A Formal Investigation of the Organization of Guidance Behavior: Implications for Humans and Autonomous Guidance

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Zhaodan Kong

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Advisor: Bérénice Mettler

October, 2012

Acknowledgements

This dissertation would not have been completed without the help of a multitude of people to whom I am indebted.

First and foremost, I would like to express my gratitude to my mentor and advisor, Prof. Bérénice Mettler. She opened to me a world that I never imagined I would and could know. She guided me to pursue ideas that were inspiring and challenging. Some of these ideas were completely novel and we had to always prepare ourselves to answer the "Why" and "How" questions. If it was not for her constant support and encouragement during all these years, I could not have gone this far. I would also like to thank her for sharing her wisdom about both academic life and beyond. She was alway patient to teach me about research approaches, presentation techniques and written skills. It was her effort in editing the drafts of my papers that made them presentable. I truly enjoyed working with her and I hope our collaboration can last a lifetime.

I would like to express my appreciation to my other dissertation committee members, Prof. Demoz Gebre-Egziabher, Prof. Peter Seiler and Prof. Victoria Interrante, for their availability, for their times spent on reading my dissertation and for the valuable advice they provided in their areas of expertise.

I am grateful to a number of other faculty members. Together with Prof. Mettler, Prof. Paul Shrater organized a discussion group on understanding cognitive science from a dynamical system perspective. Some of the ideas presented in this dissertation were inspired by the discussions there. I would also like to thank Prof. Yiyuan Zhao and Prof. Perry Leo for writing a number of reference letters for me during the past few years.

I would like to thank my colleagues and friends, Navid Dadkhah, Jon Andersh and Venkat Korukanti, at the Interactive Guidance and Control Lab and my collaborator,

Chad Goerzen, at the NASA Ames Research Center. They are all first-rate researchers and I learned a lot from them. By working with Venkat, I learned how to write efficient and user-friendly code. Chad introduced me to a research philosophy that was beyond university laboratories. Navid and Jon were more than helpful in providing their support for setting up the hardwares and computers for the guidance experiments. I could not hope for a better set of colleagues and collaborators.

I would like to thank the two pilots, Alex Burt and Mark Hammers, for their participation in the guidance experiments. Some of the experiments were conducted during weekends or summer vocation. I am really grateful for their willingness to sacrifice their time to take part in the experiments.

I am grateful to Aunt Fei Liu, Uncle Xiaogang Luo, Yiming Wu and Zhiyong Zhang for getting me familiarized with the campus, the cities and the American lifestyle. I can not imagine how the first few months would be without their help. I would also like to thank all my other friends who made all these years memorable, including: Xiaochuan Chai, Guowei Yu, Ming Zeng, Xiaolei Shi, Xingyong Song, Zheqiang Xing, Guijing Zheng, Qi Gao, Zhefeng Li, Heming Chen, Di Wu, Chonglin Zhang, Yintao Song, Xian Chen and Wei Guo.

Special thanks to the administrative staff of the Department of Aerospace Engineering and Mechanics, the Graduate School, the International Student and Scholar Services and the Center for Writing. They are always supportive and helpful.

Last but not least, I would like to thank my father and my mother to whom I am eternally grateful. They have always stood by my side. Their encouragement and their unconditional love were the reasons that I was always able to strive for better. They taught me the values of hard work, of integrity, of being humble, and of believing the power of knowledge. I owe everything that I have achieved to them.

Dedication

To my father and my mother with gratitude, appreciation and love.

Abstract

Guidance behavior generated either by artificial agents or humans has been actively studied in the fields of both robotics and cognitive science. The goals of these two fields are different. The former is the automatic generation of appropriate or even optimal behavior, while the latter is the understanding of the underlying mechanism. Their challenges, though, are closely related, the most important one being the lack of a unified, formal and grounded framework where the guidance behavior can be modeled and studied.

This dissertation presents such a framework. In this framework, guidance behavior is analyzed as the closed-loop dynamics of the whole agent-environment system. The resulting dynamics give rise to interaction patterns. The central points of this dissertation are that: first of all, these patterns, which can be explained in terms of symmetries that are inherent to the guidance behavior, provide building blocks for the organization of behavior; second, the existence of these patterns and humans' organization of their guidance behavior based on these patterns are the reasons that humans can generate successful behavior in spite of all the complexities involved in the planning and control.

This dissertation first gives an overview of the challenges existing in both scientific endeavors, such as human and animal spatial behavior study, and engineering endeavors, such as autonomous guidance system design. It then lays out the foundation for our formal framework, which states that guidance behavior should be interpreted as the collection of the closed-loop dynamics resulting from the agent's interaction with the environment. The following, illustrated by examples of three different UAVs, shows that the study of the closed-loop dynamics should not be done without the consideration of vehicle dynamics, as is the common practice in some of the studies in both autonomous guidance and human behavior analysis. The framework, the core concepts of which are symmetries and interaction patterns, is then elaborated on with the example of Dubins' vehicle's guidance behavior. The dissertation then describes the details of the agile human guidance experiments using miniature helicopters, the technique that is developed for the analysis of the experimental data and the analysis results. The results

confirm that human guidance behavior indeed exhibits invariance as defined by interaction patterns. Subsequently, the behavior in each interaction pattern is investigated using piecewise affine model identification. Combined, the results provide a natural and formal decomposition of the behavior that can be unified under a hierarchical hidden Markov model.

By employing the languages of dynamical system and control and by adopting algorithms from system identification and machine learning, the framework presented in this dissertation provides a fertile ground where these different disciplines can meet. It also promises multiple potential directions where future research can be headed.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

We start this chapter with an introduction of two fields that are going to be addressed in this dissertation. They are human spatial behavior analysis (mainly in the context of guidance), and autonomous guidance and control. One common quest for these two fields is the establishment of a framework that can potentially explain and deal with the mitigation of complexities existing in guidance. In Section 1.2 and Section 1.3, we give an overview of some existing works that are related to the solution of this quest. The overview is done in two parts: first, the scientific background, and then the engineering background. Following, in Section 1.4, we first illustrate what is currently missing from the existing works. We then provide the main thesis of this dissertation, which is the building of a unified, formal and grounded framework to model and analyze guidance behavior. We conclude this chapter with an outline of the dissertation and the first published appearance of each chapter.

1.1 Motivation

1.1.1 Human Spatial Behavior

The analysis of humans' mechanism to generate and organize their spatial behavior has been an active research topic in psychology and robotics over the past few decades [11]. What fascinates researchers is trained humans' ability to spontaneously generate behavior for problems that are often not tractable from a computational standpoint. Take

(a) (b)

Figure 1.1: Flying a helicopter in (a) an urban environment or (b) a mountainous environment (images from flickr.com).

flying a helicopter in an urban or a mountainous environment (as shown in Fig. 1.1) for instance; the task involves a pilot, a helicopter (the pilot and the helicopter together can be taken as an agent[1]), as well as the surrounding environment. All three have their own dynamics. From the pilot's perspective, for the sake of completing this task, he or she needs not only to comprehend the dynamics of each single component (himself/herself, the helicopter and the environment), but also have a holistic understanding of the dynamics of the entire agent-environment system.

First of all, the pilot needs to comprehend each component separately[2] . This requires the comprehension of the pilot's own skeletomuscular system, the comprehension of the helicopter system, which itself is already a demanding task even in today's supervisory control settings [13, 14], and the comprehension of the environment [15, 16]. Each of these is not trivial and rigorous research fields have been established for them.

In addition to the comprehension of these separate components, the pilot also needs to understand how these components play out in the current situation and in the future,

[1] Here we are going to take the artificial intelligence definition of *agent*, i.e., "an agent is anything that can be viewed as perceiving its environment through sensors and acting upon the environment through actuators" [12]. In this dissertation, the agent could be various UAVs, humans or even a system that includes both humans and the vehicles that they are operating.

[2] The term 'comprehend' or 'comprehension' here implies our inclining towards the key role of representations in human behavior generation. As will be elaborated on later, when it comes to the modeling of human behavior, depending on whether internal models are involved, there are two schools of thoughts: the model-based school and the non-representational school. We are more inclined to the first school. But certain features of the second school are also absorbed in our modeling philosophy.

when they are integrated together for the fulfillment of the overall goals (e.g., flying to a place safely and as fast as possible while obeying air traffic rules). [17]. In paper [17], the understanding of the components as a whole with respect to the current situation and the projection of them into future conditions are called Level 2 and 3 situation awarenesses, respectively(Level 1 refers to the understanding of single components). As the complexity of the environment increases, the complexity of the interactions between the agent and the environment increases, too. As a result, various factors become coupled and it becomes more and more challenging for the pilot to perform the task. For example, when flying in an urban environment, the pilot must make sure that the helicopter can safely make the turn that is just ahead of him/her; sometimes the pilot may even be forced (by the topology of the environment) to consider the subsequent turn, which might still be invisible at the current moment. (This may happen when the pilot needs to perform a "S" shaped maneuver.)

Even facing all these coupled complexities, an experienced pilot is still able to perform the guidance tasks as shown in Fig. 1.1. Then one question that is worth answering is how humans organize their spatial behavior to mitigate these complexities. For the sake of answering this question, it would be beneficial if we can build a framework where all the relevant elements, such as the pilot's perception, planning and control processes, the dynamics of the helicopter and features in the environment, can be put together and tested with evidence obtained from experiments. However, up until now, as will be shown in Section 1.2, most research in the study of spatial behavior is done with respect to single components (for instance, motor control is the study of skeletomuscular system and spatial cognition is the study of environment representation), there is rarely a framework where all the components are studied as a whole. To make things more difficult, methods of different or even contradicting natures are used in these research [18, 19]. Planning is typically studied as a deterministic process. For example, in [20], human decision making is studied with deterministic tools such as flowchart and logic. On the other hand, perception is typically studied as probabilistic process. For example, in [21], object perception is studied with Bayesian inference. To build a unified framework, we face the challenge of bringing all these methods together.

