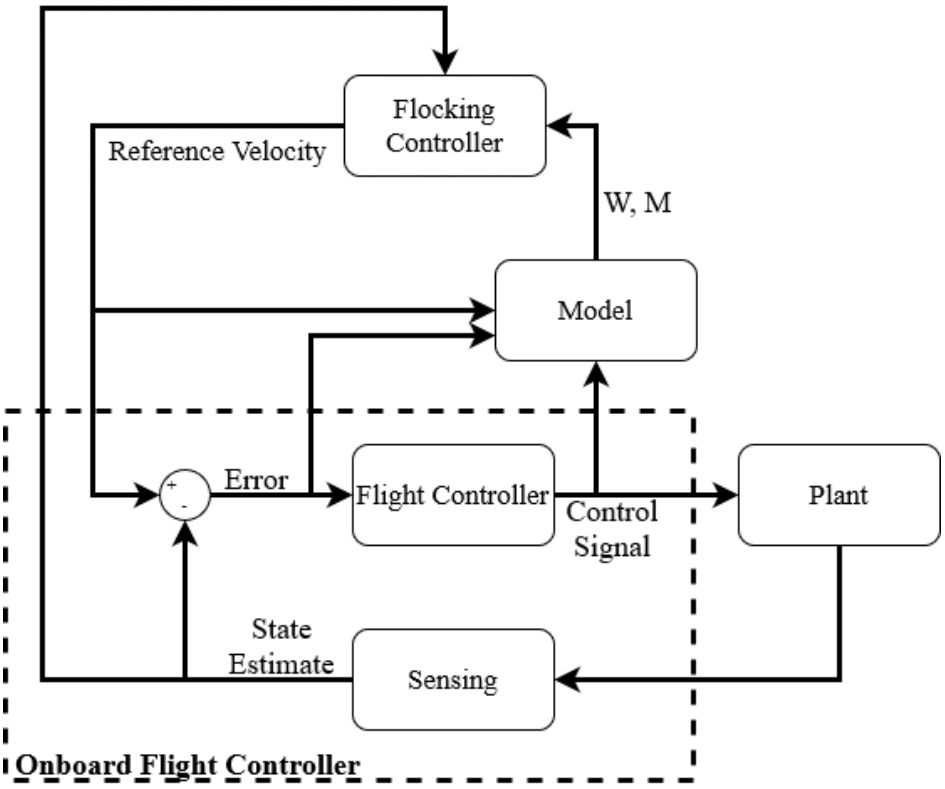

Figure 3.7: Proposed control diagram that infers the upwash force and moment imposed on the UAV by sampling signals from the onboard flight controller.

this is the fundamental problem of simultaneous actions in decentralized control. One popular approach is to use a consensus protocol to estimate $\frac{\partial E}{\partial t}$, e.g., ADMM [97], which has shown some success in the flocking literature [98]. Alternatively, it is possible for the agents to iteratively generate trajectories to converge on a locally optimal control strategy, [26, 99]. To minimize the iterative and communication cost, we instead chose to estimate $\dot{E}$ and demonstrate that the estimation error converges to zero. Each UAV estimates $\dot{E}$ by predicting control action of nearby agents. Starting at the front of the flock, each UAV i updates its control action, which all other UAVs store. Thus, the front UAV assumes that each of the followers moves at a constant speed [78], while the following UAVs can dynamically react to the control action of the leader. Ties in the sequence of updating are arbitrarily broken by vehicle index. This UAV sequencing

strategy is supported by Assumption 12 and the fact that each vehicle is primarily affected by the wake of the vehicle in front of it. We demonstrate that the system, under this update scheme, approaches a steady V formation in the following section.

3.2.4 Simulation Results

To demonstrate the performance of our control algorithm we simulate a system of $N = 4$ UAVs Using data for the RQ-11 Raven [100] and the update scheme proposed in Section 3.2.3.1. The Raven weighs 18.7 N, with a 1.4 m wing span and a nominal speed of 12 m/s. Approximating the density of air as $\rho = 1.2$ kg/m^3 yields the irrotational vortex strength using the Kutta-Joukowski theorem [93],

$$\Gamma = \frac{L}{2b\rho u} = \frac{18.7}{1.2 \cdot 1.2 \cdot 9} \approx 1.3 \text{ m}^2/\text{s}. \tag{3.65}$$

We expect r^* to be $\approx 5\%$ of the span length (e.g., [32] uses 5.4%); thus we select $r^* = 0.054$ m. This implies,

$$\Omega = \frac{\Gamma}{2\pi(r^*)^2} \approx 70 \text{ m/s}^2. \tag{3.66}$$

Next, we select $C_2 = C_1 = 1$. We initialize the UAVs in a vertical line with an initial orientation of $\theta_G = 0$ and a center-to-center spacing of $3b = 2.1$ m. We present a sequence of simulations in Fig. 3.8, which demonstrates how the agents initially coalesce to maximize their upwash benefit, begin to align in the north-eastern direction, and eventually form a V formation.

To quantify the impact of the V formation, we selecte $\kappa = 1$ and calculated the cost functional E_i for each UAV $i \in \mathcal{A}$ at each time step. We present the total cost, the maximum cost, minimum cost, and terminal cost for each UAV in Table 3.1. Note that a cost of zero corresponds to the agent flying in isolation; while the UAVs do occasionally experience a positive cost during the initial transient, the total cost for each UAV shows significant energy savings. The instantaneous cost is presented

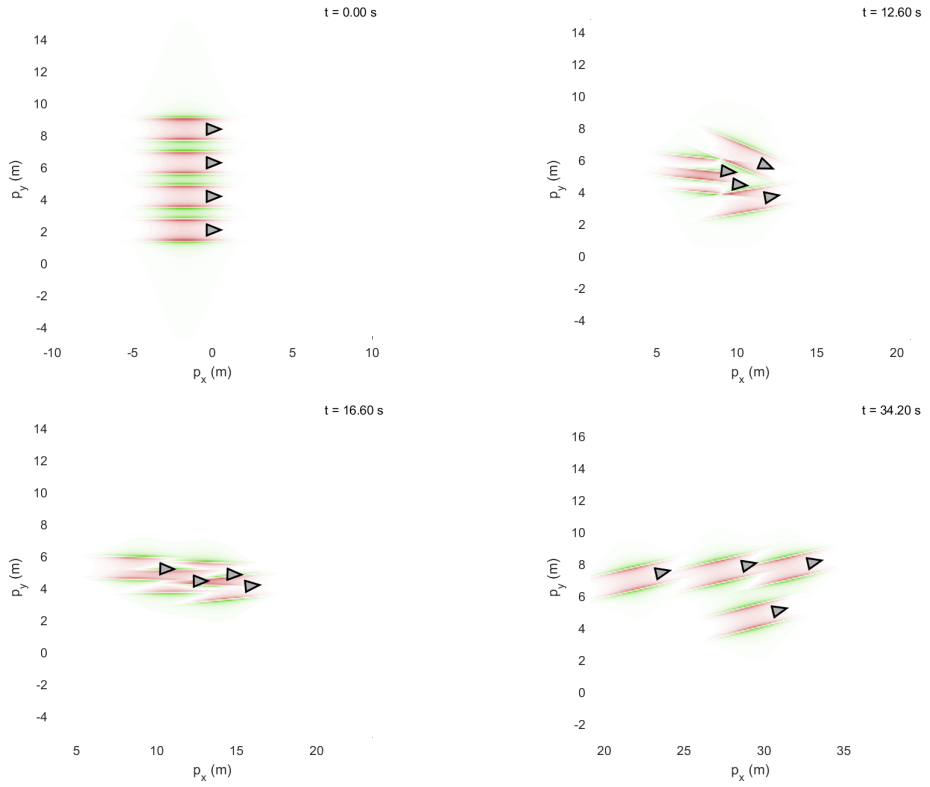

Figure 3.8: A sequence of simulation snapshots over 40 seconds for $N = 4$ UAVs initialized in a line formation

in Fig. 3.9, which shows the minimum, maximum, and mean velocity of the system. The system-level cost is the mean cost multiplied by by $N = 4$ agents. Finally, we can see that the cost remains relatively constant for the last 10 seconds of the simulation, implying the UAVs have reached a steady-state configuration.

3.2.5 A Note on Heterogeneity

One serendipitous result of this work is evidence that introducing heterogeneity has the potential to further improve the fuel consumption of all UAVs. Theorem 5 demonstrates that any UAV gaining an upwash benefit has a lower optimal speed than when flying in isolation. However, each UAV must match the speed of the UAV in front of it to maintain the formation. Thus, if UAVs toward the front of the flock have

	UAV 1	UAV 2	UAV 3	UAV 4
Total Cost	-2.5	-11.2	-65.1	-46.5
Max Cost	0.2	6.9	4.5	0.6
Min Cost	-0.5	-5.6	-8.4	-4.4
Terminal Cost	0	-0.1	-0.1	-2.0

Table 3.1: The energy cost for each UAV, including the total cost over the entire period, the maximum and minimum instantaneous value of the cost, and the cost value of the final time step.

a lower (isolated) optimal speed, the following UAVs can expend less energy while still maintaining the flock's shape. This heterogeneity could be introduced through the mechanical and aerodynamic design of the agents, but the most straightforward approach is to have UAVs with reduced mass, e.g., less fuel or a smaller payload, move toward the front of the formation; a smaller mass requires less lift to maintain a constant height, which reduces the induced drag—and thus reduces the optimal airspeed.

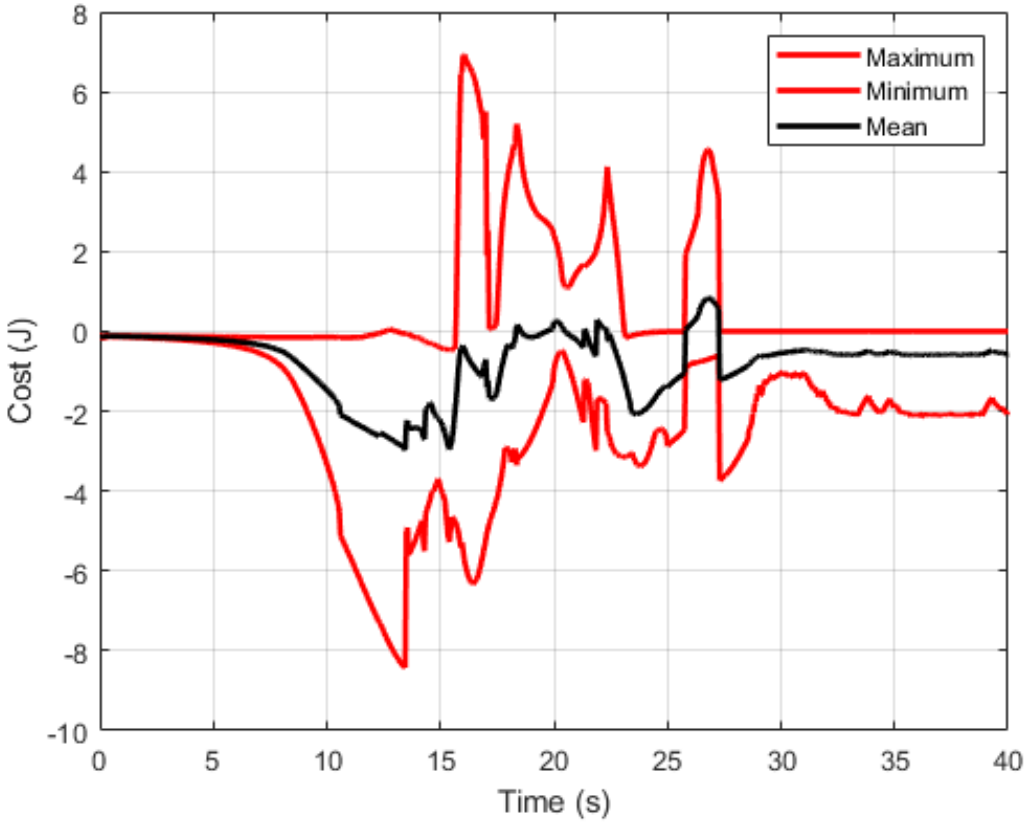

Figure 3.9: Maximum, minimum, and mean cost experienced by the UAVs for the duration of the simulation. A cost of zero corresponds to flying in isolation.

3.3 Constraint-Driven Optimal Control for Emergent Swarming and Predator Avoidance

3.3.1 Introduction

In this section, we derive a distributed control algorithm that induces cluster flocking in a multi-agent system. Prior work has primarily relied on reinforcement learning to achieve predator avoidance, including a multi-level approaches [101] and policy sharing [17, 102]. Traditional control approaches tend to achieve swarming behavior by implementing Reynolds flocking rules using potential fields [9]. These approaches have two major drawbacks. First, they inevitably drive agents into a regular

lattice formation [8], which is not conductive to swarming. Second, potential fields are known to cause steady oscillation in agent trajectories and exacerbate deadlock in constrained environments [15].

In contrast to existing approaches, we propose a biologically-inspired approach based on an analysis of sand-eel schools in the presence of predators [103] built on our previous work with set-theoretic control [59, 104, 105], where we embed inter-agent and environmental interactions as state and control constraints in an optimal control problem. Our set-theoretic approach has the advantage of being interpretable, i.e., the cause of an agents' action can be deduced by examining which constraints are currently active. Our technical results are closely related to the control barrier function (CBF) literature, particularly multi-agent CBFs [75]. However, our approach does not require the constraints to be encoded as sub-level sets of a continuous function—we work with the sets directly. We also propose a solution to the open problem of constraint incompatibility through an event-triggered constraint relaxation scheme. Finally, we present a mapping between constraint-driven control and switching systems, which provides a rigorous and interpretable description of each boids' behavior. The contributions of this section are: 1) a decentralized optimal control algorithm that yields emergent swarming behavior (Problem 6), 2) an event-triggered scheme to selectively relax constraints and guarantee feasibility (Lemmas 7–9), 3) a rigorous mapping between constraint-driven control and switching systems (Definition 12), and 4) simulation results demonstrating emergent cluster flocking and predator avoidance behaviors (Section 3.3.4).

The remainder of the section is organized as follows. In Section 3.3.2, we formulate the cluster flocking problem and discuss our working assumptions. In Section 3.3.3, we derive our optimal control policy, derive the safe action sets, and map the problem to a switching system. Finally, in Section 3.3.4, we validate our results in two simulations with 15 boids; the first demonstrates emergent cluster flocking, and the second demonstrates predator avoidance.

3.3.2 Problem Formulation

We consider a set of $N \in \mathbb{N}$ boids indexed by the set $\mathcal{B} = \{1, 2, \ldots, N\}$. Each boid $i \in \mathcal{B}$ obeys second-order integrator dynamics,

$$\begin{aligned} \dot{\boldsymbol{p}}_i(t) &= \boldsymbol{v}_i(t), \\ \dot{\boldsymbol{v}}_i(t) &= \boldsymbol{u}_i(t), \end{aligned} \tag{3.67}$$

where $\boldsymbol{p}_i(t), \boldsymbol{v}_i(t) \in \mathbb{R}^2$ correspond to the position and velocity of each boid, and $\boldsymbol{u}_i(t) \in \mathbb{R}^2$ is the control input. We also impose the state and control constraints,

$$\boldsymbol{p}_i(t) \in \mathcal{P}, \tag{3.68}$$

$$\boldsymbol{u}_i(t) \in \mathcal{U}, \tag{3.69}$$

where $\mathcal{P} \subset \mathbb{R}^2$ is a non-empty intersection of half-planes and $\mathcal{U} = \{u \in \mathbb{R}^2 : ||u||_\infty \leq u_{\max}\}$ ensures the boids' do not exceed their maximum control input. We employ the infinity norm to simplify our mathematical exposition; however, the norm does not impose any restrictions in our approach.

We account for interactions between boids using Voronoi tessellation [106]. Under this approach, each boid is considered the center of a Voronoi cell. We define the Voronoi set $\mathcal{V}(t) \subset \mathcal{B} \times \mathcal{B}$ to contain (i, j) and (j, i) when the Voronoi cells i and j share a common edge. Equivalently, the set $\mathcal{V}(t)$ is the Delaunay triangulation of the boids' positions.

Definition 10 (Voronoi Neighborhood)**.** The neighborhood of each boid $i \in \mathcal{B}$ is the set,

$$\mathcal{N}_i(t) \coloneqq \big\{ j \in \mathcal{B} \ : \ (i, j) \in \mathcal{V}(t) \big\}, \tag{3.70}$$

where boid i can receive information, via communication or sensing, with any other boid $j \in \mathcal{N}_i(t)$.

As with $k-$nearest neighbors, the sensing radius of a Voronoi neighborhood may

grow unbounded in general. Similar to past work [104, 107], we do not presume the boids possess infinite sensing capabilities; rather that Definition 10 describes the interactions between boids over their relatively small separating distances. One potential solution is to only consider Voronoi neighbors that are within a fixed sensing range [106], although results from biology demonstrate that this is, in general, unnecessary [108].

Our objective is to generate emergent swarming behavior, such that the boids remain close to their neighbors to avoid predators [13, 102, 103]. To achieve an aggregate swarming motion, we implement a variation of the disk flocking constraint proposed in [104]. First, we determine the neighborhood center for each boid $i \in \mathcal{B}$,

$$\boldsymbol{c}_i(t) = \frac{1}{|\mathcal{N}_i(t)|} \sum_{j \in \mathcal{N}_i(t)} \boldsymbol{p}_j(t). \tag{3.71}$$

Note that Definition 10 guarantees $|\mathcal{N}_i(t)| > 0$. We use the neighborhood center to construct the relative position vector,

$$\boldsymbol{r}_i(t) \coloneqq \boldsymbol{p}_i(t) - \boldsymbol{c}_i(t). \tag{3.72}$$

Finally, to achieve swarming, we require each boid i to approach and remain within a distance $R \in \mathbb{R}_{>0}$ of the neighborhood center, i.e.,

$$\begin{cases} ||\boldsymbol{r}_i(t)|| - R & \text{if } ||\boldsymbol{r}_i(t)|| \leq R, \\ \frac{||\dot{\boldsymbol{r}}_i(t)||}{u_{\max}} \boldsymbol{u}_i(t) \cdot \boldsymbol{r}_i(t) + \dot{\boldsymbol{r}}_i(t) \cdot \boldsymbol{r}_i(t) & \text{if } ||\boldsymbol{r}_i(t)|| > R, \end{cases} \leq 0.$$

Note that the first case is trivially satisfied, i.e., the boid must remain within the disk while inside the disk. Thus, we write

$$\begin{aligned} ||\boldsymbol{r}_i(t)|| > R \implies \\ \frac{||\dot{\boldsymbol{r}}_i(t)||}{u_{\max}} \boldsymbol{u}_i(t) \cdot \boldsymbol{r}_i(t) + \dot{\boldsymbol{r}}_i(t) \cdot \boldsymbol{r}_i(t) \leq 0. \end{aligned} \tag{3.73}$$

We emphasize that our objective is not to trap boid i within the disk of radius R centered at $\boldsymbol{c}_i(t)$. Instead, we expect the switching neighborhood topology and dynamic motion of $\boldsymbol{c}_i(t)$ to drive the swarming behavior. Additionally, the form of (3.73) is inspired by energy-saving techniques in [109]. Note that when boid i travels in the "correct" direction, i.e., $\dot{\boldsymbol{r}}_i(t) \cdot \boldsymbol{r} < 0$, the control action $\boldsymbol{u}_i(t)$ can take some values in the same direction as $\boldsymbol{r}_i(t)$. However, when boid i is traveling in the "wrong" direction, i.e., $\dot{\boldsymbol{r}}_i(t) \cdot \boldsymbol{r}_i(t) > 0$, the control action $\boldsymbol{u}_i(t)$ must be at least partially opposed to $\boldsymbol{r}_i(t)$ to drive boid i toward $\boldsymbol{c}_i(t)$.

Next, inspired by the empirical data collected on sand-eels [103], we model the predator as a ball of radius Γ. We define the relative distance vector between each boid i and the predator as,

$$\boldsymbol{d}_i(t) \coloneqq \boldsymbol{p}_i(t) - \boldsymbol{o}_i(t), \tag{3.74}$$

where $\boldsymbol{o}_i(t)$ is the position of the predator at time t. To ensure predator avoidance, we select a value of Γ larger than the diameter of the predator and employ a similar constraint to repel the boids,

$$\begin{aligned} ||\boldsymbol{d}_i(t)|| < \Gamma \implies \\ -\frac{||\dot{\boldsymbol{d}}_i(t)||}{u_{\max}} \boldsymbol{u}_i(t) \cdot \boldsymbol{d}_i(t) - \dot{\boldsymbol{d}}_i(t) \cdot \boldsymbol{d}_i(t) \leq 0. \end{aligned} \tag{3.75}$$

With the constraints defined, our next objective is to design an optimal control problem such that the individual boid motion generates emergent swarming behavior. To this end, we impose the following assumptions on our system.

Assumption 13. Each boid is equipped with a low-level controller that is capable of tracking the control input.

Assumption 14. Communication and sensing between the boids occurs instantaneously and noiselessly.

We impose Assumptions 13 and 14 to simplify our analysis and understand

how the system performs in the ideal case. Assumption 13 is common for trajectory generation problems, and it can be relaxed by introducing robust control terms or a safety layer, e.g., using a control barrier function [110]. Similarly, Assumption 14 can be relaxed by including time delays and uncertainty explicitly in the formulation and applying stochastic [66] or robust [92] control techniques.

Assumption 15. The boids have sufficient vertical space to avoid collisions between each other without an explicit collision-avoidance constraint.

Assumption 15 is common in 2D swarming applications [102, 111]. Furthermore, it has been thoroughly demonstrated that adding an extra dimension of motion can significantly reduce the likelihood of collisions [68].

3.3.3 Solution Approach

We employ constraint-driven control to generate the control input for each boid. This is an optimization approach wherein the desired behavior of each boid is encoded as a constraint in an optimal control problem. This technique has been used successfully to control multi-agent systems [46, 75, 84]. Each boid solves the optimal control problem reactively, i.e., they take an action at each time-instant without a planning horizon. Our motivation for this is twofold: first, it overcomes the computational and communication costs associated with decentralized trajectory planning [9]. Second, it allows boids to freely enter and leave the domain, e.g., due to operating constraints, mechanical failure, or predation, as the constraint boundaries are a function of the local system state. For the remainder of our exposition, we omit the explicit dependence of state variables on t when no ambiguity arises.

We start with the position constraint (3.68), which is not an explicit function of the control input. Let $k = 1, 2, \ldots, M$ index the M hyperplanes that define the boundary of $\mathcal{P}$. Each hyperplane $k = 1, 2, \ldots, M$ consists of a normal vector $\hat{\boldsymbol{n}}_k \in \mathbb{R}^2$

and offset $b_k \in \mathbb{R}$; the signed distance to the surface of hyperplane k is,

$$d_{ik} = \boldsymbol{p}_i \cdot \hat{\boldsymbol{n}}_k + b_k, \tag{3.76}$$

for boid $i \in \mathcal{B}$. Note that our convention assumes the normal vector $\boldsymbol{n}_k$ points away from the feasible region $\mathcal{P}$. To guarantee constraint satisfaction, we require the derivative of (3.76) to be non-positive when the constraint is active, i.e.,

$$\boldsymbol{p}_i \cdot \hat{\boldsymbol{n}}_k + b_k = 0 \implies \boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_k \leq 0. \tag{3.77}$$

This safety constraint (3.77) can be achieved by using a stopping distance constraint for each $k = 1, 2, \ldots, M$ [46],

$$g_{ik} = \Big(\boldsymbol{p}_i \cdot \hat{\boldsymbol{n}}_k + b_k\Big) + \alpha \frac{\Big(\boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_k\Big)^2}{2u_{\max}} \leq 0, \tag{3.78}$$

where $\alpha \in \mathbb{R}_{>0}$ is a parameter that determines the stopping distance. Note that (3.78) trivially satisfies (3.77). This leads to our definition of the safe action set.

Definition 11 (Safe Action Set)**.** For each boid $i \in \mathcal{B}$ at time t, the safe action set is,

$$\begin{aligned}\mathcal{A}_i^s \coloneqq \Big\{ \boldsymbol{u}_i \in \mathbb{R}^2 \;:\; & ||\boldsymbol{u}_i||_\infty - u_{\max} \leq 0, \\ & \Big(\boldsymbol{p}_i \cdot \hat{\boldsymbol{n}}_k + b_k\Big) + \alpha \frac{\Big(\boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_k\Big)^2}{2u_{\max}} = 0 \implies \\ & \boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_k \Big(1 + \frac{\alpha}{u_{\max}} (\boldsymbol{u}_i \cdot \hat{\boldsymbol{n}}_k)\Big) \leq 0, \\ & \forall k = 1, 2, \ldots, M \Big\}.\end{aligned}$$

In our approach, we constrain the boids to remain within an axis-aligned rectangular domain, i.e., $\mathcal{P}$ is constructed from two pairs of parallel hyperplanes that intersect at right angles.

With the safe action set and constraints defined, each boid i also requires a

notion of performance to select the "best" control input. Following the ecologically-inspired paradigm [45] would suggest minimizing the norm of the control input; this arguably yields a minimum effort policy. However, we have previously demonstrated that selecting an appropriate objective function is critical to achieve a desired emergent behavior [46]. As discussed in [103], sand-eels tend to cruise at a constant speed of approximately 2 body lengths per second. Thus, we require each boid to match an optimal swimming speed, denoted $||\boldsymbol{v}_i^*||$, as closely as possible, i.e.,

$$J_i\big(\boldsymbol{v}_i(t)\big) = \Big(||\boldsymbol{v}_i(t)+\boldsymbol{u}_i(t)dt|| - ||\boldsymbol{v}_i{}^*||\Big)^2. \tag{3.79}$$

We interpret the optimal swimming speed as being bio-mechanically advantageous, i.e., if $J = ||\boldsymbol{u}_i||$ minimizes energy consumption, then (3.79) corresponds to minimum-power locomotion. Combining the cost (3.79) with the previously described constraints yields the optimal control problem solved by each boid.

Problem 6. For each boid $i \in \mathcal{B}$ at time t, apply the control action that solves,

$$\min_{\boldsymbol{u}_i(t)} \Big(||\boldsymbol{v}_i(t) + \boldsymbol{u}_i(t)dt|| - ||\boldsymbol{v}_i{}^*||\Big)^2$$

subject to:

$$\boldsymbol{u}_i(t) \in \mathcal{A}_i^s, \text{ (3.67)}, \text{ (3.73)}, \text{ (3.75)}.$$

Next, we present a result that guarantees recursive feasibility for the safe action set.

Theorem 7. For a fixed value of $\alpha \geq 1$, if a boid $i \in \mathcal{B}$ satisfies (3.78) at some time t for a rectangular domain $\mathcal{P}$, then $\mathcal{A}_i^s$ satisfies recursive feasibility for all future time.

Proof. Let the rectangular domain $\mathcal{P}$ consists of four hyperplanes, indexed by $k = 1, 2, 3, 4$ such that $\hat{\boldsymbol{n}}_1 = -\hat{\boldsymbol{n}}_3$ and $\hat{\boldsymbol{n}}_2 = -\hat{\boldsymbol{n}}_4$. Without loss of generality, let $\boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_1 > 0$ and $\boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_2 > 0$. When the safety constraint (3.78) is not active, boid i may take any

action satisfying the control bounds (3.92). However, when (3.78) is active, we must ensure its derivative is non-positive to guarantee safety. Taking the derivative of (3.78) and combining terms yields,

$$\dot{g}_{ik} = \boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_k \Big(1 + \frac{\alpha}{u_{\max}} \big(\boldsymbol{u}_i \cdot \hat{\boldsymbol{n}}_k\big)\Big). \tag{3.80}$$

We seek a control input such that $\dot{g}_{ik} \leq 0$. For $k = 1, 2$, dividing by $\boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_k > 0$ yields a condition on $\boldsymbol{u}_i$,

$$\boldsymbol{u}_i \cdot \hat{\boldsymbol{n}}_1 \leq -\frac{u_{\max}}{\alpha}, \quad \boldsymbol{u}_i \cdot \hat{\boldsymbol{n}}_2 \leq -\frac{u_{\max}}{\alpha}. \tag{3.81}$$

Similarly, for $k = 3, 4$, dividing by $\boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_k < 0$ implies,

$$\boldsymbol{u}_i \cdot \hat{\boldsymbol{n}}_3 \geq -\frac{u_{\max}}{\alpha}, \quad \boldsymbol{u}_i \cdot \hat{\boldsymbol{n}}_4 \geq -\frac{u_{\max}}{\alpha}. \tag{3.82}$$

Substituting $\hat{\boldsymbol{n}}_1 = -\hat{\boldsymbol{n}}_3$ and $\hat{\boldsymbol{n}}_2 = -\hat{\boldsymbol{n}}_4$ into (3.82) yields the conditions,

$$\boldsymbol{u}_i \cdot \hat{\boldsymbol{n}}_1 \leq \frac{u_{\max}}{\alpha}, \quad \boldsymbol{u}_i \cdot \hat{\boldsymbol{n}}_2 \leq \frac{u_{\max}}{\alpha}. \tag{3.83}$$

Thus, to guarantee g_{ik} is nonincreasing, the control input must satisfy (3.81) and (3.83), i.e.,

$$\boldsymbol{u}_i \cdot \hat{\boldsymbol{n}}_1 \leq -\frac{u_{\max}}{\alpha} \leq \frac{u_{\max}}{\alpha}, \tag{3.84}$$

$$\boldsymbol{u}_i \cdot \hat{\boldsymbol{n}}_2 \leq -\frac{u_{\max}}{\alpha} \leq \frac{u_{\max}}{\alpha}. \tag{3.85}$$

This is satisfied by the candidate control action,

$$\boldsymbol{u}_i = -\frac{u_{\max}}{\alpha} \hat{\boldsymbol{n}}_1 - \frac{u_{\max}}{\alpha} \hat{\boldsymbol{n}}_2, \tag{3.86}$$

as $\hat{\boldsymbol{n}}_1 \cdot \hat{\boldsymbol{n}}_2 = 0$ by definition. In our axis-aligned domain, the control constraint implies,

$$||\boldsymbol{u}_i||_\infty = \max\left\{\frac{u_{\max}}{\alpha}, \frac{u_{\max}}{\alpha}\right\} = \frac{1}{\alpha}u_{\max}, \tag{3.87}$$

which satisfies (3.69) for $\alpha \geq 1$. Finally, for the case that $\boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_k = 0$ for any $k = 1, 2, 3, 4$, the corresponding derivative $\dot{\boldsymbol{g}}_{ik} = 0$ for every control input. □

Thus, given a feasible initial state, Theorem 7 guarantees that each boid's trajectory will remain feasible indefinitely if its control action is selected from $\mathcal{A}_i^s$. However, we require each boid to jointly satisfy the safety, swarming (3.73), and predator avoidance (3.75) constraints to achieve emergent cluster flocking behavior. Thus, guaranteeing the recursive feasibility of $\mathcal{A}_i^s$ is insufficient to ensure a feasible control action exists. The following results provide the explicit conditions for constraint incompatibility, i.e., when the set of feasible control actions becomes empty.

Lemma 7. For a boid $i \in \mathcal{B}$, let $k = 1, 2$ index two perpendicular hyperplanes in the rectangular domain such that $\boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_k \geq 0$. Then, if (3.78) is strictly equal to zero and $||\boldsymbol{r}_i|| > R_i$, there is no feasible action if none of the conditions,

$$||\dot{\boldsymbol{r}}_i||\left(\begin{bmatrix}1\\ \alpha^{-1}\\ 1\\ \alpha^{-1}\end{bmatrix}(\hat{\boldsymbol{n}}_1 \cdot \hat{\boldsymbol{r}}_i) + \begin{bmatrix}1\\ 1\\ \alpha^{-1}\\ \alpha^{-1}\end{bmatrix}(\hat{\boldsymbol{n}}_2 \cdot \hat{\boldsymbol{r}}_i)\right) \geq \dot{\boldsymbol{r}}_i \cdot \hat{\boldsymbol{r}}_i, \tag{3.88}$$

hold at time t for $k = 1, 2$.

Proof. Under the premise of Lemma 7, we must determine when the constraint,

$$\frac{||\dot{\boldsymbol{r}}_i||}{u_{\max}}\boldsymbol{u}_i \cdot \hat{\boldsymbol{r}}_i + \dot{\boldsymbol{r}}_i \cdot \hat{\boldsymbol{r}}_i \leq 0, \tag{3.89}$$

$$\tag{3.90}$$

is incompatible with $\mathcal{A}_i^s$. First, $\dot{\boldsymbol{r}}_i = 0$ satisfies (3.89) for any $\boldsymbol{u}_i$, thus, we may divide (3.89) by $||\dot{\boldsymbol{r}}_i||$ and work with unit vectors for the remainder of the proof, i.e.,

$$\frac{1}{u_{\max}} \boldsymbol{u}_i \cdot \hat{\boldsymbol{r}}_i + \hat{\dot{\boldsymbol{r}}}_i \cdot \hat{\boldsymbol{r}}_i \leq 0. \tag{3.91}$$

Next, we consider the control $\boldsymbol{u}_i = -u_1 \hat{\boldsymbol{n}}_1 - u_2 \hat{\boldsymbol{n}}_2$. From the proof of Theorem 7, (3.69) and (3.81) imply that u_1 and u_2 must satisfy,

$$1 \geq \frac{u_1}{u_{\max}} \geq \frac{1}{\alpha}, \quad 1 \geq \frac{u_2}{u_{\max}} \geq \frac{1}{\alpha}. \tag{3.92}$$

The swarming constraint (3.91) becomes,

$$\frac{u_1}{u_{\max}} (\hat{\boldsymbol{n}}_1 \cdot \hat{\boldsymbol{r}}_i) + \frac{u_2}{u_{\max}} (\hat{\boldsymbol{n}}_2 \cdot \hat{\boldsymbol{r}}_i) \geq \hat{\dot{\boldsymbol{r}}}_i \cdot \hat{\boldsymbol{r}}_i. \tag{3.93}$$

The result follows from substituting the bounds (3.92) into (3.93). □

Lemma 8. For a boid $i \in \mathcal{B}$, let $k = 1, 2$ index two perpendicular hyperplanes in the rectangular domain such that $\boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_k \geq 0$. Then, if (3.78) is strictly equal to zero and $||\boldsymbol{d}_i|| < \Gamma$, there is no feasible action if none of the conditions,

$$||\dot{\boldsymbol{d}}_i|| \left(\begin{bmatrix} 1 \\ \alpha^{-1} \\ 1 \\ \alpha^{-1} \end{bmatrix} (\hat{\boldsymbol{n}}_1 \cdot \hat{\boldsymbol{d}}_i) + \begin{bmatrix} 1 \\ 1 \\ \alpha^{-1} \\ \alpha^{-1} \end{bmatrix} (\hat{\boldsymbol{n}}_2 \cdot \hat{\boldsymbol{d}}_i) \right) \leq \dot{\boldsymbol{d}}_i \cdot \hat{\boldsymbol{d}}_i, \tag{3.94}$$

hold at time t for $k = 1, 2$.

Proof. The proof Lemma 8 is identical to Lemma 7, and thus we omit it. □

Lemma 9. For a boid $i \in \mathcal{B}$, let $k = 1, 2$ index two perpendicular hyperplanes in the rectangular domain such that $\boldsymbol{v}_i \cdot \hat{\boldsymbol{n}}_k \geq 0$. Then, if (3.78) is strictly equal to

zero, $||\boldsymbol{r}_i|| > R_i$, and and $||\boldsymbol{d}_i|| > \Gamma$, there is no feasible control action if the linear inequalities,

$$\begin{bmatrix} ||\dot{\boldsymbol{r}}_i||\hat{\boldsymbol{n}}_1 \cdot \hat{\boldsymbol{r}}_i & ||\dot{\boldsymbol{r}}_i||\hat{\boldsymbol{n}}_2 \cdot \hat{\boldsymbol{r}}_i \\ -||\dot{\boldsymbol{d}}_i||\hat{\boldsymbol{n}}_1 \cdot \hat{\boldsymbol{d}}_i & -||\dot{\boldsymbol{d}}_i||\hat{\boldsymbol{n}}_2 \cdot \hat{\boldsymbol{d}}_i \end{bmatrix} \begin{bmatrix} \frac{u_1}{u_{\max}} \\ \frac{u_2}{u_{\max}} \end{bmatrix} \geq \begin{bmatrix} \dot{\boldsymbol{r}}_i \cdot \hat{\boldsymbol{r}}_i \\ -\dot{\boldsymbol{d}}_i \cdot \hat{\boldsymbol{d}}_i \end{bmatrix}$$

has no solution that also satisfies $\frac{1}{\alpha} \leq \frac{u_1}{u_{\max}} \leq 1$ and $\frac{1}{\alpha} \leq \frac{u_2}{u_{\max}} \leq 1$.

Proof. The proof of Lemma 9 is constructed by satisfying Lemmas 7 and 8 jointly. □

The existing ecologically-inspired robotics literature suggests employing slack variables to manage constraint incompatibility [29, 45]. However, it is unclear why one would add slack to the predator avoidance constraint when the premise of Lemma 8 is not satisfied. For this reason, we use Lemmas 7–9 to selectively relax the predator avoidance and swarming constraints; this implies an equivalent switching system that completely describes the behavior of each boid. We define this system next.

Definition 12. Each boid $i \in \mathcal{B}$ can be modeled as a switching system with three states: 1) *Nominal*, which considers all constraints; 2) *Strained*, which relaxes the swarming constraint; and 3) *Evasive*, where the boid executes an evasive maneuver. Boid i transitions between these states based on whether the premises of Lemmas 7–9 are satisfied at each time; this is described by Fig. 3.10.

Note that defining an appropriate evasive behavior when Lemma 8 holds, e.g., a fountain [112] or flash [103] maneuver, is beyond the scope of this work; in our simulations (Section **??**), we simply relax the predator-avoidance constraint. The final step is to tune the system parameters, which we discuss, along with the simulation results, in the following section.

3.3.4 Simulation

To validate our optimal control policy, we solved Problem 6 for $N = 15$ boids over a 120 second time interval. Next, we present our simulation parameters and the

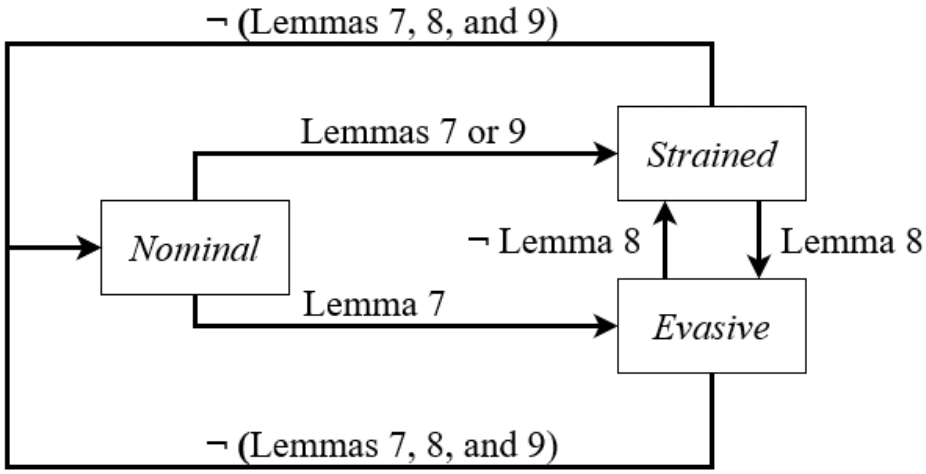

Figure 3.10: A switching system that describes each boids' feasible action space based on whether the premise of Lemmas 7–9 are satisfied.

physical intuition behind them, followed by simulations that demonstrate the desired cluster flocking and predator avoidance behaviors. Additional details and simulation videos can be found on the dedicated website of the original manuscript, https://sites.google.com/view/ud-ids-lab/swarming.

Based on the information given in [103], we selected a diameter of 5 cm for each boid, which implies an optimal speed of approximately 12.5 cm/s. Intuitively, it is desirable for each boid $i \in \mathcal{B}$ to have a small actuation limit relative to the desired speed. Each boid ought to approach its neighborhood center $\boldsymbol{c}_i$ at a high speed, overshoot it, and circle back toward $\boldsymbol{c}_i$ in a wide arc. This circling motion will also influence the topology of the Voronoi neighborhoods, which will further perturb the flock. Ideally, these perturbations will push some boids to the edge of the flock to counteract flock collapse [8]. Additionally, we select a square domain $\mathcal{P}$ that is large enough for cluster flocking to occur. We summarize our simulation parameters in Table 3.2.

To simulate the swarming behavior, we initialize all boids at rest with random initial positions within the domain $\mathcal{P}$ such that none overlap. At each time step,

Table 3.2: Simulation parameters used to generate swarming behavior.