1.1.2 Autonomous Guidance and Control

In recent years, unmanned systems have been deployed in a wide range of military applications. These systems excel in dangerous missions such as reconnaissance behind enemy lines. However, they still lack the ability to perform tasks that require agile and adaptive behaviors, such as those that need to be performed in confined environments and/or under competitive conditions. For instance, urban combat terrain is hostile to direct visual engagement, which makes it a perfect theater for the deployment of highly autonomous vehicles [22]. However, building an autonomous system that can potentially match human's performance is still out of reach at the current moment [22, 23].

A more comprehensive understanding of human's spatial behavior can potentially help engineers to design more effective autonomous systems that can generate agile and adaptive behaviors akin to humans. For instance, the lessons learned from the analysis of human guidance behavior can be integrated with learning from demonstration (LFD), a common learning technique in robotics [24, 25]. Taking flying helicopter to perform maneuvers for instance, instead of relying on handcrafted trajectory, in LFD, one can get the desired trajectory by learning from expert demonstrations [24]. The core of LFD, which is also the exact place where the analysis results of human behavior can contribute, is the knowledge representation. Appropriate knowledge representation enables transferring knowledge from teachers to students as well as between intelligent agents[3] .

The knowledge representation of LFD can be used to tackle issues, such as the *curse of dimensionality* or the *frame problem* [27, 28], in complex control problems. The end product of a control problem is a controller, which, given measurements from the sensors about the agent and the environment, assigns a control signal to the actuators. A naive controller (in the context of LFD) we can come up with is a look-up table. This look-up table records all the histories of the measurements and the demonstrated controls. In the implementation stage, if a new measurement coincides with an old

[3] The transformation of knowledge can be best understood in the context of imitation. One key issue is the *correspondence*, i.e., which action of an imitator *corresponds* to that of the imitated system [26]. Based on different correspondence criteria, in paper [26], the imitations are categorized into three types: action-level imitation, program-level imitation and effect-level imitation. Most of the imitation algorithms belong to the first two types. However, the last type, in which the effects of the actions on the environments rather than the particular physical motions are imitated, should be of the most interest to us.

measurement, the same control corresponding to the old measurement will be assigned. Though simple, such a controller would suffer from a number of problems, the most important one being generalization. Due to the high number of the data's dimension, which for a look-up table means all the histories from various sensors and controllers, the demonstrated data can only fill the state space[4] very sparsely. In such a case, interpretation or extrapolation should not be trustworthy.

To tackle the curse of dimensionality problem, LFD needs to supply *biases* [28, 30] or *patterns* [19, 31, 32][5] to the learning algorithm. The bias or pattern can be understood as "any basis for choosing one generalization over another, other than strict consistency with the observed training instances" [33]. One way of introducing bias into learning in through knowledge representation, which essentially limits the space to search for the available controllers. With the representation at hand, learning is then reduced to the selection of the right representation and the estimation of the parameters in order to fit the representation to the demonstrated data [28].

The *consistency* between the representation and the tasks, as well as the systems involved, can greatly affect the success of learning. The role of consistency can be understood in the context of imitations as elaborated on about footnote 3. Instead of being built at the action-level or at the program-level, a more appropriate representation, which can be said to be consistent, should be built in the effect-level. At this level, the *affordance* in the structural interactions between the agent and the environment needs to be analyzed [26, 34]. By studying the interactions between the agent and the environment, we can come up with representations with high fidelity, which can be generalized well and which would potentially facilitate the construction of learning algorithms for future guidance and control systems.

[4] State space includes all the situations that can arise in the world. Since we are talking about measurements and controls, a more accurate term would be *information space* [28, 29]. We keep the usage of *state space* here, since it is more intuitive and does not cause any confusion.

[5] The concept of pattern has been mainly used in the field of vision instead of guidance and control. It is relevant here in two aspects. First of all, perception, including vision, is an indispensable process of guidance. Thus, patterns can be seen as biases in the context of perception. Second, due to its abstract form, the concept of pattern can easily be extended to other processes, such as spatial representation and reasoning.

1.2 Scientific Background: Study of Human and Animal Behavior

Depending on whether internal presentations are involved, approaches of human behavior can be divided into two schools: the model-based school and the non-representational school [35]. In the model-based school, internal representations–sometimes called internal models–are constructed. The internal representations can cover the musculoskeletal system as well as the environment. The non-representational school does not rely on internal representations. Two main approaches belonging to this school are the ecological perception-action approach and the dynamical system approach.

1.2.1 Model-Based School

When it comes to understanding the role of representations in the generation of human and animal behavior, two important aspects that need to be considered are the environment and the musculoskeletal system. The environment or the space needs to be represented or modeled for the purpose of providing the basis for actions. One main model for spatial representation is the cognitive map. The musculoskeletal system needs to be represented or modeled for the purpose of generating appropriate motor actions within the environment. Two main models for motor control are closed-loop optimal control and hierarchical control.

Cognitive Map

For the study of human and animal behavior, one important quest is to understand how animals represent the environment they evolve in. The dominant view is that a neurological map is constructed by the animals. Such a map is called a *cognitive map* [36]. At first, there was no assumption that the cognitive map has biological underpinnings [36]. It was not until about 30 years after the establishment of the concept of the cognitive map that its biological underpinnings of were found in rats' brains [37]. A type of cell, which is called a place cell, was found in rats' hippocampus. A place cell fires intensively when the rat is at the cell's corresponding spatial point. And a

collection of these cells then represents the whole environment[6] . Later, head direction cells [39] and grid cells [40] were discovered. Altogether, they provide a comprehensive view of how a rat represents its spatial knowledge [41].

Paper [42] gives an explanation of how an environment-independent spatial coordinate system or cognitive map can be constructed by rats based on path integration. The main proposal is that path integration can be achieved by using velocity information and an internal timer–theta rhythm; and, when a path forms a loop (the rat returns to a point it has visited before), information along the path will be integrated into a neurological map, which encodes the spatial information of the environment.

Closed-Loop Optimal Motor Control Models

The core quest of motor control is to find the underlying movement generation principles and also the corresponding neurological mechanisms. The optimality principle is widely accepted in this field, mainly due to its consistency with the evolutionary ideas. To be more specific, at a fundamental level, the goal of motor control, or, to a larger extent, all human functionalities, is to produce responses that yield the highest possible *inclusive fitness*, which is the rate at which an animal's genes are propagated [43, 44]. If we only consider the motor system here, the animals that generate the *optimal* responses will survive and have their genes propagated. Although optimality has been used by the motor control community as the basic principle, researchers in this community are still debating what control model should be used [45]. Two popular models are closed-loop optimal control and hierarchical control.

For closed-loop optimal control models, the term *closed-loop* indicates that, while the brain is controlling the motor system, various information, from both the motor system and the sensory system, is used to correct the motor response, especially when changes happen in the environment [46]. The term *optimal* implies that the optimality principle is integrated with the closed-loop control. It is believed that, even though in principle humans obey optimality, humans do not have the computational capacity to guarantee that all the motor behaviors, e.g. muscle activation profiles and kinematic

[6] Only closed environments have been investigated for rats, at least to our knowledge. Open environments, such as the whole Pittsburgh area [38], have been used to study human spatial cognition. But no direct neurological measurements were done is the studies.

trajectories, are optimal all the time. Only the averaged behaviors are optimal with respect to some costs that haven't been determined with consensus by the researchers in the field. Some popular costs used are energy, smoothness and accuracy [45].

Though popular, closed-loop optimal control models suffer from some drawbacks. The most serious one is the difficulty involved in solving the inverse dynamic problem, i.e, the problem of transforming a desired motion to the forces that are needed to drive the limb. Even with the simplest models of muscles, the inverse dynamic problem is still a challenge (the well-known *curse of dimensionality* [47, 48]). For instance, in paper [49], 23 degrees of freedom and 54 actuators are used to model human walking. Although 23 and 54 are still extremely small numbers compared to the numbers of muscles and degrees of freedom that need to be tackled by the brain, 10000 CUP hours have to be spent to solve the problem.

Hierarchical Motor Control Models

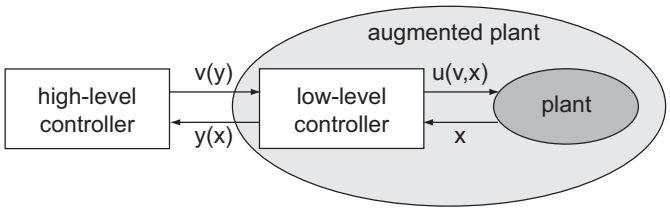


Figure 1.2: A hierarchical motor control model [1]. The higher-level controller deals with goals in the task, while the lower-level controller deals with stability or tracking issues.