Domain Size (m)	v^* (m/s)	$u_{\max}$ (m/s^2)	R (cm)	Γ (cm)
6	0.125	0.1	2.5	25

we solve Problem 6 and may relax some constraints according to Definition 12. The behavior of the swarm is visualized in Figs. 3.11 and 3.12, which show two time snapshots from the simulation. Figure 3.11 shows the initial behavior of the boids 21 seconds into the simulation. Starting near the center of the domain, the boids begin travelling in the north-western direction and exhibit a swirling motion. This is is visible from their tails, which show 8 seconds of trajectory history. After reaching the north-western hyperplane, the boids quickly turn around and travel to the south-east. Figure 3.12 shows behavior qualitatively similar to the cruising behavior described by [103], where their velocities are relatively constant in direction and magnitude.

Next, we introduce a simple predator model. The data in [103] implies that individual sand-eels treat predators as a moving obstacles. In fact, they explicitly state that "... the mackerel ate very few of the sand-eels throughout the duration of the experiment ..."—implying that the predator avoidance behavior ought to emerge without an antagonistic predator model. With this justification, our predator follows a simple rule: orient toward the center of the boid flock and travel in a straight line for 8 seconds. The predator moves 20% faster than the boids, and as such it is able to pass through the swarm and influence its behavior. We found that 8 seconds was a reasonable tradeoff to have the predator make several passes through the swarm without requiring significantly more simulation time. As with the previous simulation, the flock quickly formed and began cruising across the domain. The predator made multiple passes through the swarm, and each time the boids avoided the predator and quickly reformed. A simulation snapshot is presented in Fig. 3.13 near $t = 52$ s, where the boids qualitatively exhibit the vacuole behavior seen in the sand-eel experiments [103].

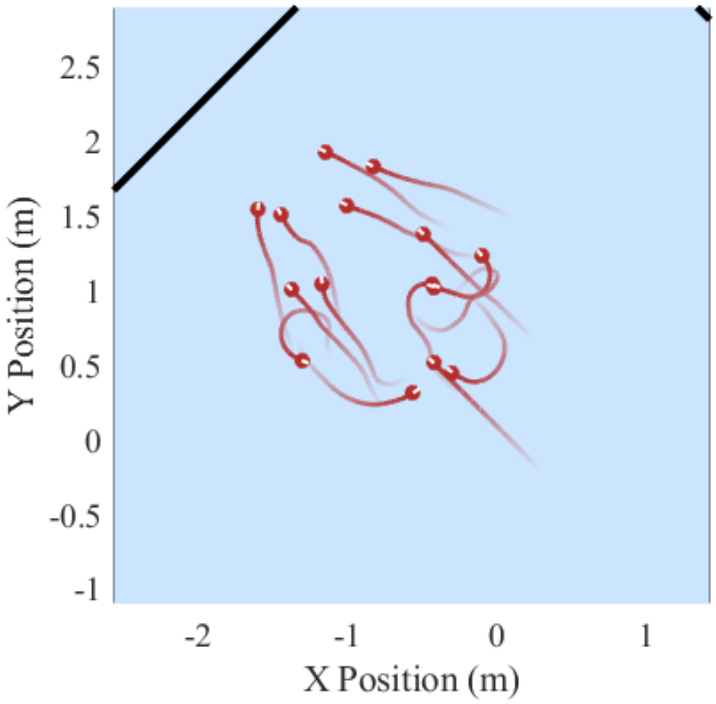

Figure 3.11: Boids circling and forming the initial flock at approximately $t = 21$ seconds; tails show 8 seconds of trajectory history.

Finally, we saved the size of each boids' neighborhood (Definition 10) at each time instant throughout the simulation. A histogram of neighborhood size is given in Fig. 3.14 for the simulation containing the predator. The distribution of neighborhood sizes approximates a discrete Weibull distribution, with 4 neighbors being the most frequent. This supports existing results in the biology literature [108], which claims that only considering 3–5 neighbors may be optimal for predator avoidance in 2D swarms.

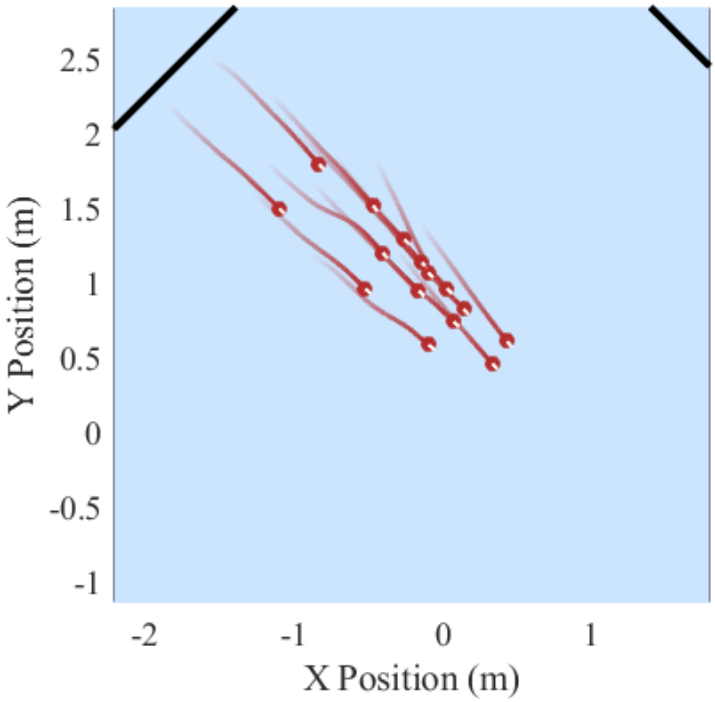

Figure 3.12: Boids cruising to the south-east at approximately $t = 85$ seconds after reaching the north-west wall and changing direction; tails show 8 seconds of trajectory history.

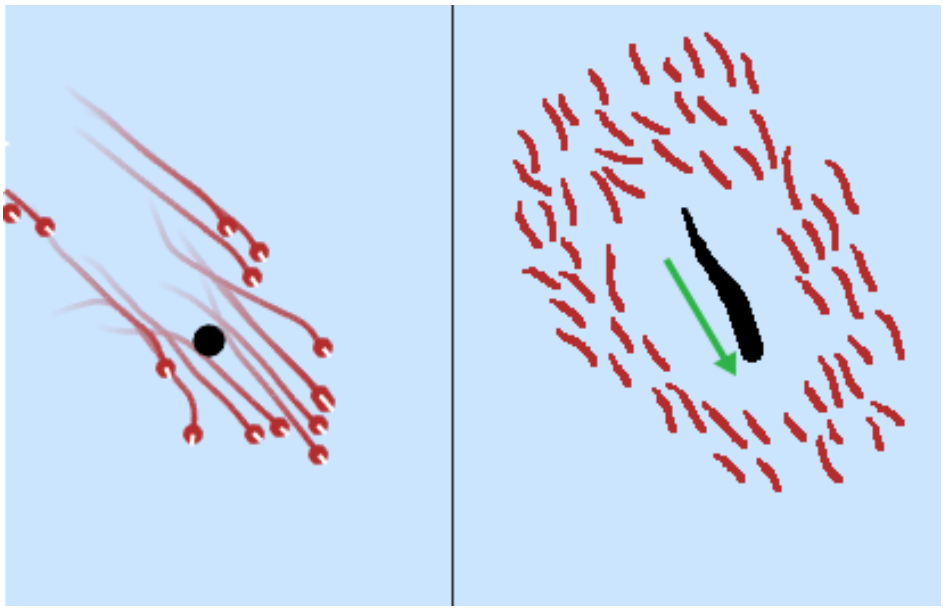

Figure 3.13: Left: apparent vacuole behavior exhibited by the boids the predator approaches from behind. Right: vacuole behavior observed in sand-eels, recreated from Pitcher and Wyche (1983).

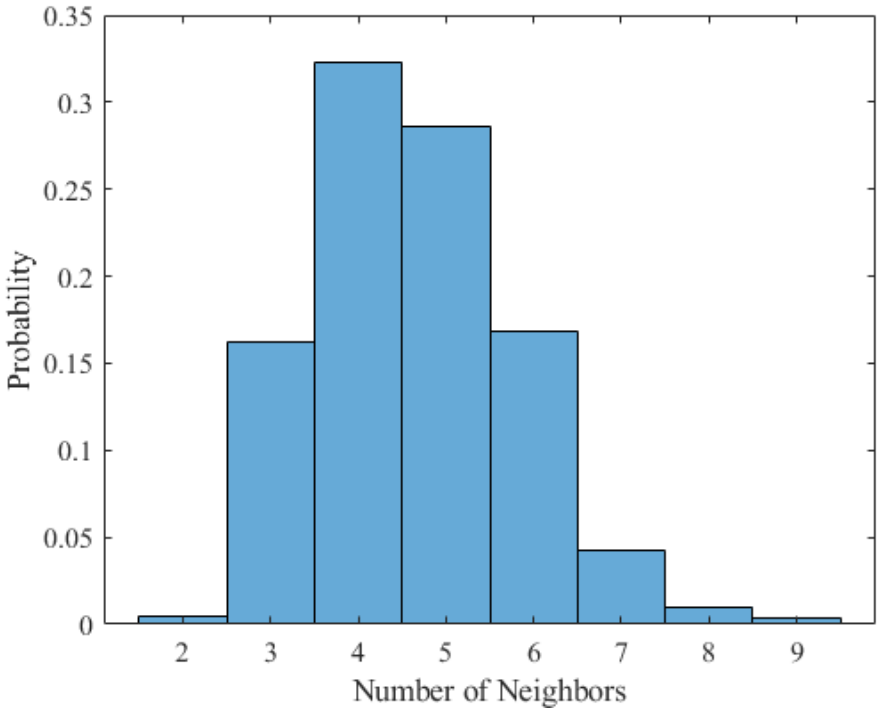

Figure 3.14: Neighborhood size histogram for $N = 15$ boids during the 120 second simulation with a predator.

3.3.5 Conclusion

In this chapter, we applied constraint-driven control to generate emergent behavior in three systems: highway platooning with CAVs, V formations with UAVs, and swarming predator avoidance with 2D agents. In each case, we constructed an optimal control policy to drive each agent to an equilibrium point—satisfying the definition of emergence proposed by Ashby. This had additional benefits on system performance in the first two cases, as the equilibrium points corresponded to energy-minimizing agent configurations. Additionally, we proved that the control strategies satisfied recursive feasibility, and thus the agents are capable of continuous operation indefinitely.

The concept of relaxing constraints to guarantee feasibility was generalized for a larger system in the third article, in which the agents were subjected to safety, predator avoidance, and neighborhood constraints. Relaxing these constraints implied an equivalent state machine formulation, which is easily interpretable while still allowing for rigorous guarantees on the behavior individual agents and the overall system. The first article also demonstrates how to derive global guarantees on system behavior from the local agent interactions. The emergent behavior of each system was demonstrated in simulation, and in the first case we validated that our data-driven approach handles the addition and removal of agents from the system, subject to the hard safety constraints.

There are many potential directions of future work in the domain for this work. One intriguing direction is extending the analysis to competitive agents, e.g., bicycle-riders, off-road vehicles, and multi-lane overtaking. Extensive simulations on larger scale systems would also be of value, and may present useful insights on how the behavior of individual agents can be adjusted to yield better system-level behavior. Finally, experiments that replicate line flocking and swarming behavior with physical agents would be of value, and may help to capture the magnitude of noise and disturbances that each agent experiences. To this end, Chapter 4 presents an original technique that quickly generates solutions to constrained continuous-time optimal control problems.

Chapter 4

REAL-TIME OPTIMAL CONTROL

> One must wait for what cannot be hastened.
>
> Anglo-Saxon Maxim
> Exter Book (10th Century CE)

To round out this dissertation, this chapter contains a new technique to quickly solve continuous-time constrained optimal control problems. For continuous-time systems, the optimality conditions take the form of an ordinary differential equation; unfortunately, these equations are generally unstable and take significant computational power and time to solve due to numerical instabilities [49]. This presents a significant challenge for multi-agent systems, and due the cost constraints on each agent, having a fast and performant trajectory generation is a prerequisite for applying the results of Chapters 2 and 3 to physical systems.

To solve constrained optimization problems in real time, we proposed a technique [59, 60] that exploits the property of differential flatness. According to the seminal paper on differential flatness [50], a system is differentially flat if a diffeomorphism between the original system dynamics an equivalent system with integrator dynamics can be constructed using endogenous variables. This enables designers to construct an optimal trajectory using integrator dynamics, and this optimal trajectory can be mapped back to the original problem using the diffeomorphism. Although a necessary and sufficient condition for differential flatness is not known to exist, many systems are known to be differentially flat. This includes ground vehicles, tractor-trailer systems, hopping robots, rigid body chains, satellites, cable towed systems, and many kinds of aircraft [113].

In the overwhelming majority of cases, existing literature selects a set of parameterized basis functions in the flat space to generate optimal trajectories. This includes polynomials [114], approximate Fourier series [51], and splines [115], which can generate efficient trajectories quickly. In contrast, this chapter presents an original technique that solves the optimality conditions numerically by exploiting the property of differential flatness to build optimal motion primitives. The resulting trajectories find better trajectories than existing techniques while using less computational power and memory, and thus they are more suitable for real-time control of multi-agent systems.

Finally, this chapter is concluded by an excerpt from an article on the real-time optimal control of connected and automated vehicles in our 1:25 scale robotic testbed [39]. While this control method [116, 117] was developed independently from my work on differential flatness, it is an excellent example of using the unconstrained optimal motion primitive to generate trajectories in real time on a physical system. Imposing the unconstrained motion primitive also has interesting consequences on the behavior of the system, and gives useful insights on how urban networks might be changed to further enhance the performance of connected and automated vehicles (CAVs).

4.1 Optimal Control of Differentially Flat Systems is Surprisingly Simple

4.1.1 Introduction

There is an increasing demand to push the boundaries of autonomy in cyber-physical systems (CPS) using experimental testbeds [38, 39, 118, 119] and outdoor experiments [16, 120]. As CPS achieve higher autonomy levels, they will be forced into complicated interactions [121] with other agents [9] and the surrounding environment [12]. These autonomous agents must be able to react quickly to their environment and re-plan efficient trajectories. To this end, we propose a new method to simplify real-time optimal trajectory planning by exploiting differential flatness.

A system is differentially flat if there exist a set of endogenous *flat* variables,

also called outputs, such that the original state and control variables can be written as an explicit function of the flat variables and a finite number of their derivatives. This yields an equivalent flat system that is completely described by integrator dynamics. Differentially flat systems have garnered significant interest since their introduction [50], and it has been shown that generating trajectories in the flat space can reduce computational time by at least an order of magnitude [122]. Differentially flat systems are closely related to feedback linearizable systems [123]; however, the standard control techniques for flat systems are distinct from feedback linearization.

The overwhelming majority of research on trajectory generation with differential flatness uses collocation techniques, i.e., finding optimal parameters for a set of basis functions in the flat output space. Under this approach, a designer selects an appropriate basis function for their application, e.g., polynomial splines [114, 124], Bezier curves [115], Fourier transforms [51], or piece-wise constant functions [125]. The parameters of these basis functions are optimally determined before they are transformed back to the original coordinates of the nonlinear dynamical system, which yields the optimal trajectory for the selected basis. A rigorous overview of this approach is given in a recent textbook [126].

In contrast, our approach is classified as indirect as we seek a solution from the optimality conditions. There is similar results for the so-called maximal inversion approach [52, 53], which proves that the optimality conditions for a differentially flat system can be separated into two parts—one describing the optimal state trajectory, and the other describing the optimal costate trajectory. This result is significant, as the general optimality conditions couple the evolution of the states and costates, which can lead to significant numerical instabilities [49]. While the authors in [53] proved that the optimality conditions are separable, in this paper, we provide the analytical form of the ordinary differential equation that describes their evolution. Furthermore, the results in [53] consider control-affine nonlinear systems. In contrast, in our approach, we do not require affinity in the control. We also derive the optimal boundary conditions in

the flat output space, which, to the best of our knowledge, has not been addressed in the literature to date. Finally, while recent work employs saturation functions to handle trajectory constraints [54], we analyze the state and control constraints directly. The main contributions of this paper are:

1. We present a set of ordinary differential equations that describe the evolution of the costates as explicit functions of the state and control variables (Theorem 8).

2. We derive an equivalent set of optimality conditions that are independent of the costates (Theorem 9). This independence property holds for interior-point constraints (Section 4.1.3.2) and trajectory constraints (Section 4.1.3.3).

3. We derive equivalent boundary conditions for the state and control variables when an initial or final state is left free or when the final time is unknown (Section 4.1.3.4).

4. We demonstrate a significant improvement in computational time and total cost compared to general-purpose numerical solvers (Section 4.1.4).

The remainder of the section is organized as follows. In Section 4.1.2, we provide the modeling framework and enumerate our assumptions before presenting our main theoretical results in Section 4.1.3. Then, in Section 4.1.4, we provide an illustrative example of controlling a double-integrator agent to avoid an obstacle and compare the performance of our approach to existing off-the-shelf optimal control libraries. Finally, we draw concluding remarks and present directions of future work in Section 4.3.

4.1.2 Problem Formulation

Consider the nonlinear dynamical system,

$$\dot{\mathbf{x}}(t) = \mathbf{f}\big(\mathbf{x}(t), \mathbf{u}(t)\big), \tag{4.1}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ and $\mathbf{u}(t) \in \mathbb{R}^m$, $n < m$, are the state and control vectors, respectively, and $t \in \mathbb{R}_{\geq 0}$ is time. A system is *differentially flat* if it satisfies the following definition.

Definition 13 (Adapted from [91])**.** The system described by (4.1) is said to be differentially flat if there exists a vector $\mathbf{y}(t) = (y_1(t), \ldots, y_m(t))$, such that:

1. The variables $y_i(t), i = 1, \ldots, m$ and their time derivatives are independent.
2. There exists a smooth mapping σ from $\mathbf{x}(t)$, $\mathbf{u}(t)$, and a finite number of its derivatives to $\mathbf{y}$, i.e.,

$$\mathbf{y}(t) = \sigma\big(\mathbf{x}(t), \mathbf{u}(t), \dot{\mathbf{u}}(t), \ldots, \mathbf{u}^{(p)}(t)\big), \tag{4.2}$$

   for some $p \in \mathbb{N}$.
3. The variables $\mathbf{x}(t)$ and $\mathbf{u}(t)$ can be expressed as smooth functions of $\mathbf{y}(t)$ and a finite number of its time derivatives, i.e.,

$$\mathbf{x}(t) = \gamma_0\big(\mathbf{y}(t), \dot{\mathbf{y}}(t), \ldots, \mathbf{y}^{(q)}(t)\big), \tag{4.3}$$
$$\mathbf{u}(t) = \gamma_1\big(\mathbf{y}(t), \dot{\mathbf{y}}(t), \ldots, \mathbf{y}^{(q)}(t)\big), \tag{4.4}$$

   for some $q \in \mathbb{N}$.

Furthermore, σ, γ_0, and γ_1 are smooth mappings between the flat and original spaces, thus, (4.2)–(4.4) constitute a diffeomorphism between the original and flat manifolds.

Note that the transformation between the original and output spaces achieved by (4.2) uses only the state and control variables, along with a finite number of derivatives. For this reason, differentially flat systems are also said to have *endogenous feedback*.

We impose the following assumptions for our analysis of the differentially flat system presented in Definition 13.

Assumption 16. The trajectory of the system is contained in open set where the functions (4.2)–(4.4) exist.

Assumption 17. The control actions in the original and flat spaces are upper and lower bounded.

Assumption 16 is a standard assumption in the literature [127] since there are no known algorithms that yield the mappings between the original and flat space [126]. Assumption 16 can be relaxed by constraining the trajectory to remain within a subset where (4.2)–(4.4) are defined.

Assumption 17 is standard in optimal control [89], particularly for physical systems where actuators are ultimately bounded by their physical strength or energy consumption. This assumption can be relaxed by allowing the control input to take the form of a Dirac delta function.

As an illustrative example of our approach, consider a unicycle traveling in the $\mathbb{R}^2$ plane.

Example 1. Let $\mathbf{x}(t) = [p_x(t), p_y(t), \theta(t)]^T$ be the state of a unicycle in the $\mathbb{R}^2$ plane, where $x(t)$ and $y(t)$ denote the position, and $\theta(t)$ denotes the heading angle. Let $\mathbf{u}(t) = [u_1(t), u_2(t)]^T$ be the vector of control actions, where $u_1(t)$ and $u_2(t)$ denote the forward and angular velocity, respectively. Then, the dynamics are given by

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} u_1(t)\cos\big(\theta(t)\big) \\ u_1(t)\sin\big(\theta(t)\big) \\ u_2(t) \end{bmatrix}. \tag{4.5}$$

This system admits $m = 2$ differentially flat output variables, $\mathbf{y}(t) = [y_1(t), y_2(t)]^T = [p_x(t), p_y(t)]^T$ [126]. The transformations (4.3) and (4.4) between the flat outputs and

original coordinates are

$$\begin{bmatrix} p_x(t) \\ p_y(t) \\ \theta(t) \end{bmatrix} = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \arctan2(\dot{y}_2, \dot{y}_1) \end{bmatrix}, \tag{4.6}$$

$$\begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} = \begin{bmatrix} \sqrt{\dot{y}_1(t)^2 + \dot{y}_2(t)^2} \\ \frac{\ddot{y}_2\dot{y}_1 - \dot{y}_2\ddot{y}_1}{\dot{y}_2^2 + \dot{y}_1^2} \end{bmatrix}, \tag{4.7}$$

which obey Assumption 16 for $u_1(t) \neq 0$.