One mechanism that humans can possibly use to handle the complexity issue is a hierarchical model. The idea of hierarchical motor control goes back to the works of Bernstein [47]. His basic idea is that what the brain actually controls is movements instead of individual muscles. The underlying encoding mechanism of these movements was found by Bizzi and his colleagues [50, 51]. They found that when a spot in a frog's spinal cord is stimulated, the frog's leg's movement exhibits a pattern that can be described as a dynamical system with a fixed equilibrium. Such a dynamical system is called a *convergent force field*. Subsequently, when two spots are stimulated at the same time, the force field describing the leg's movement can be represented as the composition

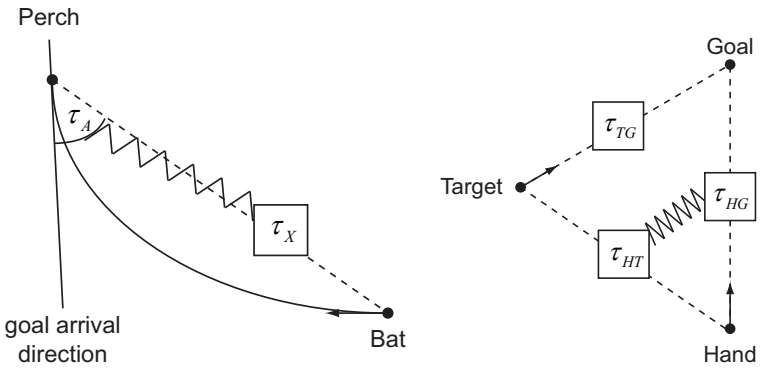
of two separate fields, each corresponding to a stimulation spot. The movement units corresponding to these fields can be called *motor primitives*. They are assumed to be innate, which means that they don't evolve over time and they are hardwired with only moderately modifiable properties [52]. They are believed to be encoded in the spinal cord.

Other findings about the mirror neuron system provide further insights about how the motor primitives are represented by the brain. Paper [53] shows that mirror neurons are active not only when a monkey executes a specific action, but also when it observes a human or other monkeys performing a similar action. Furthermore, it has been shown that the majority of mirror neurons respond selectively to one type of action. It has also been found that the mirror neuron system provides an important link in the emergence of human language [54]. All these pieces of evidence suggest that the mirror neuron system is the place where the motor primitives are represented by the brain as "action codes" in order to generate motor actions.

Based on these pieces of neurological evidence, a number of hierarchical motor control models have been proposed [55, 56, 1]. For instance, in [1], as shown in Figure 1.2, the lower-level controller together with the plant (muscles) serves the role of motor primitive. These motor primitives (augmented plants) can then be controlled by a higher-level controller. For such a framework, complexities are decoupled into different domains. The lower-level controller deals with complexities within muscle dynamics and the higher-level controller deals with complexities within task space. Such a hierarchical organization can greatly mitigate the complexity related to motor control.

1.2.2 Non-Representational School

In this subsection, we are going to review two approaches, the ecological perception-action approach and the dynamical system approach, as representatives of the non-representational school. The differences between these two approaches are rather superficial, mainly on the mathematical formulations they use. Beyond these differences, they share lots of similar ideas. For instance, both approaches believe that humans and animals are coupled with the environment and the structures existing in the coupling are the key to understanding of human and animal behavior. Actually, there has been a continual effort of trying to integrate these two approaches [57].



(a) A landing task. (b) An interception task.

Figure 1.3: Tau coupling mechanism [2]. The coupled variables are connected by zigzag lines. For instance, in Fig. 1.3(b), τ_{HT} and τ_{HG} are coupled.

The ecological perception-action approach started from Gibson's work on vision [58, 34] and was later extended to the investigation of a variety of activities, such as catching, hitting, control of braking, landing and turning, and a variety of species, such as bats, birds and humans [2, 59, 60, 61]. The main thesis of this approach is that animals and humans are physically and informationally embedded in the world and the occurrent information extracted from this embodiment is sufficient to guide actions. It is believed in this approach that the reason that information can be extracted is that, the embodiment of animals or humans within the environment is structured. The invariances related to these structured embodiment are utilized by animals or humans to guide their actions.

Lee proposed that tasks, such as landing on a perch or intercepting a goal (as shown in Fig. 1.3), can be achieved by using a variable called *tau*, which is basically the time to close a gap [2, 59, 60]. Let's take the landing task as shown in Fig. 1.3(a) as an example. In this case, there are two gaps that need to be closed: the distance between the current location and the goal, and the difference between the current heading and the goal heading. In order to reach the goal with the designated heading, both of these two gaps have to be closed. In addition, their corresponding closing rates, τ_X and τ_A, should ensure that these two gaps are closed at the same time (the coupled τs are

connected by zigzag lines in Fig. 1.3). Such a mechanism is called *tau-coupling* and it has been verified in different tasks and with different animals. Furthermore, it has been shown by neurological research that this mechanism is at least partially employed by humans during interception tasks [62].

It should be noticed that in the tau-coupling mechanism, no internal models, either of the agent or of the environment, are constructed. The absence of internal models or representations in this mechanism is exactly the essence of the non-representational approach [35]. The ecological perception-action approach connects perception to action. It analyzes the perceptual information that is needed to guide certain actions. However, it does not encompass the entire agent-environment system. Other approaches, such as the dynamical system approach which will be elaborated on later, need to be integrated with it in order to provide a systematic interpretation of human or animal behavior [35].

The Dynamical System Approach

The dynamical system approach started with Kelso's work on motor control [63]. In the study of coordinated oscillating movements of the two index fingers, he and his colleagues found that two "patterns" emerge: an in-phase motion and a counter-phase motion. As the frequency of the finger movement increases, a switch from one pattern to another occurs. This phenomenon can be interpreted as the result of a dynamical system with two attractor states. Similar coordination phenomena have been found in a number of other situations, such as between the limbs of two people [64]. They can be understood with the same dynamical system language.

Developed from these works, the main thesis of the dynamical system approach is that concepts and languages from dynamical system theory should be used to understand humans' movements as well as their interactions with the environment. One such concept is *self-organization*. In the context of human behavior, self-organization says that human behavior emerges from humans' interaction with the environment; and the behavior is not the result of some individual components but the result of their interactions under physical and informational constrains [35, 65]. Self-organization also says that behavior generation is really not a control problem but rather a coordination problem. Actions give rise to stimuli and stimuli affect actions. In a sense, humans need to grasp the interactions between perception and action, between themselves and

the environment, in order to complete their everyday tasks.

To understand human's locomotion, paper [35] and paper [66] treat human's obstacle avoidance behavior as the result of two competing terms: attraction to the goal and repulsion of the obstacles. The whole behavior can be described by a second-order nonlinear dynamical system[7] . Then the transition from one route to the other can be explained as a bifurcation in the model dynamics.

The dynamical system has also been used as a fundamental principle to understand human's cognitive processes, contrary to the traditional symbolic and connectionist approaches. The dynamical system perspective promotes an embodied and embedded view of cognition, which views that humans possess a physical body and are embedded in an environment, both physically and informationally. Please refer to papers [67, 68, 57] for details.

1.3 Engineering Background: Symbolic Planning and Control, Movement Primitive Method and Human Behavior Analysis

In this section, we are going to review three engineering research areas that we think are closely related to our work. They are more focused on addressing engineering concerns. For instance, symbolic planning and control deals with how to generate plans given some task specifications; movement primitive method deals with how to use movement primitives in robotic movement control; and human behavior analysis deals with the recognition or prediction of human behavior normally without too many concerns about the biological plausibility of the models used. These research areas are relevant in the sense that: first, they provide some hints on what mathematical languages can be used in human behavorial modeling; second, to a certain degree, humans and the autonomous systems the engineers are trying to build face problems of similar natures (e.g., the curse of dimensionality), so the insights we get from these areas can help us improve the models we build for human behavior; finally, in order to maximize the utility of

[7] The dynamical system approach always formulates behavior in the form of some low-dimensional dynamical system. In my understanding, such a practice is largely due to analytical constraints rather than physical necessities. The low-dimensional system is called coordination dynamics by Kelso [63] and behavioral dynamics by Warren [35].

human behavioral models in the design of autonomous systems, it would be best if the models were consistent in some way with the artificial systems, either conceptually or algorithmically (this last point will be further elaborated on in Section 1.4).

1.3.1 Symbolic Planning and Control

Symbolic planning and control deals with "the problem of automatic construction of robot control strategies from task specifications given in high-level, human-like language" with the help of "concepts and tools from the theory of computation such as automata and languages" [5][8] . Based on where the discretization is performed in the control hierarchy[9] , symbolic planning and control methods can be categorized into two types: environment-driven methods and control-driven methods [5].

The key of the environment-driven methods is to combine the graph representation (after environment discretization) of an environment [69] with the powerful linear temporal logic (LTL) [70, 71, 72]. With the help of LTL, it is possible to automatically generate controls for complex tasks, such as coverage, "Go to rooms P_1, P_2, P_3 in any order", and conditions, "If you see Mika, go to room P_3, otherwise stay where you are" [72]. Due to the close relationship between LTL and natural language, this method even promises a more effective human-robot interface with LTL playing the role as the interpreter from natural language to the robot. However, there are a number of issues with the environment-driven methods, the most important one being that it can not deal with dynamic a priori unknown environments[10] or systems with complex dynamics, such as the ones with nonholonomic constraints [5].

For the control-driven methods, the discretization is done at the controller level [5]. The methods are related to the development of motion descriptive language [74, 75, 76],

[8] Our paper [23] gives a detailed overview of some existing planning algorithms in the context of UAV guidance, including the symbolic planning and control methods.

[9] Please refer to paper [5] for an elaborate description of the control hierarchy in the context of symbolic planning and control. We can understand it in a very simple scenario, such as "go from A to B while avoiding obstacles". At the higher level, the environment can be first triangulated and thereby represented by a graph; then at the middle level, a sequence of control can be selected while taking into account this graph, the agent's dynamics as well as the task; finally, at the lower level, the control can be implemented as a hybrid automaton.

[10] Efforts have been made to deal with parameter uncertainty in gene network by using Computation Tree Logic (CTL) [73]. It would be interesting to see how this work can be expanded to symbolic guidance and control.

which enables the abstraction of a dynamical system via the concept of behaviors (or closed-loop control laws) and provides a formal basis for symbolic robot programming. The methods are inspired by the concept of movement primitives in motor control (as described in Section 1.2.1) and the studies on pilots' behavior while they are performing extremely challenging aerobatic maneuvers with remote control helicopters [77, 78]. For the last point, the reasoning is that it is unlikely that pilots are solving an optimization problem while performing these maneuvers, due to the myriad of complexities involved. A logical explanation would be that pilots complete these tasks by the sequencing or combining of well-rehearsed maneuvers. The work presented in this dissertation follows a similar argument. Such an argument could be traced back to Simon's study of general human strategy to perform complex tasks [79].

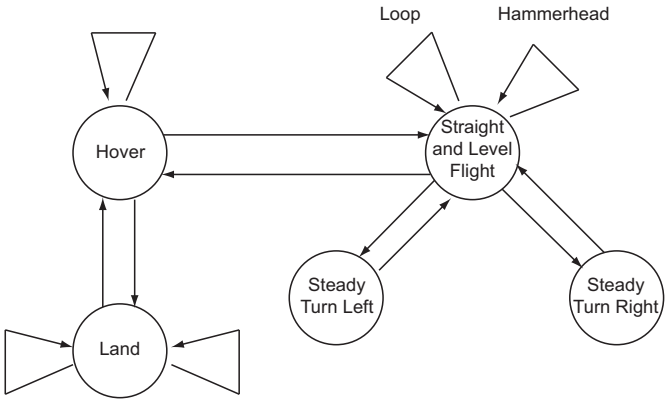


Figure 1.4: A maneuver automaton example [3, 4, 5]. Circles represent trim primitives while arrows represent maneuvers.

One control-driven method is called motion primitive method [3, 4]. For instance, as shown in Fig. 1.4, pilot's behavior while performing acrobatic maneuvers can be represented as a *maneuver automaton*. Such an automaton has two types of primitives: trim primitives, which correspond to steady states or relative equilibriums, and maneuvers, which are arbitrary trajectories that start and end at steady state conditions. This automaton can be used with other motion planning methods, such as Rapidly-Exploring Random Tree (RRT), to deal with situations where dynamic environments and systems with complex dynamics are involved [3, 4]. It is worth pointing out here that, although

the concept of motion primitive is defined over symmetries existing in vehicle dynamics, if generalized appropriately, it can include other factors, such as environments. This possibility promises a path where environment-driven method and control-driven method can be potentially integrated. Our definition of *guidance primitive*, which will be elaborated on in Chapter 5, follows this idea.

1.3.2 Movement Primitive Method

Paralleled to motor primitives, a concept called movement primitives[11] has been developed in robotic movement control, mainly in the context of learning from demonstration (LFD) [52, 80, 81, 82, 83, 84]. As explained in Section 1.1.2, biases are needed in LFD to cope with the curse of dimensionality. The role of movement primitives is to introduce biases by restricting the functional form of the controller. These movement primitives are predefined and each one of them can be formalized as a dynamical system, which is in turn characterized by some parameters. Using these movement primitives, a full scale learning problem can then be reduced to the selection of the right primitives and parameters. This reduction can greatly reduce the complexity involved in planning and control, because now the planning and control is carried out over the movement primitive space rather than over the much larger joint space [52].

Movement primitives can be combined in two different formulations: parallel or sequential. In the parallel formulation, each movement is represented by a weighted summation of movement primitives. A movement primitive is in the form of either apoint attractive behavior, which corresponds to a discrete movement, such as reaching, or a limit cycle behavior, which corresponds to a rhythmic movement, such as walking [52, 81, 82]. One problem with the parallel formulation is that the recognition of the parallel primitives is quite complex especially in situations with general movements [28]. In the sequential formulation, each movement is represented as a succession of movement primitives [80, 83, 84, 85]. Compared with the parallel formulation, this formulation is much easier to identify. The learning process can be completed by following these three steps: behavior segmentation, where a demonstration is divided into smaller segments; behavior recognition, where each segment is assigned with a particular primitive; and behavior coordination, where the switching rules between the primitives

[11] These are just different names used by different communities. Their meanings are exchangeable.

are identified [28].

Please refer to paper [86] for a more detailed discussion of this topic.

1.3.3 Human Behavior Analysis

Human behavior analysis has been studied in a variety of fields, such as computer vision and human-machine interaction [87, 88, 89, 90, 91, 92]. To understand the role of human behavior analysis in applications, let's take human-machine interaction for instance [90, 91]. Suppose we are designing a supervisory control system for UAVs, if the system can recognize and predict the operator's behavior, it can adjust its own behavior so as to serve operator's needs much better. Human behavior analysis can also be integrated with LFD [28]. The reason for such an integration is obvious, since, the first step of any LFD algorithm is always the analysis of the human's behavior for the purpose of smoothly transferring skills from a human to a robot[12] .

Based on their applications, human behavior analysis studies can be divided into two major areas: human behavior recognition [87, 88, 89] and human behavior prediction [90, 91, 92]. One key question in both these two areas is the mathematical representation of human behavior[13] . The most commonly used representation class is stochastic models, which represent human behavior as stochastic dynamical systems. The state evolutions of these dynamical systems are governed by stochastic processes and the observations are corrupted by noises. Among all the stochastic models, the mostly commonly used are Gaussian mixture models (GMMs) [93] and Hidden Markov models (HMMs) [94]. A GMM represents human behavior through a multivariate Gaussian distribution. It only describes spatial variances. In order to encode the temporal evolution of human behavior, some additional steps, such as adding time as another variable, must be taken [95, 96]. In the following, we are going to focus on the HMM.

A HMM represents human behavior through a set of hidden states. At each time

[12] The analysis of human behavior can help us figure out the question of what is learned. It is worth pointing out, though, that the analysis phase in most LFD algorithms is rather ad hoc. In most occasions, the structures about human behavior, which are derived from the analysis stage and used subsequently in the learning stage, are prescribed by humans. Such a practice has not prevented the successful applications of the LFD. However, having a more accurate model with structures that are inherent to human behavior is always a plus.

[13] Please notice that while behavior representation is essential in behavior prediction, it sometimes does not occupy the core of behavior recognition. In the latter case, behaviors only need to be classified and the key to find the most discriminant features.

frame, the behavior is assumed to be in one of these states. The state evolves according to a stochastic state transition probability. These hidden states can not be observed directly, they need to be inferred from observations. The observations and hidden states are associated with an observation probability. With a HMM, given a time history of human behavior, the transition and observation probabilities can be trained via algorithms such as Expectation Maximization (EM). Models learned from data can then be used in either human behavior recognition or human behavior prediction. Let's take behavior recognition for example. Suppose models corresponding to different activities have been learned. Given a new sequence of observations, we can recognize the most possible activity by first computing the likelihood of generating the sequence by each model and then choosing the model with highest likelihood.

HMMs used in human behavior analysis can be classified into two types of models: single-layered models or hierarchical models [87]. Single-layered models, which are also called flat models, are based directly on sets of features. For instance, in [97], for the recognition of different tennis strokes, grid-based silhouette mesh features are used. Baum-Welch algorithm is used to train the HMM and Viterbi algorithm is used to compute the likelihood of an observed sequence. One issue with single-layered models is that they typically perform purely to model complex behavior, because these models can not capture the hierarchic or shared structures that are inherent to some complex behaviors. Hierarchical models, on the other hand, are meant to describe these kinds of complex behaviors. In these models, higher level human activities, such as dancing, are described with other simpler sub-events, such as moving a leg. By using a hierarchical representation, the training process becomes computationally tractable. For instance, in [98], to recognize complex behaviors in an indoor environment, a two-layered hierarchical hidden Markov model (HHMM) is built. The top level consists of complex behaviors, such as having a short meal or having a normal meal, and the bottom level consists of primitive behaviors. Each of these primitive behaviors represents a people's action from one landmark to another. Rao-Blackwellised particle filter is used to train the HHMM. Similar models have been used in [99] to represent robot's behavior in an indoor environment and in [100] to infer a person's activities in a large and complex environment from GPS data.

1.4 Research Statement

As illustrated in Section 1.1, to better understand how humans organize their spatial behavior, especially when they are executing challenging tasks, requires a *unified* framework where the role of each separate component can be studied within the context of the overall behavior. In addition, if such a framework is *formal*, i.e., it is endowed with some mathematical languages, this would allow the researchers to codify and study human behavior within a rigorous framework as well as to facilitate the knowledge transformation from the scientific study of human behavior to the engineering design of more efficient and effective autonomous guidance and control systems. One key to guarantee the knowledge transformation is smooth is that the framework should be *grounded*, i.e., the concepts or the structures (building blocks) used in the framework should be physically, biologically and neurologically plausible, meaning that they should be based directly on the evidence from the fields of physics, biology and neuroscience. In conjunction with knowledge gained from these fields, the best way to achieve groundedness is through the data-driven methodology, which, in the context of human behavior study, means that the concepts or the structures should emerge from experimental data. They should not be hand-crafted, like the concepts or the structures used in human behavior analysis as described in Section 1.3.3. It has been shown in paper [101] that, at least for the case used in that paper, models learned from experimental data perform better than the ones prescribed by humans. To conclude, from both a scientific and an engineering perspective, we are in need of a *unified*, *formal* and *grounded* framework to model and analyze human behavior.

Existing frameworks for the study of human and animal behaviors, as described in Section 1.2, do not satisfy all these requirements. Most non-representational approaches provide unified explanations of perception-action behavior. However, the behavior is almost always formulated in terms of some low-dimensional dynamical system, the meaning of which is often ad hoc (i.e., non-representational approaches are not grounded). In addition, they lack the ability to explain complex behaviors, such as those that are constrained not only by the immediate environment, but also by some remote goal states [35]. Model-based approaches do not suffer from these shortcomings, but they tend to focus on either the perception side or the action side (i.e., they are not unified).