Finally, consider a constrained optimal control problem for a system governed by (4.1) under Assumptions 16 and 17.

Problem 7. Consider the differentially flat system with running cost $L\big(\mathbf{x}(t), \mathbf{u}(t)\big)$ over the time horizon $[t^0, t^f] \subset \mathbb{R}_{\geq 0}$ and a final cost $\phi(\mathbf{x}(t^f), \mathbf{u}(t^f))$. Determine the optimal control input that minimizes the total cost, i.e.,

$$\begin{aligned} \min_{\mathbf{u}(t)} \quad & \phi\big(\mathbf{x}(t^f), \mathbf{u}(t^f)\big) + \int_{t^0}^{t^f} L\big(\mathbf{x}(t), \mathbf{u}(t)\big) dt \\ \text{subject to: } & (4.1), \\ & \hat{\mathbf{g}}\big(\mathbf{x}(t), \mathbf{u}(t), t\big) \leq 0, \\ \text{given: } & \text{initial conditions, final conditions,} \end{aligned}$$

where the initial and final states may be fixed, a function of the state variables, or left free. In addition, the function $\hat{\mathbf{g}}(\mathbf{x}(t), \mathbf{u}(t), t)$ defines a vector of state and control trajectory constraints.

In what follows, we present our main results, which yield a set of sufficient conditions for optimality that are only dependent on the state and control variables.

4.1.3 Main Results

4.1.3.1 Separability of the Optimality Conditions

We can convert Problem 7 into Brunovsky normal form using the flat output variables $\mathbf{y}(t) \in \mathbb{R}^m$ [127]. This yields a set of integrator chains starting from $\mathbf{y}(t)$,

$$\dot{y}_i^{(j)}(t) = y_i^{(j+1)}(t), \quad \begin{aligned} i &= 1, 2, \ldots, m, \\ j &= 0, 1, \ldots, k_i - 1, \end{aligned} \tag{4.8}$$

where $y_i^{(j)}(t)$ is the jth time derivative of base state $y_i(t)$ and $k_i \in \mathbb{N}$ is the length of the integrator chain for each base state. Next, we define equivalent state and control vectors for the system in the flat output space.

Definition 14. The dynamics of the system in the flat output space consists of m integrator chains of size k_i for $i = 1, 2, \ldots, m$. The state vector $\mathbf{s}(t)$ and control vector $\mathbf{a}(t)$ are

$$\mathbf{s}(t) = \left[y_1(t),\ \ldots,\ y_1^{(k_1-1)}(t),\ \ldots,\ y_m^{(k_m-1)}(t) \right]^T, \tag{4.9}$$

$$\mathbf{a}(t) = \left[y_1^{(k_1)}(t),\ \ldots,\ y_m^{(k_m)}(t) \right]^T. \tag{4.10}$$

Remark 3. For the unicycle system in Example 1, the flat state and control variables are

$$\mathbf{s}(t) = \left[y_1(t),\ y_2(t),\ \dot{y}_1(t),\ \dot{y}_2(t) \right]^T, \tag{4.11}$$

$$\mathbf{a}(t) = \left[\ddot{y}_1(t),\ \ddot{y}_2(t) \right]^T, \tag{4.12}$$

which consists of two integrator chains, each with a length of $k_i = 2$, for $i = 1, 2$.

Next, we apply (4.3) and (4.4) (Definition 13) to map Problem 7 to the flat output space.

Problem 8. Find the cost-minimizing trajectory in the flat output space,

$$\begin{aligned} \min_{\mathbf{a}(t)} \quad & \Phi(\mathbf{s}(t^f), \mathbf{a}(t^f)) + \int_{t^0}^{t^f} \Psi\big(\mathbf{s}(t), \mathbf{a}(t)\big)\, dt \\ \text{subject to: } & (4.8), \\ & \mathbf{g}\big(\mathbf{s}(t), \mathbf{a}(t), t\big) \leq 0, \\ \text{given: } & \text{initial conditions, final conditions,} \end{aligned}$$

where Φ, Ψ, $\mathbf{g}$, and the boundary conditions are constructed by composing ϕ, L, $\hat{\mathbf{g}}$, and the original boundary conditions of Problem 7 with the inverse of (4.3) and (4.4). Without loss of generality, we consider that $\mathbf{g}$ is an explicit function of at least one component of the control variable in the remainder of this section. We prove that this is eventually implied by Assumption 17 in Section 4.1.3.3.

Note that Problem 8 takes the nonlinearities of the original system dynamics and moves them into the objective function and constraints. This can be advantageous for optimal control problems, where satisfying kinematic constraints can lead to significant numerical challenges.

Next, we present our first main result, which decouples the state and costates for the Hamiltonian function associated with Problem 8.

Theorem 8. The costates $\lambda^{y_i^{(j)}}$ for each base state $i = 1, 2, \ldots, m$ and $j = 0, 1, \ldots, k_i - 1$ that correspond to Problem 8 are equal to

$$\lambda^{y_i^{(j)}} = \sum_{n=1}^{k_i - j} (-1)^n \frac{d^{n-1}}{dt^{n-1}} \big(\Psi_{y_i^{(j+n)}} + \boldsymbol{\mu}^T \mathbf{g}_{y_i^{(j+n)}} \big). \tag{4.13}$$

Proof. We follow the standard process for solving optimal control problems [89, 90].

First, we construct the Hamiltonian for Problem 8,

$$H = \Psi(\mathbf{s}(t), \mathbf{a}(t)) + \boldsymbol{\lambda}^T(t)\mathbf{I}(\mathbf{s}(t), \mathbf{a}(t)) + \boldsymbol{\mu}^T\mathbf{g}\big(\mathbf{x}(t), \mathbf{u}(t), t\big), \tag{4.14}$$

where $\boldsymbol{\lambda}(t)$ is the vector of costates, $\mathbf{I}(\mathbf{s}(t), \mathbf{a}(t))$ corresponds to the integrator dynamics, defined by (4.8), $\mathbf{g}$ is a vector of inequality constraints, and $\boldsymbol{\mu}$ is a vector of inequality Lagrange multipliers. To simplify the notation, we omit the explicit dependence on $\mathbf{a}(t)$, $\mathbf{s}(t)$, and t where it does not lead to ambiguity. The Euler-Lagrange equations are

$$-\dot{\boldsymbol{\lambda}}^T = \Psi_{\mathbf{s}} + \boldsymbol{\lambda}^T\mathbf{I}_{\mathbf{s}} + \boldsymbol{\mu}^T\mathbf{g}_{\mathbf{s}}, \tag{4.15}$$

$$0 = \Psi_{\mathbf{a}} + \boldsymbol{\lambda}^T\mathbf{I}_{\mathbf{a}} + \boldsymbol{\mu}^T\mathbf{g}_{\mathbf{a}}, \tag{4.16}$$

where the subscripts $\mathbf{a}$ and $\mathbf{s}$ correspond to partial derivatives with respect to those variables. We simplify (4.15) by exploiting the integrator structure of $\mathbf{I}$ for each element of $\mathbf{s}(t)$. This yields,

$$\dot{\lambda}^{y_i} = -\Psi_{y_i} - \boldsymbol{\mu}^T\mathbf{g}_{y_i}, \tag{4.17}$$

$$\dot{\lambda}^{y_i^{(j)}} = -\Psi_{y_i^{(j)}} - \lambda^{y_i^{(j-1)}} - \boldsymbol{\mu}^T\mathbf{g}_{y_i^{(j)}}, \tag{4.18}$$

for $i \in \{1, 2, \ldots, m\}$ and $j \in \{1, 2, \ldots, k_i - 1\}$. Similarly, simplifying each column of (4.16) yields

$$\lambda^{y_i^{(k_i-1)}} = -\Psi_{y_i^{(k_i)}} - \boldsymbol{\mu}^T\mathbf{g}_{y_i^{(k_i)}}, \tag{4.19}$$

for each output $i \in \{1, 2, \ldots, m\}$. Note that (4.19) satisfies (4.13) for $j = k_i - 1$. Next, taking a time derivative of (4.19) and substituting it into (4.18) with $j = k_i - 1$ yields

$$\frac{d}{dt}\Big(\Psi_{y_i^{(k_i)}} + \boldsymbol{\mu}^T\mathbf{g}_{y_i^{(k_i)}}\Big) = \Psi_{y_i^{(k_i-1)}} + \mu^T\mathbf{g}_{y_i^{(k_i-1)}} + \lambda^{y_i^{(k_i-2)}},$$

which implies,

$$\begin{aligned}\lambda^{y_i^{(k_i-2)}} &= -\left(\Psi_{y_i^{(k_i-1)}} + \boldsymbol{\mu}^T \mathbf{g}_{y_i^{(k_i-1)}}\right) \\ &\quad + \frac{d}{dt}\left(\Psi_{y_i^{(k_i)}} + \boldsymbol{\mu}^T \mathbf{g}_{y_i^{(k_i)}}\right),\end{aligned} \tag{4.20}$$

which satisfies (4.13) for $j = k_i - 2$. Taking repeated time derivatives of (4.20) and substituting (4.18) completes the proof of Theorem 8. □

Remark 4. For the unicycle system in Example 1, the covectors are

$$\boldsymbol{\lambda}^{\mathbf{y}} = -\left(\psi_{\dot{\mathbf{y}}} + \boldsymbol{\mu}^T \mathbf{g}_{\dot{\mathbf{y}}}\right) + \frac{d}{dt}\left(\psi_{\mathbf{a}} + \boldsymbol{\mu}^T \mathbf{g}_{\mathbf{a}}\right), \tag{4.21}$$

$$\boldsymbol{\lambda}^{\mathbf{y}^{(1)}} = -\left(\psi_{\mathbf{a}} + \boldsymbol{\mu}^T \mathbf{g}_{\mathbf{a}}\right). \tag{4.22}$$

Applying Theorem 8 to the Euler-Lagrange equations yields an equivalent set of optimality conditions that are independent of the costate variables, which we present in Theorem 9.

Theorem 9. The optimal trajectory for the system described in Problem 8 satisfies

$$\sum_{n=0}^{k_i} (-1)^n \frac{d^n}{dt^n}\left(\Psi_{y_i^{(n)}} + \boldsymbol{\mu}^T \mathbf{g}_{y_i^{(n)}}\right) = 0, \tag{4.23}$$

for each integrator chain starting with the base state y_i, $i = 1, 2, \ldots, m$.

Proof. By Theorem 8,

$$\lambda^{y_i} = \sum_{n=1}^{k_i} (-1)^n \frac{d^{n-1}}{dt^{n-1}}\left(\Psi_{y_i^{(n)}} + \boldsymbol{\mu}^T \mathbf{g}_{y_i^{(n)}}\right). \tag{4.24}$$

Taking the derivative of (4.24) and substituting (4.17) yields,

$$\begin{aligned}\dot{\lambda}^{y_i} &= -\Psi_{y_i} - \boldsymbol{\mu}^T \mathbf{g}_{y_i} \\ &= \sum_{n=1}^{k_i} (-1)^n \frac{d^n}{dt^n} \big(\Psi_{y_i^{(n)}} + \boldsymbol{\mu}^T \mathbf{g}_{y_i^{(n)}}\big).\end{aligned} \tag{4.25}$$

This simplifies to

$$\Psi_{y_i} + \boldsymbol{\mu}^T \mathbf{g}_{y_i} + \sum_{n=1}^{k_i} (-1)^n \frac{d^n}{dt^n} \big(\Psi_{y_i^{(n)}} + \boldsymbol{\mu}^T \mathbf{g}_{y_i^{(n)}}\big) = 0, \tag{4.26}$$

which proves Theorem 9. □

Note that while we prove Theorem 9 for the flat output space, the diffeomorphism (4.3) and (4.4) can be composed with (4.23) to generate an equivalent optimality condition in the original space. This implies that the state and costate dynamics can always be separated, and that this property is independent of the coordinate system.

Remark 5. Applying Theorem 9 to Example 1 yields the optimality equation,

$$\big(\Psi_p + \boldsymbol{\mu}^T \mathbf{g}_p\big) - \frac{d}{dt}\big(\Psi_v + \boldsymbol{\mu}^T \mathbf{g}_v\big) + \frac{d^2}{dt^2}\big(\Psi_a + \boldsymbol{\mu}^T \mathbf{g}_a\big) = 0.$$

Theorem 9 describes the evolution of the optimal state trajectory. When a constraint becomes active, i.e., strictly equal to zero, discontinuities may occur in the control action and the costates. Following the standard approach [89], these discontinuities must satisfy a set of optimality conditions,

$$\boldsymbol{\lambda}^{-T} = \boldsymbol{\lambda}^{+T} + \boldsymbol{\pi}^T \mathbf{N}_\mathbf{s}, \tag{4.27}$$

$$H^+ - H^- = \boldsymbol{\pi}^T \mathbf{N}_t, \tag{4.28}$$

$$\frac{\partial H^-}{\partial \mathbf{a}^-} = \frac{\partial H^+}{\partial \mathbf{a}^+} = \mathbf{0}, \tag{4.29}$$

where the superscripts $-$ and $+$ denote the instant in time just before and just after the activation, respectively, $\boldsymbol{\pi}$ is a constant vector of Lagrange multipliers, $\mathbf{N}$ is a vector of tangency conditions, which we rigorously derive in the following subsections, and the subscripts $\mathbf{s}$ and t correspond to partial derivatives with respect to the state and time.

Equivalently, we interpret Theorem 9 as generating optimal motion primitives for the system's trajectory. Let $\mathbf{g}$ have c linearly independent rows, then $\boldsymbol{\mu}$ is a $c \times 1$ matrix. When a constraint g_i, $i = 1, 2, \ldots, c$ does not influence the system trajectory, we can equivalently state that $\mu_i(t) = 0$. Therefore, we can generate 2^c motion primitives my setting combinations of elements in $\boldsymbol{\mu}$ equal to zero. In this context, the conditions (4.27)–(4.29) describe the optimal transition between different motion primitives along the system's trajectory. As an example, we present the optimality conditions for the unconstrained trajectory next.

Corollary 2. The evolution of the optimal unconstrained trajectory obeys

$$\sum_{j=0}^{k_i} (-1)^j \frac{d^j}{dt^j} \Psi_{y_i^{(j)}} = 0. \tag{4.30}$$

Proof. The result follows by substituting $\boldsymbol{\mu} = \mathbf{0}$ into (4.23). □

In the following sections we analyze how the state trajectory is affected by the activation of state and control constraints during operation. Note that the transition between optimal motion primitives can be written independently of the costate dynamics by applying Theorem 8 to (4.27)–(4.29).

4.1.3.2 Interior-Point Constraints

First, we will consider the case where a set of state and/or control values are imposed at a single time instant. Let $\mathbf{h}\big(\mathbf{x}(t_1), t_1\big) = 0$ describe an interior point constraint

that is imposed at some time t_1. We construct the tangency vector,

$$\mathbf{N}\big(\mathbf{x}(t),t\big) = \begin{bmatrix} \mathbf{h}\big(\mathbf{x}(t),t\big) \\ t - t_1 \end{bmatrix}, \tag{4.31}$$

which guarantees constraint satisfaction when $\mathbf{N}\big(\mathbf{x}(t_1),t_1\big) = \mathbf{0}$. Note that if the time t_1 is unknown, then (4.31) is equal to $\mathbf{h}$. The tangency vector (4.31) is directly substituted into the optimality equations (4.27) and (4.28). Finally, applying Theorem 8 to (4.27)–(4.29) yields $\sum_{i=1}^{m}\{k_i - 1\} + 1$ equations that determine the optimal change in $\mathbf{a}$ and its derivatives at t_1. These equations are independent of the costate vectors. Further manipulating (4.27)–(4.29) yields a useful pair of equations that are amenable to finding an analytical solution. First, we substitute (4.14) into (4.28) and use (4.27) to eliminate $\boldsymbol{\lambda}^-$,

$$\begin{aligned}(\Psi^+ - \Psi^-) + (\boldsymbol{\mu}^{+T}\mathbf{g}^+ - \boldsymbol{\mu}^{-T}\mathbf{g}^-) \\ + \boldsymbol{\lambda}^{+T}(\mathbf{I}^+ - \mathbf{I}^-) = \boldsymbol{\pi}^T\big(\mathbf{N}_t + \mathbf{N}_s\mathbf{I}^-\big).\end{aligned} \tag{4.32}$$

Note that, by definition, $\boldsymbol{\mu}^T\mathbf{g} = 0$ along the optimal state-trajectory, thus we set those terms equal to zero. Furthermore, the state trajectory is continuous under Assumption 17 and the integrator dynamics. Thus,

$$\mathbf{I}^+ - \mathbf{I}^- = \begin{bmatrix} \mathbf{0} \\ \mathbf{a}^+ - \mathbf{a}^- \end{bmatrix}. \tag{4.33}$$

Applying Theorem 8 to (4.32) for the case $j = k_i - 1$ and simplifying yields,

$$\begin{aligned}(\Psi^+ - \Psi^-) - (\Psi_{\mathbf{a}} + \boldsymbol{\mu}^T\mathbf{g}_{\mathbf{a}})^- \cdot (\mathbf{a}^+ - \mathbf{a}^-) \\ = \boldsymbol{\pi}^T\big(\mathbf{N}_t + \mathbf{N}_s\mathbf{I}^+\big).\end{aligned} \tag{4.34}$$

Following a similar process also implies,

$$(\Psi^+ - \Psi^-) - (\Psi_{\mathbf{a}} + \boldsymbol{\mu}^T \mathbf{g}_{\mathbf{a}})^+ \cdot (\mathbf{a}^+ - \mathbf{a}^-) = \boldsymbol{\pi}^T \big(\mathbf{N}_t + \mathbf{N}_s \mathbf{I}^-\big). \tag{4.35}$$

4.1.3.3 Trajectory Constraints

Next, we consider the case where the state and/or control constraints imposed on Problem 8 influence the trajectory of the system. To generate our optimal motion primitive using Theorem 9, we first need to ensure our constraints are functions of the state and control variables. Let $h_i\big(\mathbf{s}(t), t\big) \leq 0$ denote the $i = 1, 2, ..., c$ state or control constraints that are imposed on Problem 8. Note that h_i may not explicitly be a function of the control input. Under the standard approach [89], we require that h_i is at least q_i-times differentiable, where q_i is the minimum number of derivatives required for any component of the control input to appear in $\frac{d^{q_i}}{dt^{q_i}} h_i$. To guarantee satisfaction of the original constraint h_i, we construct the tangency vector,

$$\mathbf{N}_i(\mathbf{s}(t), t) \coloneqq \begin{bmatrix} h_i\big(\mathbf{s}(t), t\big) \\ h_i^{(1)}\big(\mathbf{s}(t), t\big) \\ \vdots \\ h_i^{(q_i - 1)}\big(\mathbf{s}(t), t\big) \end{bmatrix}, \tag{4.36}$$

and we define the constraint

$$g_i\big(\mathbf{s}(t), \mathbf{a}(t), t\big) \coloneqq h_i^{(q_i)}\big(\mathbf{s}(t), \mathbf{a}(t), t\big). \tag{4.37}$$

Thus, whenever $h_i\big(\mathbf{s}(t), t\big) = 0$ over a non-zero interval, we impose $\mathbf{N}_i\big(\mathbf{s}(t), t\big) = 0$ and $\mathbf{g}_i\big(\mathbf{s}(t), \mathbf{a}(t)\big) = 0$ over its interior to guarantee constraint satisfaction. This is equivalent to satisfying the original constraint under Assumption 17 [89]. Note that, if $\mathbf{h}_i$ is explicitly a function of the control variable, $q = 0$ and $\mathbf{N}_i$ is a zero-element

vector. Furthermore, if the constraint is active over a zero-length interval, the problem reduces to the analysis in Section 4.1.3.2 with an unknown activation time.