Moreover, the rigidity of the *sense-model-plan-act* structure (used often by model-based approaches) makes it challenging to explain the flexible and adaptive nature of human spatial behavior [35, 102].

In this dissertation, we integrate features from both model-based and non-representational approaches and develop a *unified*, *formal* and *grounded* framework to model and analyze human guidance behavior. This framework provides a unified language within which the components within the perception-action loop can be investigated. Furthermore, due to the definition of *interaction patterns* (which will be briefly described in Section 1.5 and later elaborated on in Chapter 5), the structures or building blocks of our framework can be formally defined. Similar to motor primitives, these interaction patterns provide a link between higher-level control and lower-level control, thus enabling a hierarchical model of human guidance behavior. Different from the hierarchical models described in Section 1.3.3, though, the building blocks of our model as well as the model itself are derived from experimental data with only minimal human interventions. Finally, it is worth pointing out that the framework can also be used to describe the guidance behavior of artificial agents, such as UAVs and mobile robots. This feature is potentially beneficial for the design of future autonomous systems.

In this dissertation, the following research questions will be answered:

- How can we define guidance behavior in a way that is formal, which means that it could be the subject of rigorous mathematical investigation, and, at the same time, general, which means that it could be used to describe the behaviors of humans as well as artificial agents?
- What are the roles of agent dynamics, environments and performance criteria in the guidance behavior?
- Can we build a framework where the perception-action behavior can be investigated in a *unified* form, where the building blocks of this framework as well as the framework itself can be *formally* studied, and where the building blocks are inherent to guidance behavior (thus *grounded*) instead of being imposed by humans?
- Can we design experiments with which human guidance behavior can be scientifically studied? Is there a way we can develop algorithms which can analyze the

collected experimental data based on the proposed formal framework?

- Finally, what conclusions can be reached from the empirical analysis? What are the implications for the understanding of human guidance behavior and the design of autonomous systems?

1.5 Dissertation Outline and Contributions Per Chapter

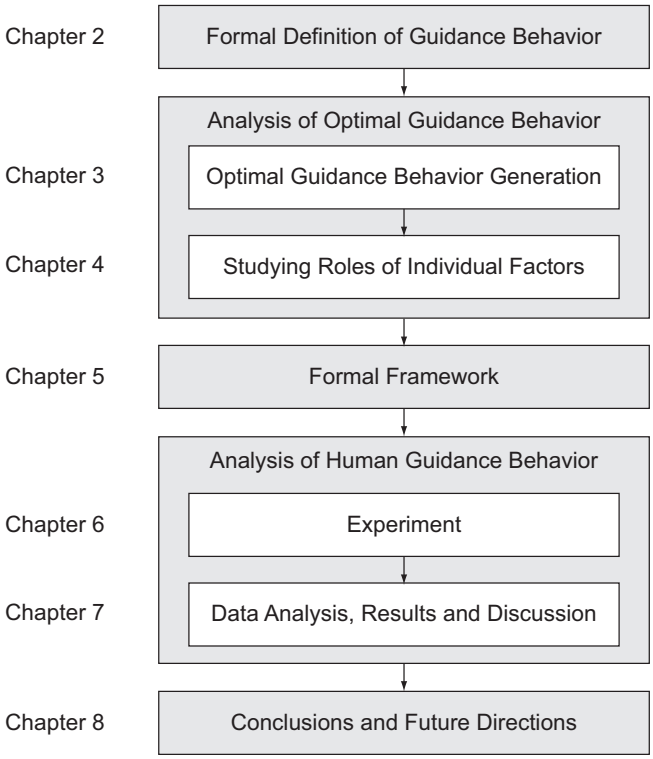


Figure 1.5: Dissertation outline.

As shown in Fig. 1.5, the dissertation is organized as follows. The contributions of each chapter are also outlined.

- **Chapter 2**

  To build the foundations of a formal and unified framework for the modeling and analysis of guidance behavior, the first thing we need to do is to define *guidance behavior*. In order to take perception, control and action all into consideration,

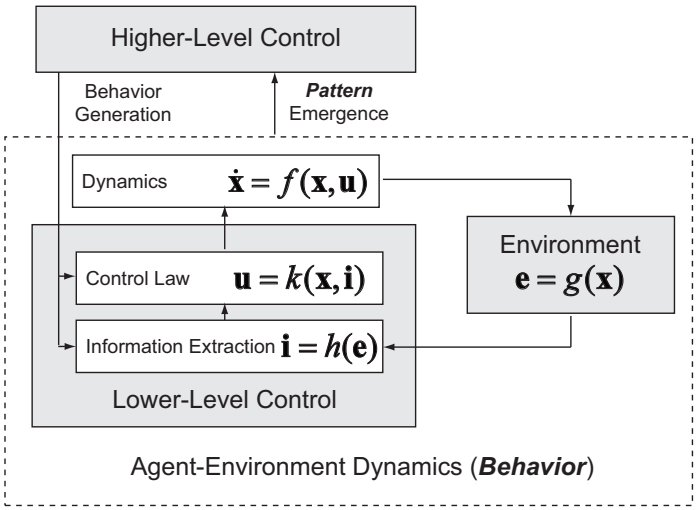


Figure 1.6: Illustration of the closed-loop agent-environment interactions, their manifestation as interaction patterns, and finally how these patterns are used within a hierarchical model of the guidance behavior.

in this chapter, as shown by the part encircled by the dotted lines in Fig. 1.6, we define guidance behavior as the collection of all the closed-loop dynamics resulting from an agent's interaction with the environment. We show that such a definition is consistent with the embodied and embedded view of human behavior. This definition also allows us to employ mathematical languages from dynamical system and control theory to describe and analyze the agent's behavior, which have shown their powers in computational motor control as described in Section 1.2.1.

- **Chapter 3**

  In this chapter and the next chapter, we analyze the guidance behavior of agents obeying optimality principles. These two chapters mainly serve two purposes: one is to get a more comprehensive understanding of the interactions between the agent and the environment by analyzing the roles of individual factors, such as agent dynamics, environments and performance criteria, on the whole guidance behavior; the other one is to inspire the necessary concepts, such as *patterns*, which will be used in later chapters.

In this chapter, we describe a computational framework where the optimal guidance behaviors of different agents, environments and performance criteria can be generated. This framework is based on the works of computing cost-to-go (CTG) functions by using *motion primitive automaton* (MPA) [4, 103, 104, 105]. However, in this chapter, we extend the framework from 2D to 3D; we also extend the framework from working only with time criterion to others, such as energy and length; furthermore, we extend the framework to work with a popular map format, Digital Terrain Elevation Data (DTED) map, which is available in the public domain for a variety of terrains and areas [106]; finally, by comparing the results obtained by using this framework with the ones obtained by using nonlinear programming, which can be considered as optimal, we are able to study the effects of quantization on the performance of this framework.

- **Chapter 4**

  In this chapter, based on the optimal guidance behaviors we obtained for different agents, environments and performance criteria in the previous chapter, we are able to study the roles of these individual factors. Three small-scale UAV types are used: a fixed-wing aircraft, a standard helicopter and a quad-rotorcraft. And two performance criteria are used: time and energy. The computed optimal guidance behaviors are then analyzed by two methods: based on trajectories and based on what we call *spatial cost-to-go* (SCTG) maps. These maps describe the spatial distributions of optimal state and cost-to-go for a given geographical environment and they embed the interaction effects between vehicle dynamics and the environment. We show that agent dynamics play an important role in guidance behavior, and they should not be ignored during the study of guidance behavior or the design of autonomous guidance systems.

- **Chapter 5**

  This dissertation has three different parts: a theoretical one, an experimental one and an analytical one. This chapter, together with Chapter 2, introduces the theoretical part. It elaborates on the *unified*, *formal* and *grounded* framework that we intend to build for the modeling and analysis of guidance behavior. In this chapter, we first use the guidance behavior of Dubins' vehicle as an example to

introduce some of the key concepts. We then use the formal definition of guidance behavior as described in Chapter 2 to bring out two types of equivalences or symmetries that are inherent to guidance behavior. These equivalences or symmetries are developed from some existing concepts, motion primitives [4] and causal states [107]. However, we unify them by using dynamical system language and we also modify them so that they can be used to describe agent-environment interactions. Furthermore, based on these two equivalences, we propose the concept of *interaction pattern*, which is the key of our formal framework and serves the role of a keystone where theoretical (Chapter 2 and this chapter), experimental (Chapter 6) and analytical (Chapter 7) works meet.

The overall framework can be illustrated by Fig. 1.6. We use the equivalences or symmetries that are inherent in the dynamic interactions between the agent and the environment as the basis for the formalization of human guidance behavior. These equivalences are then integrated under the concept of *interaction patterns*. These patterns are the building blocks with which guidance behavior is organized. The patterns provide a way that general behaviors can be decomposed into less complex, stereotypical components. They also enable the codification system needed for the formal modeling of guidance behavior. Together, these concepts and building blocks describe the natural hierarchical organization of behavior and provide the basis of a fundamental framework needed to understand and model human guidance behavior.

- **Chapter 6**

  This chapter describes the experimental part of this dissertation. In this chapter, we introduce our lab facility, the experimental procedures and data pre-processing procedures. We also give some preliminary analysis results of the experimental data (The data will be more systematically analyzed in the next chapter). Compared with the experiments typically conducted in motor control, some of which are described in Section 1.2.1 or in locomotion, some of which are described in Section 1.2.2, in our experiments, the subjects need to solve problems that are more challenging and beyond skill levels required by everyday tasks. In order to replicate real world guidance tasks as closely as possible, they are asked to perform agile guidance tasks in cluttered lab environments with remote control

helicopters, which have complex dynamics. They need to negotiate the various factors ranging from the vehicle dynamics, environments and task requirements. Our lab facility also allows us to collect data from a variety of domains, such as the pilots' controls, the helicopters' dynamics and the pilots' gaze locations. The richness of the data provides an opportunity to get a more comprehensive view of human's guidance behavior.

- **Chapter 7**

  This chapter covers the core analytical part of this dissertation. In this chapter, based on the formal framework introduced in Chapter 5, algorithms from system identification and machine learning are adopted or developed to extract the interaction patterns from the experimental data collected as described in Chapter 6. These patterns are then used to build a *hierarchical hidden Markov model* (HHMM) of human guidance behavior. The patterns and the model are then used to help us gain insights into the generation mechanism of human guidance behavior. Compared with the human behavior model as presented in Section 1.2 and 1.3.3, our model is unified due to the definition of guidance behavior, which takes both perception and action into consideration; it is also formal, thanks to the concept of interaction patterns and also the HHMM formulation; finally, it is grounded. The components of this model, the interaction patterns, are extracted from experimental data; although the organizational principle of these patterns is not derived via data-driven methods, we are able to partially confirm the principle by, for instance, comparing the pilots' gaze patterns to the switches as predicted by the principle.

- **Chapter 8**

  In this chapter, we gives key conclusions to the work presented in this dissertation. We also lay out some of the possible future directions of the work.

1.6 List of Publications by Chapters

The chapters of this dissertation have first appeared as various publications. Below are the publications corresponding to each of the chapters[14] :

- **Chapter 2**

  Z. Kong and B. Mettler, "Foundations of Formal Language for Humans and Artificial Systems Based on Intrinsic Structure in Spatial Behavior", *IEEE International Conference on Intelligent Robots and Systems (IROS)*, San Francisco, CA 2011.

- **Chapter 3**

  Z. Kong, V.R. Korukanti and B. Mettler, "Mapping 3D Guidance Performance Using Approximate Optimal Cost-to-go Function", *AIAA Guidance, Navigation, and Control Conference (GNC)*, Chicago, IL, 2009.

  B. Mettler, Z. Kong, C. Goerzen, and M. Whalley, "Benchmarking Guidance Performance for Autonomous Rotorcraft", *Journal of the American Helicopter Society*, accepted.

- **Chapter 4**

  Z. Kong and B. Mettler, "Evaluation of Guidance Performance in Urban Terrains for Different UAV Types and Performance Criteria Using Spatial CTG Maps", *Journal of Intelligent and Robotic Systems*, Volume 61, Issue 1, Page 135-156, 2011.

- **Chapter 5**

  Z. Kong and B. Mettler, "On the General Characteristics of 2D Optimal Obstacle-Field Guidance Solution", *IEEE Conference on Decision and Control (CDC)*, Shanghai, China, 2009.

  Z. Kong and B. Mettler, "Interaction Patterns in Agent-Environment Dynamics and Foundations for Formal Modeling of Human Guidance Behavior", *IEEE Transactions on Systems, Man, and Cybernetics*, submitted.

- **Chapter 6**

[14] Our other publications that are not included in this dissertation are [23, 105, 108, 109, 110, 111, 112]

Z. Kong and B. Mettler, "An Investigation of Spatial Behavior in Agile Guidance Tasks", *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Anchorage, AK, 2011.

- **Chapter 7**

  Z. Kong and B. Mettler, "Interaction Patterns in Agent-Environment Dynamics and Foundations for Formal Modeling of Human Guidance Behavior", *IEEE Transactions on Systems, Man, and Cybernetics*, submitted.

  Z. Kong and B. Mettler, "Modeling and Prediction of Human Guidance Behavior Based on Agent-Environment Interaction Patterns", *The 2013 IEEE International Conference on Robotics and Automation (ICRA)*, submitted.

Chapter 2

Formal Definition of Guidance Behavior: Closed-Loop Agent-Environment Dynamics

2.1 Introduction

In this chapter, we define guidance behavior as the closed-loop dynamics resulting from the agent's interaction with the environment. Such a definition provides a foundation to study guidance behavior formally. Chapter 5 will show how a formal framework for the modeling and analysis of guidance behavior can be built based on this definition.

The material presented in this chapter is first published in papers [113, 114].

2.2 Agent-Environment Dynamics

2.2.1 Agent Dynamics and Constraints

The dynamics of an agent S can be described as:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \tag{2.1}$$

where $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state of the agent, $\mathbf{u} \in \mathcal{U} \subseteq \mathbb{R}^m$ is the control, which is assumed to be piecewise-continuous. For guidance problems, dynamics described in (2.1) usually

includes not only the dynamics of the musculoskeletal system but also the dynamics of the vehicle. For the agent, given an initial state $\mathbf{x}_0$ and a control trajectory $\mathbf{u}(.)$, the state trajectory $\mathbf{x}^{\mathbf{u}}(.;\mathbf{x}_0)$ is the solution to the following equation:

$$\mathbf{x}^{\mathbf{u}}(t;\mathbf{x}_0) = \mathbf{x}_0 + \int_0^t f(\mathbf{x}^{\mathbf{u}}(\tau;\mathbf{x}),\mathbf{u}(\tau))\mathrm{d}\tau,\ \forall t \in [0,t_f] \tag{2.2}$$

In what follows, we will use $\mathbf{x}(t)$ instead of $\mathbf{x}^{\mathbf{u}}(t;x_0)$ to represent the state trajectory.

To complete a guidance task, the agent must satisfy two sets of constraints. First of all, there are constraints on the possible evolution of the agent's state. This kind of constraints is called *internal constraints*. These constraints, such as the flight envelope constraints on the aircraft, are normally imposed to guarantee the safe operation of the agent. And they are closely related to the agent and its dynamics, especially its physical limitations, such as the actuator saturation of the musculoskeletal system. Internal constraints can be encoded in the form of a set of inequalities:

$$F(\mathbf{x}(t),\mathbf{u}(t)) \leq 0,\ \forall t \in [0,t_f] \tag{2.3}$$

Second, there are also constraints that are not directly related to the agent and its dynamics. And we call these constraints as *external constraints*. They are related to various external sources that the agent needs to avoid, such as buildings, other vehicles or no fly zones. External constraints can also be encoded as a set of inequalities:

$$G(\mathbf{x}(t)) \leq 0,\ \forall t \in [0,t_f] \tag{2.4}$$

2.2.2 Agent-Environment Interactions

In order to complete a guidance task, due to the dynamic, uncertain or even hostile nature of the environment, the information flow between the agent and the environment should not be just in one direction, either from the agent to the environment or from the environment to the agent. Here we are going to take an embodied and embedded view of guidance, which, as explained in Section 1.2.2, means that the agent has a physical body and is embedded in the physical environment[1] . This means that the

[1] Although here we understand the embodied and embedded view in the context of guidance, it should be noticed that this view can be interpreted in a more general sense, such as embodied cognitive science [57].

traditional unidirectional information flow view is not sufficient; the agent is *coupled* with the environment and the information flow is bi-directional [35].

The agent is coupled with the environment in two different ways. First, when the agent's behavior unfolds in the environment, the agent's own dynamics affects the environment, which can be described by some environment states. This process is described as:

$$\mathbf{e}(t) = g(\mathbf{x}(t)) \tag{2.5}$$

In guidance problems, the environment state $\mathbf{e}(t)$ is mostly expressed in the form of some quantities that relate the agent with some objects in the environment, such as the distance between the agent's current position and a specified reference location in the nearby environment[2] .

Second, as is the case in non-representational approaches [2, 35, 58, 60], for the purpose of guiding actions, the perceptual apparatus must be able to extract information from the environment. This process can be described as:

$$\mathbf{i}(t) = h(\mathbf{e}(t)) \tag{2.6}$$

While the environment state $\mathbf{e}(t)$ is general and is the results of some physical processes, the definition of the extracted information $\mathbf{i}(t)$ and therefore the perceptual process is task specific. $\mathbf{i}(t)$ does not have to be an exact copy of the 3D environment. For most of the everyday tasks humans perform, $\mathbf{i}(t)$ resides in a much lower dimensional space than $\mathbf{e}(t)$. Such a task-specific perspective allows the agent to adopt special-purpose solutions that make minimal demands on representation rather than general-purpose solutions that require elaborate representation of the environment [35].

2.2.3 Control

As shown in Fig. 2.1(a), in order for an agent to perform a guidance task, it should be able to perceive the world via (2.6), respect all the constraints encoded by (2.3) and (2.4), and choose an appropriate action sequence $\mathbf{u}(t), t \in [0, t_f]$ that will bring it from

[2] It should be noticed that the agent state $\mathbf{x}(t)$ and the environmental state $\mathbf{e}(t)$ do not have to be of the same order. In Chapter 5, we define $\mathbf{e}(t)$ in such a way that the space containing it will be a subspace of the state space $\mathcal{X}$.