Finally, to construct the tangency matrix for the c constraints, we construct the stacked tangency vector,

$$\mathbf{N}\big(\mathbf{s}(t),t\big) = \begin{bmatrix} \mathbf{N}_1\big(\mathbf{s}(t),t\big) \\ \mathbf{N}_2\big(\mathbf{s}(t),t\big) \\ \vdots \\ \mathbf{N}_c\big(\mathbf{s}(t),t\big) \end{bmatrix}, \tag{4.38}$$

which accounts for all of the constraints that may influence the state and control trajectory. As with the previous section, (4.27)–(4.29) determine the required instantaneous change in the control variables and their derivatives for an optimal trajectory. Note that, by construction,

$$\boldsymbol{\pi}^T \dot{\mathbf{N}}^+ = 0, \tag{4.39}$$

as $\mathbf{N}_i = \mathbf{0}$ and $\mathbf{g}_i^+ = 0$ when constraint i is active, and the corresponding $\boldsymbol{\pi}_i = 0$ otherwise. Thus, taking the full derivative implies,

$$\boldsymbol{\pi}^T \dot{\mathbf{N}}^+ = \boldsymbol{\pi}^T \Big(\mathbf{N}_t + \mathbf{N}_{\mathbf{s}} \cdot \mathbf{I}^+ \Big) = \mathbf{0}, \tag{4.40}$$

by construction. Thus, applying (4.34) at the end of a constrained motion primitive yields

$$(\Psi^+ - \Psi^-) - (\Psi_{\mathbf{a}} + \boldsymbol{\mu}^T \mathbf{g}_{\mathbf{a}})^- \cdot (\mathbf{a}^+ - \mathbf{a}^-) = 0. \tag{4.41}$$

This leads directly to our next result,

Corollary 3. If the system exits from or enters to an unconstrained motion primitive,

the optimal control input satisfies

$$\Psi^+ - \Psi^- - \Psi^-_{\mathbf{a}}(a^+ - a^-) = 0, \text{ or} \tag{4.42}$$

$$\Psi^+ - \Psi^- - \Psi^+_{\mathbf{a}}(a^+ - a^-) = 0, \text{ respectively.} \tag{4.43}$$

Proof. When the system exits from an unconstrained motion primitive, $\boldsymbol{\mu}^- = \mathbf{0}$ and the result follows by (4.41). When the system enters an unconstrained motion primitive, $\boldsymbol{\mu}^+ = \mathbf{0}$ and $\boldsymbol{\pi} = \mathbf{0}$, thus the result follows by (4.34). □

4.1.3.4 Boundary Conditions

The results of Sections 4.1.3.2 and 4.1.3.3 completely describe the evolution of the system if the boundary conditions are known. Next, we extend this result to the case that a boundary condition is unspecified by applying Theorem 8.

Corollary 4. Let the state $y_i^{(j)}(t)$ for $i \in \{1, 2, \ldots, m\}$ and $j \in \{0, 1, 2, \ldots, k_i - 1\}$ be unspecified at a boundary, i.e., it can be arbitrarily selected. There exists an equivalent boundary condition that guarantees optimality of the system trajectory.

Proof. Without loss of generality, let the state variable $y_i^{(j)}(t)$ be undefined at the final time t^f. Under the standard approach [89], the corresponding boundary condition $\lambda^{y_i^{(j)}}(t^f) = 0$ is required to guarantee optimality. Thus, by Theorem 8,

$$\sum_{n=1}^{k_i - j} (-1)^n \frac{d^{n-1}}{dt^{n-1}} \left(\Psi_{y_i^{(j+n)}} + \boldsymbol{\mu}^T \mathbf{g}_{y_i^{(j+n)}} \right) \Big|_{t^f} = 0 \tag{4.44}$$

is an equivalent boundary condition. □

In practice, it is likely that Problem 8 will have boundary conditions defined by functions of the state variables. This may arise from undefined state variables, as with Corollary 4, or from mapping known boundary conditions to the flat output

space using Definition 13. Without loss of generality, let $\mathbf{B}(\mathbf{x}(t^f), t^f) = 0$ describe the functional constraints at t^f. This implies that [89]

$$\boldsymbol{\lambda}^T(t^f) = \left(\frac{\partial \Psi}{\partial \mathbf{s}} + \boldsymbol{\nu}\frac{\partial \mathbf{B}}{\partial \mathbf{s}}\right)_{t=t^f}, \tag{4.45}$$

$$\mathbf{B}(\mathbf{x}(t^f), t^f) = \mathbf{0}, \tag{4.46}$$

where $\boldsymbol{\nu}$ is a constant Lagrange multiplier that guarantees constraint satisfaction. Applying Theorem 8 to (4.45) results in a system of equations that guarantees constraint satisfaction at the boundaries. Thus, the flat system is guaranteed to have m initial and final conditions.

Finally, it's possible that the boundary conditions are described at an unknown terminal time. In this case, the optimal terminal time t^f satisfies [89]

$$\Omega = \left[\frac{\partial \Phi}{\partial t} + \boldsymbol{\nu}\frac{\partial \mathbf{B}}{\partial t} + \left(\frac{\partial \Phi}{\partial \mathbf{s}} + \boldsymbol{\nu}^T\frac{\partial \mathbf{B}}{\partial \mathbf{s}}\right)\mathbf{I} + \Psi\right]_{t=t^f} = 0. \tag{4.47}$$

Thus, Problem 8 always corresponds to a two-point boundary value problem with m initial conditions and m final conditions that are independent of the costates. Furthermore, (4.47) can be employed to determine an unknown terminal time using standard techniques [89]. Next, we present a numerical example for generating the trajectory of a double-integrator system in real time.

4.1.4 Double-Integrator Example

In this section we consider an agent moving in $\mathbb{R}^2$ with the states,

$$\mathbf{x}(t) = \begin{bmatrix} \mathbf{p}(t) \\ \mathbf{v}(t) \end{bmatrix}, \tag{4.48}$$

where $\mathbf{p}(t) \in \mathbb{R}^2$ is the agent's position, and $\mathbf{v} \in \mathbb{R}^2$ is the agent's velocity. We consider double-integrator dynamics,

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \mathbf{v}(t) \\ \mathbf{u}(t) \end{bmatrix}, \tag{4.49}$$

where $\mathbf{u}(t) \in \mathbb{R}^2$ is the control action, which is bounded under Assumption 17. We consider the objective,

$$J(\mathbf{x}(t), \mathbf{u}(t)) = \frac{1}{2}||\mathbf{u}(t)||^2, \tag{4.50}$$

where the coefficient $\frac{1}{2}$ simplifies the derivative of J without influencing the optimal trajectory. This illustrative example is applicable to many mobile robots, including those with unicycle dynamics (Example 1) and differential drive robots.

First, we apply Theorem 8, which yields the costates,

$$\boldsymbol{\lambda}^{\mathbf{v}}(t) = -\mathbf{u}(t), \tag{4.51}$$

$$\boldsymbol{\lambda}^{\mathbf{p}}(t) = \dot{\mathbf{u}}(t). \tag{4.52}$$

Our objective is to move the system from a fixed initial state to a final position. If the initial and final states do not activate any constraints, the corresponding boundary conditions are

$$\big(\mathbf{p}(t^0), \mathbf{p}(t^f)\big) = \big(\mathbf{p}^0, \mathbf{p}^f\big), \tag{4.53}$$

$$\big(\mathbf{v}(t^0), \mathbf{u}(t^f)\big) = \big(\mathbf{v}^0, \mathbf{0}\big). \tag{4.54}$$

Note that the value of $\mathbf{u}(t^f)$ is implied by (4.51).

Let the agent be circumscribed in a circle of radius $R > 0$, and let the environment contains c obstacles that are indexed by the set $\mathcal{C} = \{1, 2, \ldots, c\}$. Each obstacle $i \in \mathcal{C}$ is centered at the point $\mathbf{O}_i \in \mathbb{R}^2$ and has a radius $R_i \in \mathbb{R}_{>0}$. To guarantee safety,

we impose the constraints

$$h_i\big(\mathbf{p}(t)\big) = D_i^2 - \hat{\mathbf{p}}_i(t) \cdot \hat{\mathbf{p}}(t) \leq 0, \tag{4.55}$$

where $D_i \coloneqq R_i + R$ is the minimum separating distance between the agent and obstacle i and $\hat{\mathbf{p}}_i(t) \coloneqq \mathbf{p}(t) - \mathbf{O}_i$. We use the equivalent squared form of (4.55) to simplify its time derivative. To simplify our analysis, we require obstacles to satisfy the spacing constraints,

$$||\mathbf{O}_i - \mathbf{O}_j|| > D_i + D_j, \quad \forall i, j \in \mathcal{C}, i \neq j, \tag{4.56}$$

$$||\mathbf{O}_i - \mathbf{p}^0|| > D_i \text{ and } ||\mathbf{O}_i - \mathbf{p}^f|| > D_i, \tag{4.57}$$

These spacing constraints are not restrictive on our approach, (4.56) ensures that only one obstacle avoidance constraint can become active at any instant in time, and (4.57) ensure that no constraints are active at the initial or final time. The constraints that we append to the Hamiltonian are the second derivative of (4.55),

$$\mathbf{g}_i\big(\mathbf{x}(t), \mathbf{u}(t)\big) \coloneqq \mathbf{u}(t) \cdot \hat{\mathbf{p}}_i(t) + \mathbf{v}(t) \cdot \mathbf{v}(t) \leq 0. \tag{4.58}$$

Applying Theorem 9 in this context yields 2 motion primitives that completely describe the evolution of the system,

$$\text{unconstrained:} \qquad \ddot{\mathbf{u}}(t) = \mathbf{0}, \tag{4.59}$$

$$\text{safety constrained:} \qquad \ddot{\mathbf{u}}(t) + \ddot{\mu}(t)\hat{\mathbf{p}}_i(t) = \mathbf{0}. \tag{4.60}$$

Finally, we present Proposition 1, which describes the optimal transition between motion primitives.

Proposition 1. Continuity in the state variables, the $2m$ boundary conditions, the tangency conditions, and the following equations are sufficient to guarantee optimality

when the system activates the safety constraint,

$$\mathbf{u}(t_1^-) = \mathbf{u}(t_1^+), \tag{4.61}$$

$$\mathbf{p}(t_1) \cdot \mathbf{v}(t_1) = 0, \tag{4.62}$$

$$\dot{\mathbf{u}}(t_1^-) \cdot \mathbf{v}(t_1) = \dot{\mathbf{u}}(t_1^+) \cdot \mathbf{v}(t_1), \tag{4.63}$$

The proof of Proposition 1 is presented in the Appendix. In the following subsections we derive the equations of motion for each of the motion primitives.

4.1.4.1 Optimal Motion Primitives

Under our imposed obstacle spacing (4.56), we must only consider two possible cases: no constraints influence the agent's trajectory, or avoiding one obstacle influences the agent's trajectory.

Case I: Unconstrained Motion. When the safety constraint is not active, the optimal control input is zero-snap, i.e., integrating (4.59) yields,

$$\mathbf{u}(t) = 6\mathbf{a}t + 2\mathbf{b}, \tag{4.64}$$

$$\mathbf{v}(t) = 3\mathbf{a}t^2 + 2\mathbf{b}t + \mathbf{c}, \tag{4.65}$$

$$\mathbf{p}(t) = \mathbf{a}t^3 + \mathbf{b}t^2 + \mathbf{c}t + \mathbf{d}, \tag{4.66}$$

where the constants of integration $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ are determined by the boundary conditions. Thus, each unconstrained motion primitives introduces 8 unknowns.

Case II: Constrained Motion When the safety constraint is active, the tangency conditions imply that

$$\hat{\mathbf{p}}(t) \cdot \mathbf{v}(t) = 0. \tag{4.67}$$

Thus, taking the dot product of (4.60) with $\mathbf{v}(t)$ implies,

$$\ddot{\mathbf{u}}(t) \cdot \mathbf{v}(t) = 0. \tag{4.68}$$

Applying integration by parts to (4.68) yields,

$$\dot{\mathbf{u}}(t) \cdot \mathbf{v}(t) = C + \frac{1}{2}\mathbf{u}(t) \cdot \mathbf{u}(t), \tag{4.69}$$

where C is a constant of integration. Similarly, taking a derivative of (4.55) yields,

$$\dot{\mathbf{u}}(t) \cdot \mathbf{p}(t) = -3\mathbf{u}(t) \cdot \mathbf{v}(t). \tag{4.70}$$

Note that, by Proposition 1, each term in (4.69) and the right-hand side of (4.70) are both continuous. In addition, $\hat{\mathbf{p}}(t)$ and $\mathbf{v}(t)$ constitute an orthogonal basis for the system's trajectory. Thus, the constrained motion primitive is completely determined by the state of the system the instant it transitions, and it introduces only two unknowns—the entry and exit times.

Finally, for completeness, we must describe how the system transitions from the constrained motion primitive to the unconstrained motion primitive. In this case there are no active constraints, thus $\boldsymbol{\pi} = \mathbf{0}$ in (4.27). Substituting (4.51) and (4.52) into (4.27) implies continuity in $\mathbf{u}(t)$ and $\dot{\mathbf{u}}(t)$, which, alongside continuity in the state, determines the unconstrained trajectory. Next, we solve these reduced systems of equations numerically and compare the performance to standard numerical solvers.

4.1.5 Numerical Simulation

To demonstrate the performance of our approach, we generated trajectories for the double integrator agent around a single obstacle. The parameters describing the simulation are presented in Table 4.1, and the optimal trajectory is annotated in Fig. 4.1. For further details on our simulation approach, please see our public code

repository on github[1].

$\mathbf{p}^0$ (m)	$\mathbf{v}^0$ (m/s)	$\mathbf{p}^f$ (m)	$\mathbf{O}_i$ (m)	D_i (m)	$t^f - t^0$ (s)
$(-2, -2)$	$(0, 0)$	$(1, 2)$	$(0, 0)$	1.25	10

Table 4.1: Boundary conditions and obstacle parameters that describe the simulation environment.

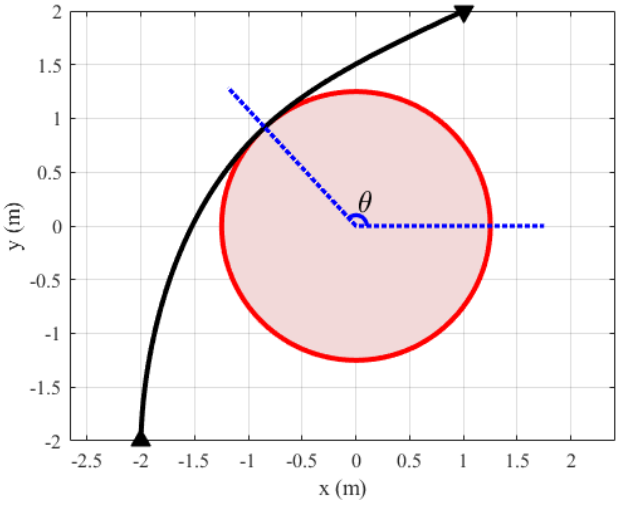

Figure 4.1: Optimal trajectory (black) that avoids the obstacle (red). The variable θ describes where the trajectory instantaneously contacts the obstacle at time t_1.

We apply the following steps to generate the optimal trajectory of the agent around the obstacle:

1. Connect the boundary conditions with an unconstrained motion primitive.
2. If the unconstrained trajectory is infeasible, find the optimal trajectory assuming the safety constraint is only activated instantaneously.
3. If the previous case is infeasible, find the optimal trajectory assuming the safety constraint is activated over a non-zero interval.

[1] https://github.com/UD-IDS-LAB/Flatness-Obstacle_Avoidance

The first step consists of solving a set of 8 linear equations to determine the coefficients of the unconstrained motion primitive. This can be precomputed offline and has a negligible computational cost as a consequence. In the second step, we assume that the agent only contacts the obstacle instantaneously. The entire trajectory can be parameterized by two variables, (θ, t_1); Fig. 4.1 shows how θ defines the agent's trajectory relative to the obstacle. In the third step, we assume that the agent travels along the constrained motion primitive over a non-zero interval of time. The entire trajectory can be parameterized by three variables, (θ, t_1, t_2).

After each step, we check if the resulting trajectory satisfies the collision avoidance constraint. If the initial unconstrained trajectory is infeasible, we use the time and angle of the constraint violation to calculate an initial guess for (θ, t_1) in Step 2. If the trajectory generated in Step 2 is infeasible, we reuse the values of θ and t_1, as well as the earliest time that the constraint was violated, as an initial guess for (θ, t_1, t_2). To demonstrate the performance of our approach, we generated the optimal state trajectory for the agent in Matlab 2018b, using *fmincon* to solve the system of constrained nonlinear equations. We simulated the system on a desktop computer (Intel I5-3570k @3.4 GHz), and compared the performance to two general-purpose optimal control packages [56, 57]. We averaged the computation time over 10 trials; note that existing solvers use collocation methods, thus they require a large number of mesh points to guarantee constraint satisfaction near the obstacle. In contrast, our solution only saves the states at the boundary and motion primitives transitions. Table 4.2 demonstrates that our algorithm generates a superior trajectory in less time and with fewer mesh points stored in memory.

Approach	Time	Mesh Points	Cost	Max Violation
Proposed	**34** ms	**3**	**0.064** m/s^3	**0** m
ICLOCS2	1500 ms	35	0.080 m/s^3	3×10^{-4} m
OpenOCL	98 ms	35	0.080 m/s^3	4×10^{-15} m

Table 4.2: Performance comparison of the generated trajectories.

4.1.6 Proofs

Proof of Proposition 1. There are two cases when system can transition between the constrained and unconstrained motion primitives.

Case I: Instantaneous Constraint Activation. In this case, the collision avoidance constraint is only activated instantaneously at some unknown time t_1. Following the analysis in Section 4.1.3.2, and given the obstacle spacing constraint (4.56), we must only consider a single tangency equation,

$$\mathbf{N}\big(\mathbf{x}(t), t\big) = \big[D^2 - \hat{\mathbf{p}}_i(t) \cdot \hat{\mathbf{p}}_i(t)\big]. \tag{4.71}$$

The boundary conditions yield 8 equations, which can satisfy the 8 unknowns that describe one unconstrained motion primitive. When the safety constraint is activated instantaneously, the agent transitions between unconstrained motion primitives. This introduces 9 additional unknowns—the 8 coefficients of the new unconstrained motion primitive and the unknown transition time, t_1. Proposition 1 provides the corresponding 9 equations: 4 from continuity in the state, 1 from (4.71), and 4 from (4.61)–(4.63). Condition (4.61) follows trivially from substituting (4.51) and (4.52) into (4.27), which yelds,

$$\mathbf{u}^+ - \mathbf{u}^- = 0, \tag{4.72}$$

$$\dot{\mathbf{u}}^+ - \dot{\mathbf{u}}^- = 2\mathbf{p}_i(t_1)\pi. \tag{4.73}$$

Furthermore, (4.35) implies that

$$0 = -2\pi\hat{\mathbf{p}}(t_1) \cdot \mathbf{v}(t_1). \tag{4.74}$$

This implies that either 1) $\pi = 0$ or 2) $\mathbf{p}(t_1) \cdot \mathbf{v}(t_1) = 0$. Assuming $\pi = 0$ implies that $\dot{\mathbf{u}}$ is continuous at t_1, and thus both constrained motion primitives can be replaced with a single unconstrained motion primitive. This contradicts our premise that a constraint

becomes active at t_1. Therefore, the only possible solution is $\pi \neq 0$. This implies that $\mathbf{p}(t_1) \cdot \mathbf{v}(t_1) = 0$, which also implies (4.63) by taking dot product of (4.73) with $\mathbf{v}(t_1)$.

Case II: Transition to Constrained Motion Primitive. In this case, the system transitions to a constrained motion primitive at t_1, and it transitions back to another unconstrained motion primitive at some time $t_2 > t_1$. Following the analysis in Section 4.1.3.3, we must satisfy,

$$\mathbf{N}\big(\mathbf{x}(t), t\big) = \begin{bmatrix} D_i^2 - \hat{\mathbf{p}}_i(t) \cdot \hat{\mathbf{p}}_i(t) \\ \mathbf{v}(t) \cdot \hat{\mathbf{p}}_i(t) \end{bmatrix} = \mathbf{0}, \tag{4.75}$$

$$\mathbf{g}\big(\mathbf{x}(t), \mathbf{u}(t)\big) = \mathbf{u}(t) \cdot \hat{\mathbf{p}}_i(t) + \mathbf{v}(t) \cdot \mathbf{v}(t) = 0, \tag{4.76}$$

over the open interval (t_1, t_2). Thus, (4.62) is true via the tangency conditions. Corollary 3 implies,

$$\begin{aligned} &||\mathbf{u}(t_1^+)||^2 - ||\mathbf{u}(t_1^-)||^2 - 2\mathbf{u}(t_1^-) \cdot \big(\mathbf{u}(t_1^+) - \mathbf{u}(t_1^-)\big) \\ &\quad = ||\mathbf{u}(t_1^+)||^2 + ||\mathbf{u}(t_1^-)||^2 - 2\mathbf{u}(t_1^+) \cdot \mathbf{u}(t_1^-) = 0, \end{aligned} \tag{4.77}$$

which implies that $\mathbf{u}(t)$ is continuous at t_1 [59]. Finally, substituting (4.52) into (4.27) yields,

$$\dot{\mathbf{u}}(t_1^-) = \dot{\mathbf{u}}(t_1^+) + 2\hat{\mathbf{p}}(t_1)\pi_1\pi_2, \tag{4.78}$$

which implies (4.68) by dotting (4.78) with $\mathbf{v}(t_1)$. □

4.2 Experimental Validation of a Real-Time Optimal Controller for Coordination of CAVs in a Multi-Lane Roundabout

In this section I present a subset of the analysis from an earlier paper [58], which involves the optimal coordination of connected and automated vehicles in a multi-lane intersection. The optimal control problem is identical to the previous example in 1D: each vehicle uses double-integrator dynamics (4.49), their cost is the L^2 norm of the

control input, and the final velocity is left free. Rather than generating the entire constrained trajectory, the vehicles select the minimum final time t_i^f such that the unconstrained trajectory is feasible; this is also a tradeoff between travel time and energy consumption. This is in contrast to existing work, where the final time is selected by a coordinator [117, 128], possibly with vehicle re-sequencing [129], and each CAV generates its own feasible energy-minimizing trajectory.

4.2.1 The Roundabout Scenario

In this paper, we consider a multi-lane roundabout with three CAV inflows and three areas where lateral collisions between CAVs may occur, as shown in Fig. 4.2. However, our proposed solution does not depend on the specific paths presented in this work and can be applied in any scenario in a roundabout. To navigate the roundabout, we define a *control zone*, which starts upstream from the roundabout and ends at each roundabout exit (Fig. 4.2). The control zone has an associated *coordinator*, which stores information about the geometry of the roundabout and the trajectory information of each CAV in the control zone. The coordinator does not make any decisions and only acts as a database.

Let $\mathcal{Q}(t) \subset \mathbb{N}$ be the set of CAVs at time $t \in \mathbb{R}$ which are inside the control zone. Upon entering the control zone at time $t_i^0 \in \mathbb{R}$, CAV $i \in \mathcal{Q}(t)$ retrieves the trajectory information of every other CAV $j \in \mathcal{Q}(t) \setminus \{i\}$ and generates an energy-optimal safe trajectory through the control zone. Then, CAV i broadcasts its trajectory to the coordinator. Finally, when CAV i exits the control zone at time t_i^f, it is removed from the set $\mathcal{Q}(t)$. A detailed discussion about the communication of the coordinator with the CAVs is presented in [38, 39].

Definition 15. We define each lateral node with a unique index, $n \in \{1, 2, 3\}$, at each area inside the roundabout where there might be a potential lateral collision (Fig. 4.2).

Definition 16. For each CAV $i \in \mathcal{Q}(t)$, we define $\mathcal{N}_i$ as the set of all lateral nodes on the path of CAV i.

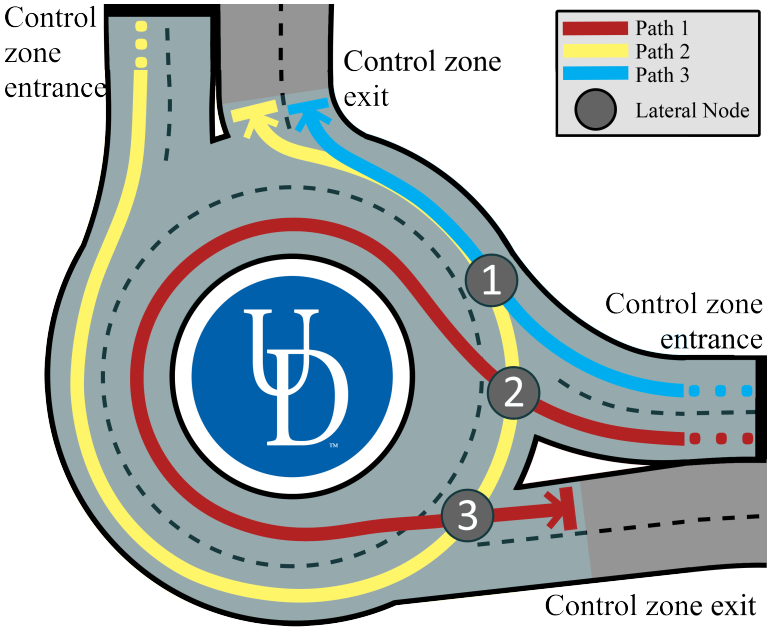

Figure 4.2: A schematic of the roundabout scenario. The highlighted control zone continues upstream from the roundabout.

For instance, if CAV i is traveling along the path 1 (Fig. 4.2), the set of collision nodes is $\mathcal{N}_i = \{2, 3\}$.

Definition 17. For each CAV $i \in \mathcal{Q}(t)$, upon entering the control zone at time t_i^0, we define the set of lateral nodes shared with each CAV $j \in \mathcal{Q}(t_i^0) \setminus \{i\}$ as,

$$\mathcal{C}_{i,j} = \{n \mid n \in \mathcal{N}_i \cap \mathcal{N}_j\}. \tag{4.79}$$

4.2.2 Vehicle Model and Constraints

Each CAV obeys double-integrator dynamics (4.49); the control input and speed of each CAV $i \in \mathcal{Q}(t)$ at time $t \in [t_i^0, t_i^f]$ are bounded by,

$$\begin{aligned} u_{\min} \leq u_i(t) \leq u_{\max}, \\ 0 < v_{\min} \leq v_i(t) \leq v_{\max}, \end{aligned} \tag{4.80}$$

where $u_{\min}$, $u_{\max}$ are the minimum deceleration and maximum acceleration, and $v_{\min}$, $v_{\max}$ are the minimum and maximum speed limits respectively.

Definition 18. For any CAV $i \in \mathcal{Q}(t)$, if there exists a CAV $k \in \mathcal{Q}(t)$ which leads CAV i, we define $d_i(t)$ as the bumper-to-bumper distance from CAV k to CAV i. If no such CAV k leads CAV i, then we let $d_i(t) \to \infty$.

To guarantee no rear-end collision occurs between CAV $i \in \mathcal{Q}(t)$ and the preceding CAV $k \in \mathcal{Q}(t)$, we impose the following rear-end safety constraint,

$$d_i(t) \geq \delta_i(t), \tag{4.81}$$

where $\delta_i(t)$ is the safe distance that depends on CAV's speed,

$$\delta_i(t) = \gamma + \varphi\, v_i(t), \tag{4.82}$$

where $\gamma, \varphi \in \mathbb{R}$ are the standstill distance and reaction time, respectively.

For each CAV $i \in \mathcal{Q}(t)$, the distance to a node $n \in \mathcal{N}_i$ is denoted by the function $l_i : \mathcal{N}_i \to \mathcal{P}_i$. To guarantee lateral collision avoidance, we impose the following time headway constraint for every CAV $i \in \mathcal{Q}(t)$,

$$\begin{aligned} &\left| p_i^{-1}(l_i(n)) - p_j^{-1}(l_j(n)) \right| \geq t_h, \\ &\forall\, n \in \mathcal{C}_{i,j}, \quad \forall j \in \mathcal{Q}(t) \setminus \{i\}, \end{aligned} \tag{4.83}$$

where $t_h \in \mathbb{R}^+$ is the minimum time headway between any two vehicles entering node n. Note that, as position is strictly-increasing for all $t \in [t_i^0, t_i^f]$, the inverse position (4.83) has a closed-form representation [121].

Remark 6. The lateral safety constraint (4.83) relaxes the first-in-first-out coordination policy for CAVs $i, j \in \mathcal{Q}(t)$, which is common in the literature.

Next, we formulate a decentralized optimal control problem for each CAV $i \in \mathcal{Q}(t)$ in order to minimize their energy consumption over the interval $t \in [t_i^0, t_i^f]$.

Problem 9. When a CAV $i \in \mathcal{Q}(t)$ enters the control zone, it solves the following optimal control problem:

$$\begin{aligned} \min_{u_i(t)\in\mathcal{U}_i} \quad & \tfrac{1}{2}\int_{t_i^0}^{t_i^f} u_i^2(t)\ dt, \\ \textit{subject to}: \quad & (4.49), (4.80), (4.81), (4.83), \\ \textit{given} \quad & p_i(t_i^0), v_i(t_i^0),\ p_i(t_i^f). \end{aligned}$$

As an alternative to the standard approach, we require each CAV to solve Problem 9 using the unconstrained motion primitive by selecting the minimum feasible t_i^f. This has the advantage of guaranteed energy-optimality while being significantly easier to implement on a real vehicle. We employ the optimal unconstrained motion primitive (4.64)–(4.66), which yields,

$$p_i(t) = a_i t^3 + b_i t^2 + c_i t + d_i, \tag{4.84}$$

$$v_i(t) = 3a_i t^2 + 2b_i t + c_i, \tag{4.85}$$

$$u_i(t) = 6a_i t + 2b_i, \tag{4.86}$$

with the boundary conditions

$$p_i(t_i^0) = p_i^0,\ v_i(t_i^0) = v_i^0,\ p_i(t_i^f) = p_i^f,\ u_i(t_i^f) = 0, \tag{4.87}$$

where $u_i(t_i^f) = 0$ results from the velocity being unspecified at t_i^f.

To derive an energy-optimal control input that can be computed in real time, we seek to minimize the exit time of CAV $i \in \mathcal{Q}(t)$ from the control zone and impose (4.84) - (4.86) as an *energy-optimal motion primitive*. This results in a new energy and time-optimal scheduling problem [121].

Problem 10. When a CAV $i \in \mathcal{Q}(t)$ enters the control zone, it derives its minimum travel time such that the resulting trajectory is unconstrained and does not violate any state, control, or safety constraints.

$$\begin{aligned} \min_{a_i, b_i, c_i, d_i} \quad & t_i^f, \\ \textit{subject to:} \quad & (4.80), (4.81), (4.83), \\ & (4.84), (4.85), (4.86), (4.87). \end{aligned}$$

The solution of Problem 10 yields the minimum t_i^f such that the generated trajectory is the unconstrained optimal solution to Problem 9. Next, we provide the assumptions we imposed in our approach on each CAV $i \in \mathcal{Q}(t)$.

Assumption 18. There are no errors or delays in the vehicle-to-vehicle and vehicle-to-infrastructure communication.

Assumption 19. Vehicle-level control is handled by a low-level controller which can perfectly track the trajectory generated by solving Problem 10.

The first assumption ensures that we address the deterministic case. It is relatively straightforward to relax this assumption as long as the noise or delays are bounded [92]. The second assumption is to decouple the motion planning and vehicle control, which makes the problem tractable. By tuning the low-level controller, it can be ensured that the prescribed trajectory is followed.

4.2.3 Analytical solution

We seek to transform Problem 10 into an equivalent formulation with a single optimization variable. First, without loss of generality, we consider the domain of Problem 10 to be $t \in [0, t_i^f]$ and $p_i(t) \in [0, S_i]$, where S_i is the length of the control

zone corresponding to CAV i's path. This results in a new set of boundary conditions,

$$p_i(t_i^0 = 0) = 0, \qquad v_i(t_i^0 = 0) = v_i^0, \tag{4.88}$$

$$p_i(t_i^f) = S_i, \qquad u_i(t_i^f) = 0. \tag{4.89}$$

Next, we substitute (4.88) into (4.84) and (4.85), yielding

$$p_i(t_i^0 = 0) = d_i = 0, \tag{4.90}$$

$$v_i(t_i^0 = 0) = c_i = v_i^0, \tag{4.91}$$

which must always hold for Problem 10. Next, we substitute (4.89) into (4.86), which yields, $u_i(t_i^f) = 6a_i t_i^f + 2b_i = 0$. This implies that

$$a_i = -\frac{b_i}{3t_i^f}. \tag{4.92}$$

Next, we substitute (4.90)–(4.92) into (4.84), yielding

$$p_i(t_i^f) = -\frac{b_i}{3t_i^f} {t_i^f}^3 + b_i {t_i^f}^2 + v_i^0 t_i^f = S_i. \tag{4.93}$$

Hence, equation (4.93) simplifies to

$$b_i = \frac{3(S_i - v_i^0 t_i^f)}{2{t_i^f}^2}. \tag{4.94}$$

We may further simplify Problem 10 by finding a compact domain of feasible t_i^f by considering the speed and control constraints. Let $t_{i,\min}^f$ and $t_{i,\max}^f$ denote the lower bound and upper bound on t_i^f respectively, which is imposed by the state and control constraints.

Proposition 2. For each CAV $i \in \mathcal{Q}(t)$, the lower bound on exit time of the control

zone, $t^f_{i,\min}$, is computed as follows

$$t^f_{i,\min} = \min\{t^f_{i,u_{\max}}, t^f_{i,v_{\max}}\}, \tag{4.95}$$

where

$$t^f_{i,u_{\max}} = \frac{\sqrt{9(v^0_i)^2 + 12S_i u_{\max}} - 3v^0_i}{2u_{\max}}, \tag{4.96}$$

$$t^f_{i,v_{\max}} = \frac{3S_i}{v^0_i + 2v_{i,\max}}. \tag{4.97}$$

Proof. There are two cases to consider: **Case 1:** CAV i achieves its maximum control input at entry of the control zone, when $u_i(t^0_i) = u_{\max}$ and $u_i(t^f_i) = 0$. **Case 2:** CAV i achieves its maximum speed at the end of control zone, as $v_i(t)$ is strictly increasing and $v_i(t^f_i) = v_{\max}$.

In case 1, by (4.86), we have

$$u_i(t^0_i = 0) = 2b_i = u_{\max}. \tag{4.98}$$

Substituting (4.94) into (4.98) and solving for t^f_i, yields the quadratic equation

$$u_{\max} {t^f_i}^2 + 3v^0_i t^f_i - 3S_i = 0, \tag{4.99}$$

which has two real roots with opposite signs, as $t^f_{i,1} t^f_{i,2} = \frac{-3S_i}{u_{\max}} < 0$. Thus, $t^f_{i,u_{\max}} > 0$ is computed by

$$t^f_{i,u_{\max}} = \frac{\sqrt{9(v^0_i)^2 + 12S_i u_{\max}} - 3v^0_i}{2u_{\max}}. \tag{4.100}$$

For case 2, by (4.85), we have

$$v_i(t^f_i) = \frac{a_i}{3} {t^f_i}^2 + \frac{b_i}{2} t^f_i + v^0_i = v_{\max}. \tag{4.101}$$

Substituting (4.92) and (4.94) into (4.101) yields

$$v_i(t_i^f) = 3\Big(\frac{-b_i}{3t_i^f}\Big){t_i^f}^2 + 2b_i t_i^f + v_i^0 \tag{4.102}$$
$$= b_i t_i^f + v_i^0 = \frac{3(S_i - v_i^0 t_i^f)}{2t_i^f} + v_i^0 = v_{\max},$$

which simplifies to

$$t_{i,v_{\max}}^f = \frac{3S_i}{v_i^0 + 2v_{\max}}. \tag{4.103}$$

Thus, our lower bound on t_i^f is given by

$$t_{i,\min}^f = \min\{t_{i,u_{\max}}^f, t_{i,v_{\max}}^f\}. \tag{4.104}$$

□

Proposition 3. For each CAV $i \in \mathcal{Q}(t)$, the upper bound on exit time of the control zone, $t_{i,\max}^f$, is computed as follows

$$t_{i,\max}^f = \begin{cases} t_{i,v_{\min}}, & \text{if } \; 9{v_i^0}^2 + 12S_i u_{i,\min} < 0, \\ \max\{t_{i,u_{\min}}^f, t_{i,v_{\min}}^f\}, & \text{otherwise.} \end{cases} \tag{4.105}$$

where

$$t_{i,v_{\min}} = \frac{3S_i}{v_i^0 + 2v_{\min}}, t_{i,u_{\min}} = \frac{\sqrt{9(v_i^0)^2 + 12S_i u_{\min}} - 3v_i^0}{2u_{\min}}. \tag{4.106}$$

Proof. Similar steps to Proposition 2 can be followed to find the upper bound for t_i^f. Note that when $9(v_i^0)^2 + 12S_i u_{\min} < 0$ there is no real value of t_i^f which satisfies all of the boundary conditions simultaneously. In this case the lower bound is given by the $v_{\min}$ case. The proof for the $v_{\min}$ case is identical to Proposition 3 and is omitted. □

Finally, we may write an equivalent formulation of Problem 10, which optimizes a single variable, t_i^f over a compact set $[t_{i,\min}^f, t_{i,\max}^f]$.

Problem 11. When CAV $i \in \mathcal{Q}(t)$ enters the control zone it derives the minimum exit time such that the resulting unconstrained trajectory does not violate any safety constraints.

$$\begin{aligned} & \min_{t_i^f} t_i^f, \\ \textit{subject to:} \quad & (4.81), (4.83), \\ & (4.90), (4.91), (4.92), (4.94), \\ & t_i^f \in [t_{i,\min}^f, t_{i,\max}^f]. \end{aligned}$$

4.2.4 Experimental Results

To validate our proposed controller in our scaled smart city, we collected several pieces of data throughout five experiments. First, the position, speed, and a timestamp for each CAV was streamed back to the mainframe at a rate of 20 Hz. Furthermore, the state and time of each CAV entering the control zone were recorded, as well as the computed and achieved exit time. These results are summarized in Table 4.3. Note that the minimum speed of any CAV across all five experiments is 0.12 m/s (7 mph at full scale), which demonstrates that stop and go driving has been completely eliminated. Additionally, the average CAV speed is 0.42 m/s (24 mph at full scale), which implies that most CAVs are traveling near $v_{\max}$ and must apply minimal control effort.

Table 4.3: Average velocity and travel time results for the 5 experiments. RMSE is normalized by travel time for each CAV.

Experiment	$v_{\min}$ [m/s]	v_{avg} [m/s]	Travel Time RMSE
1	0.16	0.41	2.71 %
2	0.27	0.45	1.54 %
3	0.18	0.41	4.03 %
4	0.12	0.43	1.92 %
5	0.21	0.42	1.38 %

The exit time data for each CAV is visualized in Fig. 4.3, where the grey bars represent the feasible space of t_i^f, the wide black bars correspond with the solution of

Problem 11, and the thin red bars show the achieved exit time for each CAV. From Table 4.3, the error between desired and actual exit time varies between $2-4\%$. This error comes from the CAV's ability to track the desired trajectory and shows that Assumption 19 is reasonable for well-tuned CAVs in our testbed.

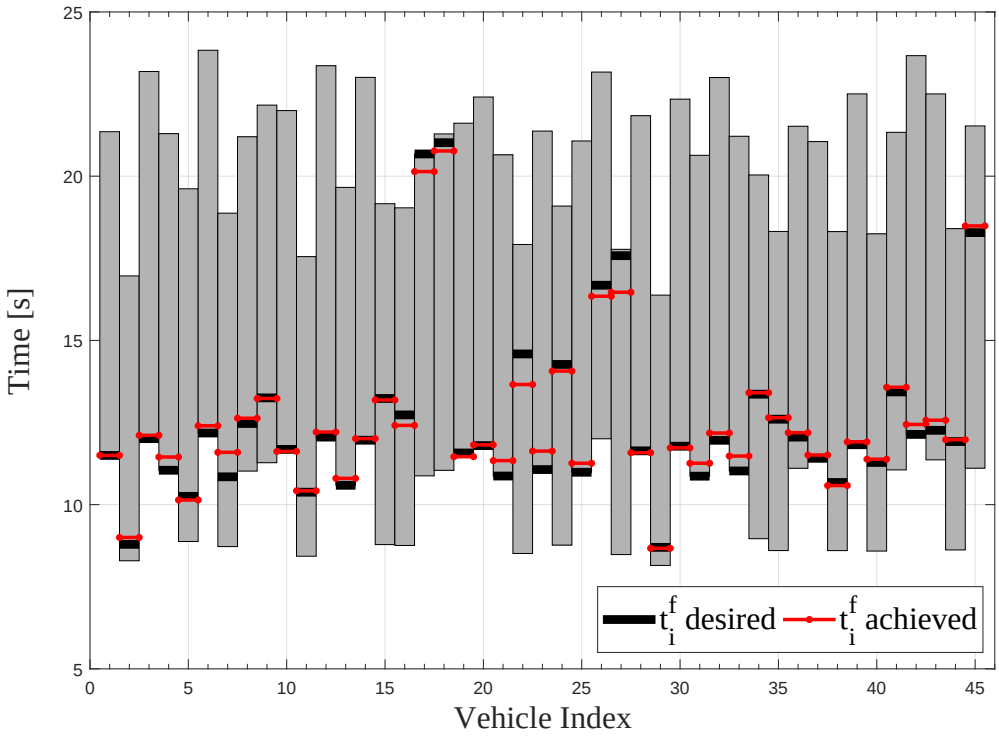

Figure 4.3: Estimated and actual arrival time for each vehicle over all experiments.