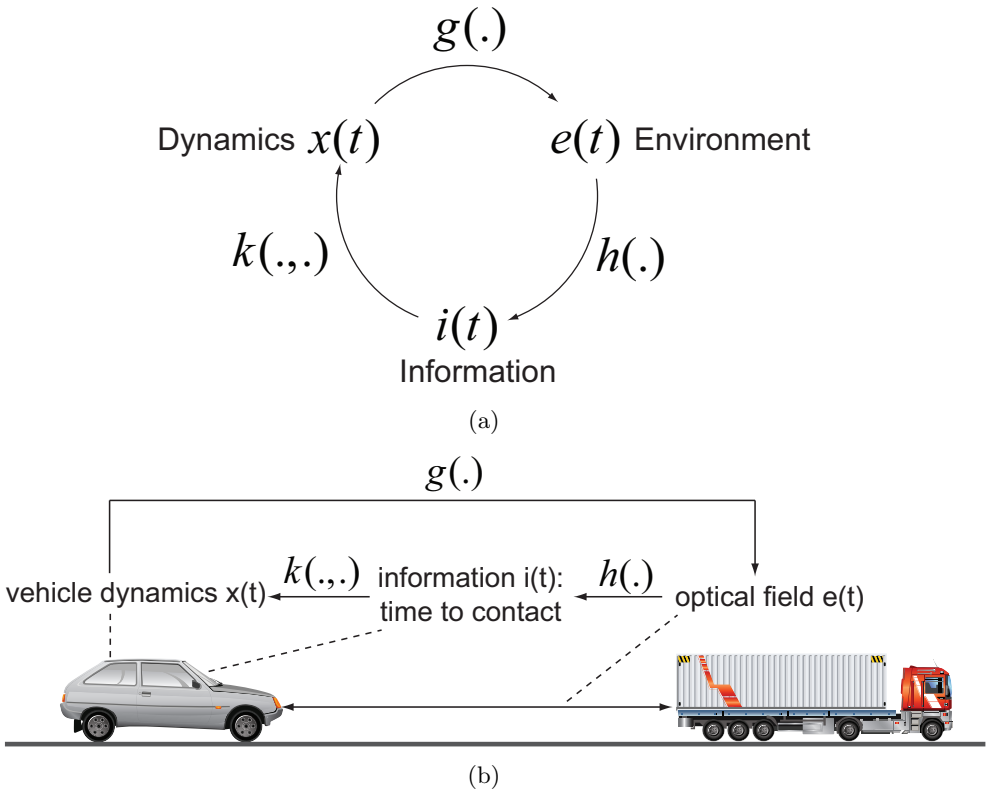


Figure 2.1: (a) Guidance behavior resulting from the agent-environment interactions and (b) a driving example.

its initial state $\mathbf{x}_0$ to the desired goal state $\mathbf{x}_f$. The process of choosing the right action can be written in terms of a control policy as follows:

$$\mathbf{u}(t) = k(\mathbf{x}(t), \mathbf{i}(t)) \tag{2.7}$$

Here the control policy $k(.,.)$ is task specific and it maps the continuous state vector $\mathbf{x}(t)$ and information vector $\mathbf{i}(t)$ to a continuous control vector $\mathbf{u}(t)$.

From a computational standpoint, there are a number of methods such a control policy can be represented and acquired [115]. It can be obtained by using optimization theory. For instance, in the motion planning formulation [29], some criterion $r(\mathbf{x}(t), \mathbf{u}(t), t)$ can be chosen. This criterion can evaluate the quality of a control $\mathbf{u}(t)$ in a particular state $\mathbf{x}(t)$. And a control policy $k(.,.)$ is chosen in a way that this criterion is minimized. It can also be represented as a feedback controller. In such a case, some

strong priori assumptions about the control policy, most commonly in the form of a desired trajectory, are made. Both these two methods are popular in motor control community [46, 115, 116].

2.2.4 Definition of Guidance Behavior

As shown in Fig. 2.1(a), the closed-loop agent-environment dynamics can be obtained by combining (2.1), (2.5), (2.6) and (2.7) as follows:

$$\dot{\mathbf{x}} = f(\mathbf{x}, k(\mathbf{x}, h(g(\mathbf{x})))) \tag{2.8}$$

The *guidance behavior* is defined by the collection of all agent's state trajectories (2.2), taken together with the corresponding environment state trajectories (2.5), resulting from the closed-loop dynamics. Any trajectory belonging to this collection is called a *valid* guidance behavior.

Such a definition provides us a foundation where the guidance behavior can be studied formally. Just as the perception-action dynamics as defined in paper [35], the closed-loop agent-environment dynamics as shown in Fig. 2.1(a) enables us a hierarchical way of analyzing guidance behavior. At the level of agent-environment dynamics (the lower level), we can analyze the physical constraints on the agent, the way the agent affects the environment, the information that is necessary for the completion of the task, the control policy chosen by the agent to use the information to regulate the action. And, as will be elaborated on later in Chapter 5, certain regularities or patterns emerge from this agent-environment dynamics. And at the higher level, we can analyze the way the patterns emerge, the characterization of the patterns and the principle employed by the agent to organize these patterns in order to generate appropriate guidance behavior[3]

.

[3] In paper [35], the higher level is described in terms of a number of behavioral variables. The corresponding dynamics associated with these variables is called *behavioral dynamics*. Such a formulation allows the application of tools from dynamical system theory, such as bifurcation. In our case, at the higher level, the agent-environment dynamics is studied in terms of pattern, which is a more general concept than behavioral dynamics and allows us to employ tools from formal methods as well as dynamical system and control theory.

2.3 An Example: Driving

In this section, we use a driving example as shown in Fig. 2.1(b) to explain agent-environment dynamics. The goal for the driver of the car could be, for instance, to maintain a safe distance from the truck in front of his or her car. Here the agent encapsulates both the driver and the car. Together they constitute a system which can be described by an equation (2.1). The state of this system can be of the driver, such as those related to muscle activation, or of the car, such as the speeds of the four wheels.

The agent interacts with the environment in two ways. First of all, the agent affects the environment. For this particular case, the dynamics of the agent affects environmental states $\mathbf{e}(t)$, such as the relative distance between the car and the truck. This change of distance further affects the optical field, which is of high dimensional. Second, the driver needs to exact information $\mathbf{i}(t)$ from the interactions between the agent and the environment. Based on perceptual guidance principles, the information $\mathbf{i}(t)$ required to perform the driving task is simply $\dot{\tau}$, the time derivative of the time-to-closure [2]. In other words, in order to adjust a car's speed to prevent a collision with another car, a driver does not need to perceive the velocity of her own car or the velocity of the car in front, nor the distance between the two cars. For this example, the time-rate of the time to close the gap between the two cars $\dot{\tau}$ is all the information required to guide the motion. In addition, to get $\dot{\tau}$, no significant mental computation has to be performed; this information can be extracted directly from a sequence of retinal images [2].

Based on the agent state and the information obtained from the interactions, the driver can then choose a control policy $k(.,.)$. The control policy defined by us is not necessarily a simple feedback control as those used in motor control community. (This point will be elaborated on later in Chapter 7.) The control policy defined by us can possess some complicated forms, such as hierarchical forms. The complicated forms imply that some high-level cognitive functions are involved in the generation of our control policy, which is difference from motor control where generally only low-level functions are involved. Furthermore, the control policy defined by us is acted upon the information $\mathbf{i}(t)$, which is extracted actively from the environment, while in motor control the control policy is usually passively fed with some direct environmental states, which are essentially $\mathbf{e}(t)$ as defined in (2.5).

Chapter 3

Generation of Optimal Guidance Behavior for Different UAV Types and Performance Criteria

3.1 Introduction

In the previous chapter, we defined guidance behavior as the collection of closed-loop agent-environment dynamics. As described in Section 2.2.3, the control policy chosen by the agent can be represented and acquired in a number of ways. In this chapter and the next chapter, we adopt the optimal theoretical formulation. The purpose of these two chapters are three-folds: first, to characterize the guidance behavior of agents obeying optimality principles; second, to analyze the roles of agent dynamics and performance criteria[1] on the guidance behavior; third, to inspire some concepts that can be generalized to the description and analysis of general agents' guidance behavior.

We use three different UAV types, a quadrotor, a standard helicopter and a fixed-wing aircraft, and two different performance criteria, time and energy, as examples. Except the assumptions that these UAVs obey optimality principles, we also assume they have a priori knowledge of the environment. It should be pointed out that, despite being restricted to scenarios with optimal behavior and a priori known environments,

[1] Performance criteria can be used interchangeably with performance metrics or performance indexes.

the analysis framework presented in this chapter and the next chapter can easily be extended to scenarios where the environments are uncertain and the behavior is not optimal. In this chapter, we first introduce a guidance behavior generation and guidance performance evaluation framework based on the optimal control formulation. In particular, we introduce the concept of spatial cost-to-go (SCTG) map, which is an approximation of the optimal cost-to-go function and the associated optimal states. Section 3.3 first gives a brief overview of the three UAVs used in this chapter and the next chapter (the Hornet UAV, the MD4-200 UAV and the Raven UAV). It then describes the approach used to build the motion primitive automaton (MPA) models based on the UAV type. The section also provides a short description of the algorithm used to compute the SCTG maps. Finally, in order to evaluate the performance of our MPA framework, Section 3.4 compares the solutions obtained by using nonlinear programming and the solutions obtained by using the MPA framework.

The material presented in this chapter is first published in papers [110, 117, 118, 119, 120].

3.2 Guidance Behavior Generation and Guidance Performance Evaluation Framework

This section gives an introduction to our guidance behavior generation and guidance performance evaluation framework. Our framework is based on optimality principles and the core of our framework is the concept of spatial cost-to-go (SCTG). Our framework also allows a situated perspective of understanding and analyzing guidance behavior, which is consistent with the guidance behavior definition presented in Chapter 2. The implementation details regarding this framework will be presented in the next section.

3.2.1 Guidance Performance Evaluation and Existing Frameworks

Performance evaluation is a fundamental issue in the development of guidance systems for unmanned aerial vehicles (UAVs). Through quantitative evaluation, it is possible to measure the effect of different factors, including vehicle configurations, environments, and guidance and control system components. In addition, quantitative frameworks

can provide an unambiguous representation to verify results, thus helping to assess new applications and prevent duplication of efforts [121].