The position trajectory of an ego-CAV following path 2 is given in Fig. 4.4. The ego-CAV's position is denoted by the dashed red line, while the positions of two other CAVs are represented by dotted black lines. The lateral collision constraints are denoted by vertical black bars, and the rear-end safety constraint is the hashed region on the graph. There are two other CAVs shown; one is on Path 3 and merges in front of the ego-CAV at collision node 1 (Fig. 4.2) and the second CAV leads the ego-CAV on path 2. Figure 4.4 demonstrates that although the trajectory generated by the ego-vehicle may not violate any constraints, the actual trajectory violates the

rear-end safety constraint by a car length (0.2 m). However, at this speed, the rear-end safety constraint requires a three-car length gap, so a robust control formulation of Problem 11 could likely guarantee collision avoidance. This can also be seen in the lateral collision avoidance constraint in Fig. 4.4, where a later CAV crosses node 3 in a way that violates the time headway constraint (again, without leading to an actual collision).

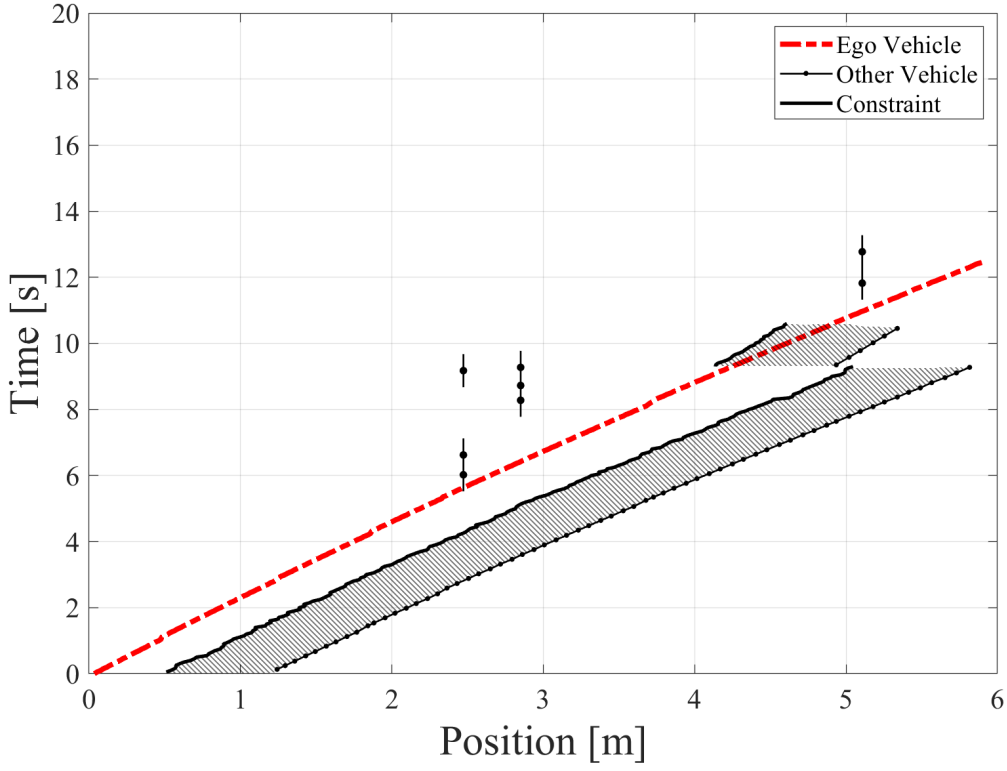

Figure 4.4: Position trajectory for the third vehicle entering from path 2 in the 5th experiment. The lateral constraints are shown as vertical lines, and the rear-end safety constraint is the hashed region.

4.3 Conclusion

This chapter presents an original technique to generate optimal trajectories for differentially flat systems, which outperforms existing approaches by every metric. First, we derived an optimality condition that describes the optimal state evolution

independently of the costates. Second, we applied Theorem 8 to derive additional boundary conditions for the flat system, which, to the best of our knowledge, has not yet been reported in the literature. Third, we proposed a motion primitive generator in Theorem 9 and derived the conditions to optimally switch between different motion primitives. Finally, we applied our results to a double-integrator system and generated a minimum-control trajectory with 20% less energy consumption and 2.8 times faster computational speed compared to the best existing solver. In addition, an earlier analysis on connected and automated vehicles [58] demonstrates how the resulting optimal control problem can be further simplified down to a one dimensional system that can be solved in real-time to control a multi-robot system. In fact, our most recent experiment generated unconstrained trajectories in approximately 2 ms for traffic corridor containing 15 CAVs [39].

There are several intriguing directions for future work. First, it is practical, for given dynamics, to determine what functional forms of the objective guarantee that an analytical solution to the motion primitive generator (4.23) exists. Another potential direction for future research is to relax Assumptions 16 and 17 and derive similar results for systems with singularities and unbounded actuation capabilities. Developing a general-purpose numerical method for differentially flat systems is another potential research direction, and implementing the solver on embedded hardware for multi-robot experiments will likely yield fruitful insights. Finally, it is important to note that the constraint violations seen in our experiment stem almost entirely from model mismatch and tracking error; tackling this problem is critical for the application of flatness-based methods to physical systems [130].

Chapter 5

CONCLUSIONS AND FUTURE WORK

> I learned this, at least, by my experiment: that if one advances confidently in the direction of his dreams, and endeavors to live the life which he has imagined, he will meet with a success unexpected in common hours.
>
> Henry David Thoreau
> Walden (1854)

Now, more than ever, it is critical that we develop tools to help us understand the behavior of the complex and interconnected systems that make up our everyday lives. The contributions of the work presented in this dissertation look to advance the state-of-the-art in constraint-driven control, and to bring multi-agent robotic systems one step closer to physical realization. Additionally, this dissertation provides a rigorous framework to analyze emergent phenomena in complex systems, and it provides some insights in how these systems might be effectively designed and controlled.

In terms of engineering outcomes, this dissertation proposes a new framework to control multi-agent systems, which provides strong guarantees on the safety of individual agents and the behavior of the overall system. This framework is fundamentally data-driven, in the sense that each agent's actions are a function of external signals produced by the environment and neighboring agents. In contrast to machine learning techniques, constraint-driven control has strong safety and performance guarantees; additionally, the behavior of each agent is interpretable, i.e., it is straightforward for a designer to understand why an agent takes any particular action. Furthermore, this

dissertation demonstrates the deployment of the proposed algorithms in a real-time control application using our 1:25 scale testbed for emergent mobility systems.

As for broader impacts, this dissertation represents another union between emergence in complex systems and control theory. Using the definition of emergence given by Ashby [4], it is straightforward to determine whether a multi-agent system will generate emergent behavior. The questions raised by Ashby can be interrogated directly using the engineering design process; the "complexity" of a system corresponds to the size of the sub-space containing all equilibrium points, and the "goodness" of a system corresponds to how well the system achieves a particular objective function. In fact, the language of dynamical systems, stability, and optimization seems particularly well-suited for rigorous discussion of emergent behavior in complex systems.

5.1 Summary of Contributions

Chapter 2 presents work on the formation reconfiguration problem, where a collection of agents must move into a desired formation. This problem is well-studied, and there are many heuristic techniques to achieve a desired formation. Viewing this problem through Ashby's framework allows us to apply engineering design principles directly to a problem of emergence. In particular, it allows us to weigh the benefits (in terms of convergence time) versus the cost (in terms of communication and actuation) of different interaction rules to achieve desirable emergence. The major contributions of this chapter are:

1. An equivalence between the assignment problem and Ashby's definition of emergence.
2. A new, decentralzied approach to goal assignment.
3. A guarantee that all agents converge to a unique goal in finite time under reasonable assumptions about the initial and goal states.

4. An optimization-based approach to goal selection, which minimizes the expected energy consumption of the individual agents—particularly in the case of dynamic moving goals.

Chapter 3 presents several contributions to constraint-driven control of multi-agent systems. Constraint-driven techniques allow agents to optimally decide their control action at each time-instant, but they also require guarantees that the space of feasible actions is non-empty. While this may require additional effort in terms of designing the appropriate constraints, it is not inherently more work than moving all of the constraints into the objective and selecting appropriate weights to achieve a desired behavior [131, 132]. The major contributions of this chapter are:

1. An original framework, with three examples, for constraint-driven control of multi-agent systems.
2. Sufficient conditions on the local states of ground vehicles to guarantee the emergence of platooning behavior at the system level.
3. The first article, to the best of our knowledge, demonstrating how the physics of fixed-wing UAVs can be exploited to generate stable emergent V formations.
4. A novel switching system architecture, which is interpretable without sacrificing the strong guarantees on safety and system-level behavior.

Chapter 5 proposes an original technique for the optimal control of constrained continuous-time systems; increasingly more efficient techniques will be critical before multi-agent systems can be deployed en masse. Additionally, for systems with differentially flat dynamics, this method allows designers to plan exact trajectories using a set of simplified dynamics. The results of this chapter are coordinate-independent, which implies that the separation result is a fundamental property of differentially flat systems—it is not just a result of the transformation. The major contributions of this chapter are:

1. A differential equation that is independent of the costates, which can be used to generate optimal motion primitives.
2. An equivalent set of optimality conditions that describe how the system reacts to constraint activations.
3. A numerical example of energy-optimal obstacle avoidance that significantly outperforms state-of-the-art open source optimal control libraries for one example.
4. Experimental validation of an unconstrained motion primitive being used to coordinate a system of connected and automated vehicles at an unsignalized roundabout to eliminate stop-and-go driving.

5.2 Future Research Directions

While this dissertation provides some insights on multi-agent systems and emergence, it also raises a number of compelling questions. From an engineering perspective, there are many unknowns that must be resolved before multi-agent and robotic swarm systems become commonplace. First, the cost of individual robots must be significantly reduced, either through Moore's law or clever algorithms, to make multi-agent systems cost-effective. Another interesting research direction is demonstrating the benefit of line flocking with aerial vehicles; UAVs with broad wingspans traveling in calm air show the most promise. A more theoretical question concerns emergence in competitive games—how might the emergence of a peloton in a bicycle race occur? This research direction seems amenable to a differential games approach, where Nash equilibria correspond to the emergent behavior of the cooperative setting. Finally, can an emergent system, as defined by Ashby, be abstracted to multiple layers? I have demonstrated, for example, that individual vehicles can be abstracted to platoons; might these platoons contain their own collection of equilibrium points, which can be further abstracted to a general transportation network? Such an analysis would be the greatest test of this dissertation's framework, to find a class of systems that can be repeatedly abstracted until only trivial equilibria remain. Or, in the tradition of Richardson and De Morgan, "Systems with equilibria that describe their complexity, their abstractions are the same, and so on to simplicity."

BIBLIOGRAPHY

[1] Henrik Christensen, Nancy Amato, Holly Yanco, Maja Mataric, Howie Choset, Ann Drobnis, Ken Goldberg, Jessy Grizzle, Gregory Hager, John Hollerbach, Seth Hutchinson, Venkat Krovi, Daniel Lee, William D. Smart, Jeff Trinkle, and Gaurav Sukhatme. A Roadmap for US Robotics – From Internet to Robotics 2020 Edition. *Foundations and Trends in Robotics*, 8(4):307–424, 2021.

[2] Susan Stepney, Fiona A.C. Polack, and Heather R. Turner. Engineering Emergence. In *Proceedings of the 11th IEEE International Conference on Engineering of Complex Computer Systems*, Stanford, CA, USA, 2006. IEEE Computer Society.

[3] Rick L. Sturdivant and Edwin K.P. Chong. The Necessary and Sufficient Conditions for Emergence in Systems Applied to Symbol Emergence in Robots. *IEEE Transactions on Cognitive and Developmental Systems*, 10(4):1035–1042, 12 2018.

[4] W Ross Ashby. Principles of the self-organizing system. *Emergence: Complexity and Organization*, 6(2):102–126, 2004.

[5] Hassan K Khalil. *Nonlinear systems*. Prentice-Hall, Inc., Upper Saddle River, NJ, second edition edition, 1996.

[6] David V Newman. Emergence and Strange Attractors. *Philosophy of Science*, 63(2):245–261, 1996.

[7] Takamitsu Watanabe, Satoshi Hirose, Hiroyuki Wada, Yoshio Imai, Toru Machida, Ichiro Shirouzu, Seiki Konishi, Yasushi Miyashita, and Naoki Masuda. Energy landscapes of resting-state brain networks. *Frontiers in Neuroinformatics*, 8(FEB), 2 2014.

[8] Reza Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420, 3 2006.

[9] Logan E. Beaver and Andreas A Malikopoulos. An Overview on Optimal Flocking. *Annual Reviews in Control*, 51:88–99, 2021.

[10] Scott E. Page. Understanding Complexity, 2009.

[11] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. *Computer Graphics*, 21(4):25–34, 1987.

[12] Hyondong Oh, Ataollah Ramezan Shirazi, Chaoli Sun, and Yaochu Jin. Bio-inspired self-organising multi-robot pattern formation: A review. *Robotics and Autonomous Systems*, 91:83–100, 2017.

[13] Iztok Lebar Bajec and Frank H. Heppner. Organized flight in birds. *Animal Behaviour*, 78(4):777–789, 10 2009.

[14] Herbert G. Tanner, Ali Jadbabaie, and George J. Pappas. Flocking in fixed and switching networks. *IEEE Transactions on Automatic Control*, 52(5):863–868, 2007.

[15] Yoram Koren and Johann Borenstein. Potential Field Methods and their Inherent Limitations for Mobile Robot Navigation. In *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, 1991.

[16] Gábor Vásárhelyi, Csaba Virágh, Gergő Somorjai, Tamás Nepusz, Agoston E Eiben, and Tamás Vicsek. Optimized flocking of autonomous drones in confined environments. *Science Robotics*, 3(20), 2018.

[17] Koichiro Morihiro, Teijiro Isokawa, Haruhiko Nishimura, and Nobuyuki Matsui. Emergence of Flocking Behavior Based on Reinforcement Learning. In *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, pages 699–706, 2006.

[18] Chao Wang, Jian Wang, and Zhang Xudong. A Deep Reinforcement Learning Approach to Flocking and Navigation of UAVs in Large-Scale Complex Environments. In *2018 IEEE Global Conference on Signal and Information Processing*, 2018.

[19] Haibin Duan and Peixin Qiao. Pigeon-inspired optimization: A new swarm intelligence optimizer for air robot path planning. *International Journal of Intelligent Computing and Cybernetics*, 7(1):24–37, 2014.

[20] James Kennedy and Russell Eberhart. Particle Swarm Optimization. In *International Conference on Neural Networks*, pages 1942–1948, 1995.

[21] Hande Celikkanat. Optimization of self-organized flocking of a robot swarm via evolutionary strategies. In *23rd International Symposium on Computer and Information Sciecnes*, 2008.

[22] Huaxin Qiu and Haibin Duan. A multi-objective pigeon-inspired optimization approach to UAV distributed flocking among obstacles. *Information Sciences*, 509:515–529, 1 2020.

[23] Hai Tao Zhang, Michael Zhiqiang Chen, Guy Bart Stan, Tao Zhou, and Jan M. MacIejowski. Collective behavior coordination with predictive mechanisms. *IEEE Circuits and Systems Magazine*, 8(3):67–85, 9 2008.

[24] Jingyuan Zhan and Xiang Li. Flocking of Discrete-time Multi-Agent Systems with Predictive Mechanisms. In *18th IFAC World Congress*, pages 5669–5674, 2011.

[25] Jingyuan Zhan and Xiang Li. Decentralized Flocking Protocol of Multi-agent Systems with Predictive Mechanisms. In *Proceedings of the 30th Chinese Control Conference*, pages 5995–6000, 2011.

[26] Quan Yuan, Jingyuan Zhan, and Xiang Li. Outdoor flocking of quadcopter drones with decentralized model predictive control. *ISA Transactions*, 71:84–92, 11 2017.

[27] Benedetto Piccoli, Nastassia Pouradier Duteil, and Benjamin Scharf. Optimal Control of a Collective Migration Model. *Mathematical Models and Methods in Applied Sciences*, 26(2), 2016.

[28] Steven A.P. Quintero, Gaemus E. Collins, and Joao P. Hespanha. Flocking with fixed-wing UAVs for distributed sensing: A stochastic optimal control approach. In *Proceedings of the American Control Conference*, pages 2025–2031, 2013.

[29] Tatsuya Ibuki, Sean Wilson, Junya Yamauchi, Masayuki Fujita, and Magnus Egerstedt. Optimization-Based Distributed Flocking Control for Multiple Rigid Bodies. *IEEE Robotics and Automation Letters*, 5(2):1891–1898, 4 2020.

[30] Andre Nathan and Valmir C Barbosa. V-like Formations in Flocks of Artificial Birds. *Artificial Life*, 14(2):197–188, 2008.

[31] R A R Bedruz, Jose Martin Z Maningo, Arvin H Fernando, Argel A Bandala, Ryan Rhay P Vicerra, and Elmer P Dadios. Dynamic Peloton Formation Configuration Algorithm of Swarm Robots for Aerodynamic Effects Optimization. In *Proceedings of the 7th International Conference on Robot Intelligence Technology and Applications*, pages 264–267, 2019.

[32] Junxing Yang, Radu Grosu, Scott A. Smolka, and Ashish Tiwari. Love thy neighbor: V-formation as a problem of model predictive control. In *Leibniz International Proceedings in Informatics, LIPIcs*, volume 59, 2016.

[33] A. Mirzaeinia, M. Hassanalian, K. Lee, and M. Mirzaeinia. Energy conservation of V-shaped swarming fixed-wing drones through position reconfiguration. *Aerospace Science and Technology*, 94, 11 2019.

[34] Gabriele Ribichini and Emilio Frazzoli. Efficient Coordination of Multiple Aircraft Systems. In *4nd IEEE Conference on Decision and Control*, pages 1035–1040, 2003.

[35] Thomas Eliot Kent. *Optimal Routing and Assignment for Commercial Formation Flight*. PhD thesis, University of Bristol, 2015.

[36] C M A Verhagen. *Formation flight in civil aviation Development of a decentralized approach to formation flight routing*. PhD thesis, Delft University of Technology, 2015.

[37] Zhuoyuan Song, Doug Lipinski, and Kamran Mohseni. Multi-vehicle cooperation and nearly fuel-optimal flock guidance in strong background flows. *Ocean Engineering*, 141:388–404, 2017.

[38] Logan E. Beaver, Behdad Chalaki, A. M. Ishtiaque Mahbub, Liuhui Zhao, Ray Zayas, and Andreas A. Malikopoulos. Demonstration of a Time-Efficient Mobility System Using a Scaled Smart City. *Vehicle System Dynamics*, 58(5):787–804, 2020.

[39] Behdad Chalaki, Logan E. Beaver, A M Ishtiaque Mahbub, Heeseung Bang, and Andreas A. Malikopoulos. A research and educational robotic testbed for real-time control of emerging mobility systems: From theory to scaled experiments. *IEEE Control Systems Magazine*, 2022 (in press).

[40] Kwang-Kyo Oh, Myoung-Chul Park, and Hyo-Sung Ahn. A survey of multi-agent formation control. *Automatica*, 53:424–440, 2015.

[41] Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. Swarm robotics: A review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, 2013.

[42] Logan E. Beaver and Andreas A Malikopoulos. A Decentralized Control Framework for Energy-Optimal Goal Assignment and Trajectory Generation. In *IEEE 58th Conference on Decision and Control*, pages 879–884, 2019.

[43] Logan E. Beaver and Andreas A Malikopoulos. An Energy-Optimal Framework for Assignment and Trajectory Generation in Teams of Autonomous Agents. *Systems & Control Letters*, 138, April 2020.

[44] Heeseung Bang, Logan E Beaver, and Andreas A Malikopoulos. Energy-optimal goal assignment of multi-agent system with goal trajectories in polynomials. In *2021 29th Mediterranean Conference on Control and Automation (MED)*, pages 1228–1233, 2021.

[45] Magnus Egerstedt, Jonathan N. Pauli, Gennaro Notomista, and Seth Hutchinson. Robot ecology: Constraint-based control design for long duration autonomy. *Annual Reviews in Control*, 46:1–7, 1 2018.

[46] Logan E. Beaver and Andreas A. Malikopoulos. Constraint-driven optimal control of multi-agent systems: A highway platooning case study. *IEEE Control Systems Letters*, 6:1754–1759, 2022.

[47] Logan E. Beaver and Andreas A. Malikopoulos. Constraint-driven optimal control for emergent swarming and predator avoidance. *arXiv:2203.11057 (in review)*, 2022.

[48] Logan E Beaver, Christopher Kroninger, and Andreas A Malikopoulos. A Constraint-Driven Approach to Line Flocking: The V Formation as an Energy-Saving Strategy. In *(in preparation)*, 2022.

[49] Arthur E. Bryson, Jr. Optimal Control-1950 to 1985. *IEEE Control Systems Magazine*, 16(3):26–33, 1996.

[50] Michel Fliess, Jean Levine, Philippe Martin, and Pierre Rouchon. Flatness and defect of non-linear systems: Introductory theory and examples. *International Journal of Control*, 61(6):1327–1361, 1995.

[51] Oladapo Tolulola Ogunbodede. *Optimal Control of Differentially Flat Systems*. PhD thesis, The University at Buffalo, 2020.

[52] F Chaplais and N Petit. Inversion in indirect optimal control: constrained and unconstrained cases. In *46th IEEE Conference on Decision and Control*, pages 683–689, 2007.

[53] François Chaplais and Nicolas Petit. Inversion in indirect optimal control of multivariable systems. *ESAIM: COCV*, 14:294–317, 2008.

[54] Knut Graichen, Andreas Kugi, Nicolas Petit, and Francois Chaplais. Handling constraints in optimal control with saturation functions and system extension. *Systems and Control Letters*, 59(11):671–679, 11 2010.

[55] Pierre-Cyril Aubin-Frankowski. *Estimation and control under constraints through Kernel methods*. PhD thesis, Mines ParisTech, Paris, 7 2021.

[56] Jonas Koenemann, Giovanni Licitra, and Moritz Politecnico, Mustafa AlpDiehl. OpenOCL - Open Optimal Control Library. In *Robotics Science and Systems*, 2019.

[57] Yuanbo Nie, Omar Faqir, and Eric C Kerrigan. ICLOCS2: Try this Optimal Control Problem Solver Before you Try the Rest;. In *2018 UKACC 12th International Conference on Control (CONTROL)*, 2018.

[58] Behdad Chalaki, Logan E Beaver, and Andreas A Malikopoulos. Experimental validation of a real-time optimal controller for coordination of cavs in a multi-lane roundabout. In *31st IEEE Intelligent Vehicles Symposium (IV)*, pages 504–509, 2020.

[59] Logan E. Beaver, Michael Dorothy, Christopher Kroninger, and Andreas A Malikopoulos. Energy-Optimal Motion Planning for Agents: Barycentric Motion and Collision Avoidance Constraints. In *2021 American Control Conference*, pages 1037–1042, 2021.

[60] Logan E. Beaver and Andreas A Malikopoulos. Optimal Control of Differentially Flat Systems is Surprisingly Simple. *arXiv:2103.03339 (in review)*, 2022.

[61] Matthew Turpin, Nathan Michael, and Vijay Kumar. Concurrent Assignment and Planning of Trajectories for Large Teams of Interchangeable Robots. In *2013 IEEE International Conference Robotics and Automation (ICRA)*, pages 842–848, Karlsruhe, Germany, 2013.

[62] Hang Ma, Daniel Harabor, Peter J Stuckey, Jiaoyang Li, and Sven Koenig. Searching with Consistent Prioritization for Multi-Agent Path Finding. In *33rd AAAI Conference on Artificial Intelligence*, pages 7643–7650, 2019.

[63] Wenying Wu, Subhrajit Bhattacharya, and Amanda Prorok. Multi-Robot Path Deconfliction through Prioritization by Path Prospects. In *2020 IEEE International Conference on Robotics and Automation*, pages 9809–9815, 8 2020.

[64] Guillaume Sartoretti, Justin Kerr, Yunfei Shi, Glenn Wagner, T. K. Satish Kumar, Sven Koenig, and Howie Choset. PRIMAL: Pathfinding via Reinforcement and Imitation Multi-Agent Learning. *IEEE Robotics and Automation Letters*, 4 (3):2378–2385, 2018.

[65] H S Witsenhausen. A counterexample in stochastic optimum control. *Siam Journal of Control*, 6(1):131–147, 1968.

[66] Aditya Dave and Andreas A Malikopoulos. Structural results for decentralized stochastic control with a word-of-mouth communication. In *2020 American Control Conference (ACC)*, pages 2796–2801, 2020.

[67] Aditya Dave and Andreas A Malikopoulos. A dynamic program for a team of two agents with nested information. In *2021 IEEE Conference on Decision and Control (CDC)*, pages 3768–3773. IEEE, 2021.

[68] Daniel Morgan, Giri P. Subramanian, Soon-Jo Chung, and Fred Y. Hadaegh. Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming. *International Journal of Robotics Research*, 35(10):1261–1285, 2016.

[69] Matthew Turpin, Nathan Michael, and Vijay Kumar. CAPT: Concurrent assignment and planning of trajectories for multiple robots. *International Journal of Robotics Research*, 33(1):98–112, 2014.

[70] Matthew Turpin, Kartik Mohta, Nathan Michael, and Vijay Kumar. Goal Assignment and Trajectory Planning for Large Teams of Aerial Robots. *Proceedings of Robotics: Science and Systems*, 37:401–415, 2013.

[71] Stephen P. Boyd and Lieven. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[72] Gennaro Notomista, Siddharth Mayya, Seth Hutchinson, and Magnus Egerstedt. An optimal task allocation strategy for heterogeneous multi-robot systems. In *18th European Control Conference*, pages 2071–2076, 6 2019.

[73] Wei Xiao, Calin A. Belta, and Christos G. Cassandras. Sufficient conditions for feasibility of optimal control problems using Control Barrier Functions. *Automatica*, 135, 1 2022.

[74] Lars Lindemann and Dimos V. Dimarogonas. Control barrier functions for signal temporal logic tasks. *IEEE Control Systems Letters*, 3(1):96–101, 1 2019. ISSN 24751456. doi: 10.1109/LCSYS.2018.2853182.

[75] Li Wang, Aaron D. Ames, and Magnus Egerstedt. Safety barrier certificates for collisions-free multirobot systems. *IEEE Transactions on Robotics*, 33(3): 661–674, 2017.

[76] A M Ishtiaque Mahbub and Andreas A Malikopoulos. A Platoon Formation Framework in a Mixed Traffic Environment. *IEEE Control Systems Letters (LCSS)*, 6:1370–1375, 2021.

[77] S.D. Kumaravel, Andreas A Malikopoulos, and R. Ayyagari. Optimal coordination of platoons of connected and automated vehicles at signal-free intersections. *IEEE Transactions on Intelligent Vehicles*, pages 1–1, 2021. doi: 10.1109/TIV.2021.3096993.

[78] A M Ishtiaque Mahbub and Andreas A. Malikopoulos. Platoon Formation in a Mixed Traffic Environment: A Model-Agnostic Optimal Control Approach. *Proceedings of 2022 American Control Conference*, October 2022 (to appear).

[79] Danielle Fredette. *Fuel-Saving behavior for Multi-Vehicle Systems: Analysis, Modeling, and Control*. PhD thesis, The Ohio State University, 2017.

[80] Yang Zhu and Feng Zhu. Barrier-function-based distributed adaptive control of nonlinear CAVs with parametric uncertainty and full-state constraint. *Transportation Research Part C: Emerging Technologies*, 104:249–264, 7 2019.

[81] Julien M. Hendrickx, Balázs Gerencsér, and Baris Fidan. Trajectory Convergence From Coordinate-Wise Decrease of Quadratic Energy Functions, and Applications to Platoons. *IEEE Control Systems Letters*, 4(1):151–156, 1 2020.

[82] Andrea Tagliabue, Aleix Paris, Suhan Kim, Regan Kubicek, Sarah Bergbreiter, and Jonathan P. How. Touch the wind: Simultaneous airflow, drag and interaction sensing on a multirotor. In *IEEE International Conference on Intelligent Robots and Systems*, pages 1645–1652, 2020.

[83] Yousef Emam, Paul Glotfelter, and Magnus Egerstedt. Robust Barrier Functions for a Fully Autonomous, Remotely Accessible Swarm-Robotics Testbed. In *58th Conference on Decision and Control*, pages 3984–3990, 2019.

[84] Gennaro Notomista and Magnus Egerstedt. Constraint-Driven Coordinated Control of Multi-Robot Systems. In *Proceedings of the 2019 American Control Conference*, 2019.

[85] Johan Löfberg. Oops! I cannot do it again: Testing for recursive feasibility in MPC. *Automatica*, 48:550–555, 2012.

[86] Mingming Shi and Julien M. Hendrickx. Are energy savings the only reason for the emergence of bird echelon formation? *ArXiv: 2103.13381*, 3 2021.

[87] A. Mirzaeinia, F. Heppner, and M. Hassanalian. An analytical study on leader and follower switching in V-shaped Canada Goose flocks for energy management purposes. *Swarm Intelligence*, 14(2):117–141, 6 2020.

[88] Shouvik Roy, Usama Mehmood, Radu Grosu, Scott A. Smolka, Scott D. Stoller, and Ashish Tiwari. Learning distributed controllers for V-formation. In *2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems*, pages 119–128. Institute of Electrical and Electronics Engineers Inc., 8 2020.

[89] A. E. Bryson, Jr. and Yu-Chi Ho. *Applied Optimal Control: Optimization, Estimation, and Control*. John Wiley and Sons, 1975.

[90] I. Michael Ross. *A Primer on Pontryagin's Principle in Optimal Control*. Collegiate Publishers, San Francisco, 2nd edition, 2015.

[91] Jean Lévine. On necessary and sufficient conditions for differential flatness. *Applicable Algebra in Engineering, Communications and Computing*, 22(1):47–90, 2011.

[92] Behdad Chalaki and Andreas A Malikopoulos. Robust learning-based trajectory planning for emerging mobility systems. In *2022 American Control Conference (ACC)*, 2022 (accepted) arXiv:2103.03313.

[93] John Anderson. *Fundamentals of Aerodynamics*. McGraw-Hill, New York, NY, sixth edition edition, 2017.

[94] Magnus Egerstedt. *Robot Ecology: Constraint-Based Design for Long-Duration Autonomy*. Princeton University Press, 2021.

[95] Jianan Wang and Ming Xin. Distributed optimal cooperative tracking control of multiple autonomous robots. *Robotics and Autonomous Systems*, 60(4):572–583, 4 2012.

[96] María Santos, Siddharth Mayya, Gennaro Notomista, and Magnus Egerstedt. Decentralized Minimum-Energy Coverage Control for Time-Varying Density Functions. In *2019 International Symposium on Multi-Robot and Multi-Agent Systems*, pages 155–161, 2019.

[97] Tyler H. Summers and John Lygeros. Distributed model predictive consensus via the Alternating Direction Method of Multipliers. In *50th Annual Allerton Conference on Communication, Control, and Computing*, pages 79–84, 2012.

[98] Yang Lyu, Jinwen Hu, Ben M. Chen, Chunhui Zhao, and Quan Pan. Multivehicle Flocking With Collision Avoidance via Distributed Model Predictive Control. *IEEE Transactions on Cybernetics*, pages 1–12, 10 2019.

[99] Jingyuan Zhan and Xiang Li. Flocking of multi-agent systems via model predictive control based on position-only measurements. *IEEE Transactions on Industrial Informatics*, 9(1):377–385, 2013.

[100] AeroVironment. Raven RQ-11B, 2021. URL https://www.avinc.com/images/uploads/product_docs/Raven_Datasheet_07122021.pdf.

[101] Hung Manh La, Ronny Lim, and Weihua Sheng. Multirobot cooperative learning for predator avoidance. *IEEE Transactions on Control Systems Technology*, 23 (1):52–63, 1 2015.

[102] Carsten Hahn, Thomy Phan, Thomas Gabor, Lenz Belzner, and Claudia Linnhoff-Popien. Emergent Escape-based Flocking Behavior using Multi-Agent Reinforcement Learning. In *Artificial Life Conference*, pages 598–605, 2019.

[103] Tony J. Pitcher and Christopher J. Wyche. Predator-avoidance behaviours of sand-eel schools: why schools seldom split. In *Predators and Prey in Fishes*, pages 193–204, 1983.

[104] Logan E. Beaver and Andreas A Malikopoulos. Beyond Reynolds: A Constraint-Driven Approach to Cluster Flocking. In *IEEE 59th Conference on Decision and Control*, pages 208–213, 2020.

[105] Logan E. Beaver, Chris Kroninger, and Andreas A Malikopoulos. An Optimal Control Approach to Flocking. In *2020 American Control Conference*, pages 683–688, 2020.

[106] Benjamin T. Fine and Dylan A. Shell. Unifying microscopic flocking motion models for virtual, robotic, and biological flock members. *Autonomous Robots*, 35(2-3):195–219, 10 2013.

[107] Emiliano Cristiani, Paolo Frasca, and Benedetto Piccoli. Effects of anisotropic interactions on the structure of animal groups. *Journal of Mathematical Biology*, 62(4):569–588, 4 2011.

[108] M Ballerini, N Cabibbo, R Candelier, A Cavagna, E Cisbani, I Giardina, V Lecomte, A Orlandi, G Parisi, A Procaccini, M Viale, and V Zdravkovic. Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study. *Proceedings of the National Academy of Sciences of the United States of America*, 105(4):1232–1237, 2008.

[109] Lifeng Zhou and Shaoyuan Li. Distributed model predictive control for multi-agent flocking via neighbor screening optimization. *International Journal of Robust and Nonlinear Control*, 27(9):1690–1705, 6 2017.

[110] Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European Control Conference, ECC 2019*, pages 3420–3431. Institute of Electrical and Electronics Engineers Inc., 6 2019.

[111] Carsten Hahn, Fabian Ritz, Paula Wikidal, Thomy Phan, Thomas Gabor, and Claudia Linnhoff-Popien. Foraging Swarms using Multi-Agent Reinforcement Learning. In *Artificial Life Conference*, pages 333–340, 2020.

[112] Florian Berlinger, Paula Wulkop, and Radhika Nagpal. Self-Organized Evasive Fountain Maneuvers with a Bioinspired Underwater Robot Collective. In *2021 IEEE International Conference on Robotics and Automation*, pages 9204–9211. Institute of Electrical and Electronics Engineers (IEEE), 10 2021.

[113] Richard M Murray, Muruhan Rathinam, and Willem Sluis. Differential Flatness of Mechanical Control Systems: A Catalog of Prototype Systems. In *ASME International Mechanical Engineering Congress and Expo*, 1995.

[114] Koushil Sreenath, Nathan Michael, and Vijay Kumar. Trajectory generation and control of a quadrotor with a cable-suspended load - A differentially-flat hybrid system. In *IEEE International Conference on Robotics and Automation*, pages 4888–4895, 2013.

[115] Mark B Milam. *Real-Time Optimal Trajectory Generation for Constrained Dynamical Systems*. PhD thesis, California Institute of Technology, 2003.

[116] Andreas A Malikopoulos. On team decision problems with nonclassical information structures. *IEEE Transactions on Automatic Control*, 2022 (conditionally accepted) arXiv:2101.10992.

[117] Behdad Chalaki and Andreas A Malikopoulos. An optimal coordination framework for connected and automated vehicles in two interconnected intersections. In *2019 IEEE Conference on Control Technology and Applications (CCTA)*, pages 888–893, 2019.

[118] Michael Rubenstein, Christian Ahler, and Radhika Nagpal. Kilobot: A low cost scalable robot system for collective behaviors. In *Proceedings of the 2012 IEEE International Conference on Robotics and Automation*, pages 3293–3298, 2012.

[119] Kathy Jang, Eugene Vinitsky, Behdad Chalaki, Ben Remer, Logan Beaver, Andreas A Malikopoulos, and Alexandre Bayen. Simulation to scaled city: zero-shot policy transfer for traffic control via autonomous vehicles. In *Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems*, pages 291–300, 2019.

[120] A M Ishtiaque Mahbub and Andreas A Malikopoulos. Concurrent optimization of vehicle dynamics and powertrain operation using connectivity and automation. In *SAE Technical Paper 2020-01-0580*. SAE International, 2020.

[121] Andreas A Malikopoulos, Logan E Beaver, and Ioannis Vasileios Chremos. Optimal time trajectory and coordination for connected and automated vehicles. *Automatica*, 125(109469), 2021.

[122] Nicolas Petit, Mark B Milam, and Richard M Murray. Inversion Based Constrained Trajectory Optimization. *IFAC Proceedings Volumes*, 34(6):1211–1216, 2001.

[123] Jean Lévine. On The Equivalence Between Differential Flatness and Dynamic Feedback Linearizability. *IFAC Proceedings Volumes*, 40(20):338–343, 2007.

[124] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *IEEE International Conference on Robotics and Automation*, pages 2520–2525, 2011.

[125] Bernd Kolar, Hubert Rams, and Kurt Schlacher. Time-optimal flatness based control of a gantry crane. *Control Engineering Practice*, 60:18–27, 3 2017.

[126] Herbertt Sira-Ramirez and Sunil K. Agrawal. *Differentially Flat Systems*. CRC Press, 1st edition, 2018.

[127] M. Van Nieuwstadt, M. Rathinam, and R. M. Murray. Differential flatness and absolute equivalence. In *Proceedings of the IEEE Conference on Decision and Control*, volume 1, pages 326–332, 1994.

[128] Behdad Chalaki and Andreas A. Malikopoulos. An Optimal Coordination Framework for Connected and Automated Vehicles in two Interconnected Intersections. In *2019 IEEE Conference on Control Technology and Applications*, Hong Kong, 2 2019. URL http://arxiv.org/abs/1903.00120.

[129] Behdad Chalaki and Andreas A. Malikopoulos. A priority-aware replanning and resequencing framework for coordination of connected and automated vehicles. *IEEE Control Systems Letters*, 6:1772–1777, 2022. doi: 10.1109/LCSYS.2021.3133416.

[130] Melissa Greeff and Angela P Schoellig. Exploiting Differential Flatness for Robust Learning-Based Tracking Control using Gaussian Processes. *IEEE Control System Letters*, 5(4):1121–1126, 2021.

[131] Inmo Jang, Hyo-Sang Shin, and Antonios Tsourdos. Anonymous Hedonic Game for Task Allocation in a Large-Scale Multiple Agent System. *IEEE Transactions on Robotics*, 34(6):1534–1548, 2018.

[132] Behdad Chalaki, Logan E Beaver, Ben Remer, Kathy Jang, Eugene Vinitsky, Alexandre Bayen, and Andreas A Malikopoulos. Zero-shot autonomous vehicle policy transfer: From simulation to real-world via adversarial learning. In *IEEE 16th International Conference on Control & Automation (ICCA)*, pages 35–40, 2020.

Appendix A

REPUBLICATION PERMISSIONS

This appendix lists the articles that appear, partially or in full, in this dissertation. The articles are presented based on the publisher of the original article, and a RightsLink page for each publisher follows.

Articles Previously Published by Elsevier:

- [9] L.E. Beaver and A.A. Malikopoulos, "An Overview on Optimal Flocking," in *Annual Reviews in Control*, vol. 51, pp. 88–99.
- [43] L.E. Beaver and A.A. Malikopoulos, "An Energy-Optimal Framework for Assignment and Trajectory Generation in Teams of Autonomous Agents" in *Systems & Control Letters*, vol. 138, April 2020.

Articles Previously Published by IEEE

- [46] L.E. Beaver and A.A. Malikopoulos, "Constraint-Driven Optimal Control of Multiagent Systems: A Highway Platooning Case Study" in *IEEE Control Systems Letters*, vol. 6, pp. 1754–1759, 2022.
- [58] B. Chalaki, L.E. Beaver and A.A. Malikopoulos, "Experimental Validation of a Real-Time Optimal Controller for Coordination of CAVs in a Multi-Lane Roundabout" 2020 Intelligent Vehicles Symposium, pp. 775–780, 2020.

An overview on optimal flocking

Author: Logan E. Beaver,Andreas A. Malikopoulos
Publication: Annual Reviews in Control
Publisher: Elsevier
Date: 2021

Journal Author Rights

Please note that, as the author of this Elsevier article, you retain the right to include it in a thesis or dissertation, provided it is not published commercially. Permission is not required, but please ensure that you reference the journal as the original source. For more information on this and on your other retained rights, please visit: https://www.elsevier.com/about/our-business/policies/copyright#Author-rights

BACK CLOSE WINDOW