Up to now, most of the performance evaluation works in the field of unmanned vehicles have focused on gathering metrics [122, 123, 124]. Those metrics, though meaningful, are normally mission and vehicle dependent. So it is difficult to use them to compare the performance across different systems. From a system's acquisition perspective, it is important to be able to evaluate the performance of different competing platforms. Such a need is especially pressing nowadays due to the rapid development of a wide range of UAVs based on a variety of airframe configurations with different dynamic capabilities [125, 126]. Furthermore, the majority of existing evaluations are performed based on flight data. The conclusions obtained from these evaluations, therefore, do not provide a direct feedback during the design process. Ideally, a quantitative evaluation framework should allow studying the effect of changes in airframe configurations, system and mission parameters, etc., thus providing insights into ways to improve the overall UAV design with respect to goals associated with mission profiles.

3.2.2 A Brief Overview of Our Framework

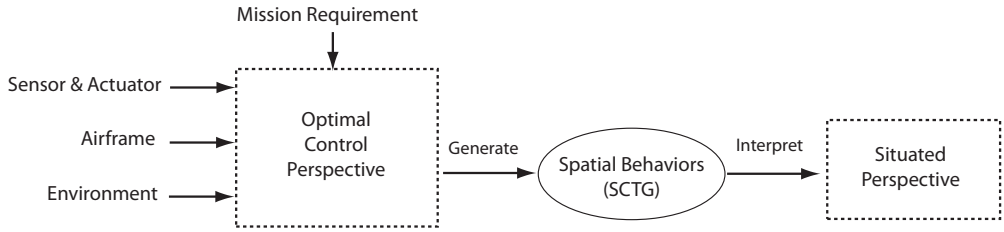


Figure 3.1: Our guidance behavior generation and guidance performance evaluation framework.

As shown in Fig. 3.1, our framework is based on optimal control formulation and enables quantifying influences of several individual factors on the overall performance. Factors of interests include dynamic capabilities, environment characteristics and mission requirements. The framework's key features are the following. First, it is based on an optimal control perspective [48]. Optimal control principles are used to generate behaviors given a selected UAV system, an environment, and a mission. This provides a basis to evaluate and compare performance across different vehicles and to investigate

performance differences with respect to environment characteristics and choices of performance criteria. Second, the approach adopts a so-called *situated perspective*, which provides a way to study the guidance behavior as a function of environmental factors similar to the approach used in ecological psychology [34].

In our framework, autonomous guidance is first formalized as an optimal control problem, which serves as a principle for driving the interaction of the UAV system and the operational environment. Instead of generating individual trajectories, a spatial cost-to-go (SCTG) map is used to describe the vehicle's autonomous guidance behavior. STG map, which provides a spatial description of performance, can then be analyzed to obtain insights into phenomena that would otherwise be difficult to identify by studying individual, point-to-point trajectories. The specific analysis results will be illustrated in the next chapter.

3.2.3 Autonomous Guidance as Optimal Control Problem

Classical aerospace guidance problems, such as orbit transfer of satellite or trajectory planning of long-range missiles, are usually solved as optimal control problems. The autonomous guidance of UAVs, especially those operating within cluttered environments, are much more challenging to solve as fully specified optimal control problems [23]. The reason for this is that guidance problems formulated as trajectory optimization problems are intrinsically NP-hard [127], which means that the computational complexity grows exponentially with the order of the vehicle's dynamics and the size of the environment. Since performance evaluation can be done offline, computational efficiency is less of a concern when used for an evaluation framework.

Formulating guidance problems as optimization problems has a number of benefits. First, it provides a way to account for the performance of each system components and the performance of the whole system within its operational environment. Such a formulation, thus, allows to analyze the relative contributions of the different components. Second, the optimization will drive the system performance and hence provide an opportunity to understand how vehicle flight-dynamic capabilities play out in the overall behavior. For instance, as we will show later, once a mission and its typical operational environment is given, such a formulation makes it possible to generate optimal behaviors for different UAVs thus providing a quantitative basis to compare their

performances. Third, the optimal control framework provides a formal computational architecture and also a mathematical language to generate and describe operationally meaningful guidance behaviors. Although UAVs may not always operate according to their theoretically 'best' performances, it is a fact that they cannot exceed the optimal performance. Thus a framework based on optimal control allows setting a clear and objective upper bound on what is possible and then allows investigating how these 'best' performances are affected by different factors. Finally, an optimal control formulation provides the necessary foundations for future extensions of the framework. In particular to account for stochastic factors such as environment uncertainty or disturbances. Such a statistical approach would ultimately provide a way to capture real world conditions.

A general trajectory optimization formulation of a guidance task is to determine a control history $\mathbf{u}(t)$ which will drive the vehicle from its current state $\mathbf{x}_0$ to a desired goal state $\mathbf{x}_{\text{goal}}$ while minimizing a chosen performance objective of the form:

$$J_\infty(\mathbf{u}) = \int_{t_0}^{\infty} r(\mathbf{x}, \mathbf{u}) dt, \tag{3.1}$$

where r is the instantaneous cost function. And the optimal command history $\mathbf{u}^*(t)$ at the current state $\mathbf{x}$ is given by:

$$\mathbf{u}^*_\infty(t) = \text{argmin}\{J^*_\infty(\mathbf{x})\}. \tag{3.2}$$

3.2.4 Situated Analysis

As illustrated in the previous chapter, guidance involves constant interactions between an agent and its environment, both via sensors and actuators. From a cognitive standpoint, the guidance performance depends on how an agent is able to use the 'affordances' associated with these interactions. Affordances are the perceived properties of an environment that are used by an agent to determine its behavior [34, 128]. A particular urban environment will result in different 'affordances' for small UAVs (such as those investigated in this chapter and the next chapter) than for a large ones (such as the Global Hawk UAV). Affordances, therefore, depend not only on vehicles flight dynamics, but also on how these dynamic capabilities play out and can be utilized in a particular environment. To understand these phenomena, it is therefore necessary to study guidance as a spatial behavior. The idea of the framework is to describe the behavior as spatial

distributions, i.e., vector fields, so as to be able to analyze the spatial characteristics of the guidance performance and identify phenomena that are associated with the critical aspects of the agent-environment interactions.

3.2.5 Spatial Cost-to-go (SCTG) Framework

The concept of spatial cost-to-go function is first explicitly proposed in [111] as part of the receding horizon guidance framework. In this chapter and the next chapter, we use it as an analytical tool for studying guidance performance. The SCTG function is computed as an approximation of the problem's value function and its associated states based on a finite-state approximation of the vehicle dynamics and a discretization of the geographical environment. Such a formulation enables to account for state-dependent performance criteria such as energy. The details regarding the MPA model and SCTG computation are provided in Section 3.3.

Determining the value functions V^* involves solving the Hamilton-Bellman-Jacobi (HBJ) differential equation [129].

$$\frac{dV^*}{dt}(\mathbf{x}, t) = -\left\{ \frac{\partial V^*}{\partial \mathbf{x}} f(\mathbf{x}, \mathbf{u}) + r(\mathbf{x}, \mathbf{u}) \right\} \tag{3.3}$$

For most problems of practical interest, however, the HBJ equation cannot be solved analytically. Therefore approximate, computational techniques have to be used to obtain the value function. We call these approximate value functions together with their approximate solutions *spatial cost-to-go functions (SCTG)* [111]:

$$\Upsilon : x_f \mapsto \{V_S(x_f), v_S(x_f)\} \tag{3.4}$$

where $x_f \in F$ are the vector coordinates of a point in the geographical free space F. The corresponding optimal functional with respect to SCTG is:

$$\Upsilon^* : x_f \mapsto \{V_S^*(x_f), v_S^*(x_f)\} \tag{3.5}$$

where V_S^* is the optimal value function and v_S^* is the optimal velocity (see [111] for more details).

3.3 MPA Model and SCTG Computation

The motion primitive automaton (MPA) representation has originally been proposed as part of a hybrid guidance architecture [103, 104]. The benefit of the motion primitive formulation is the reduction in the complexity of the control problem. With the MPA formulation, an optimization problem becomes a sequential decision problem, which can be solved by dynamic programming.

In this section, we first provide a brief introduction of the UAVs used in the investigation and also the procedure used to determine their respective MPA models. Following, we provide a description of the terrain data and the SCTG algorithm.

	Hornet UAS	Raven UAV	MD4-200
Weight (kg)	1.10	1.90	1.1
Length (m)	0.63	0.90	0.91
Width (m)	N/A	1.40	0.91
Height (m)	0.24	N/A	0.20
Rotor Diameter (m)	0.71	-	0.37
Operating Altitude (m)	1 to 150	30 to 152	<150
Duration (min)	20	60 to 90	20
Min Speed (m/s)	0.00	8.89	0.00
Max Cruise Speed (m/s)	15.27	22.50	N/A
Max Climb Speed (m/s)	N/A	N/A	N/A

Table 3.1: Specification of the UAVs used in the investigation.

3.3.1 Overview of UAV Airframes

There are a variety of UAVs that are capable of performing surveillance and reconnaissance missions within urban terrain. In this chapter and the next chapter, we choose three representative platforms: the Raven UAV from AeroVironment [6], the Hornet UAV from Adaptive Flight [7] and the MD4-200 UAV from Microdrones GmbH [8]. Fig. 3.2 shows their respective pictures along with their corresponding power curves. Table 3.1 summarizes the key physical and performance data based on public domain information. These three UAVs are based on different airframe configurations and hence are expected to have different dynamic capabilities. For instance, the Hornet UAV and MD4-200 UAV are capable of hovering, which is critical for 'stare' type missions.

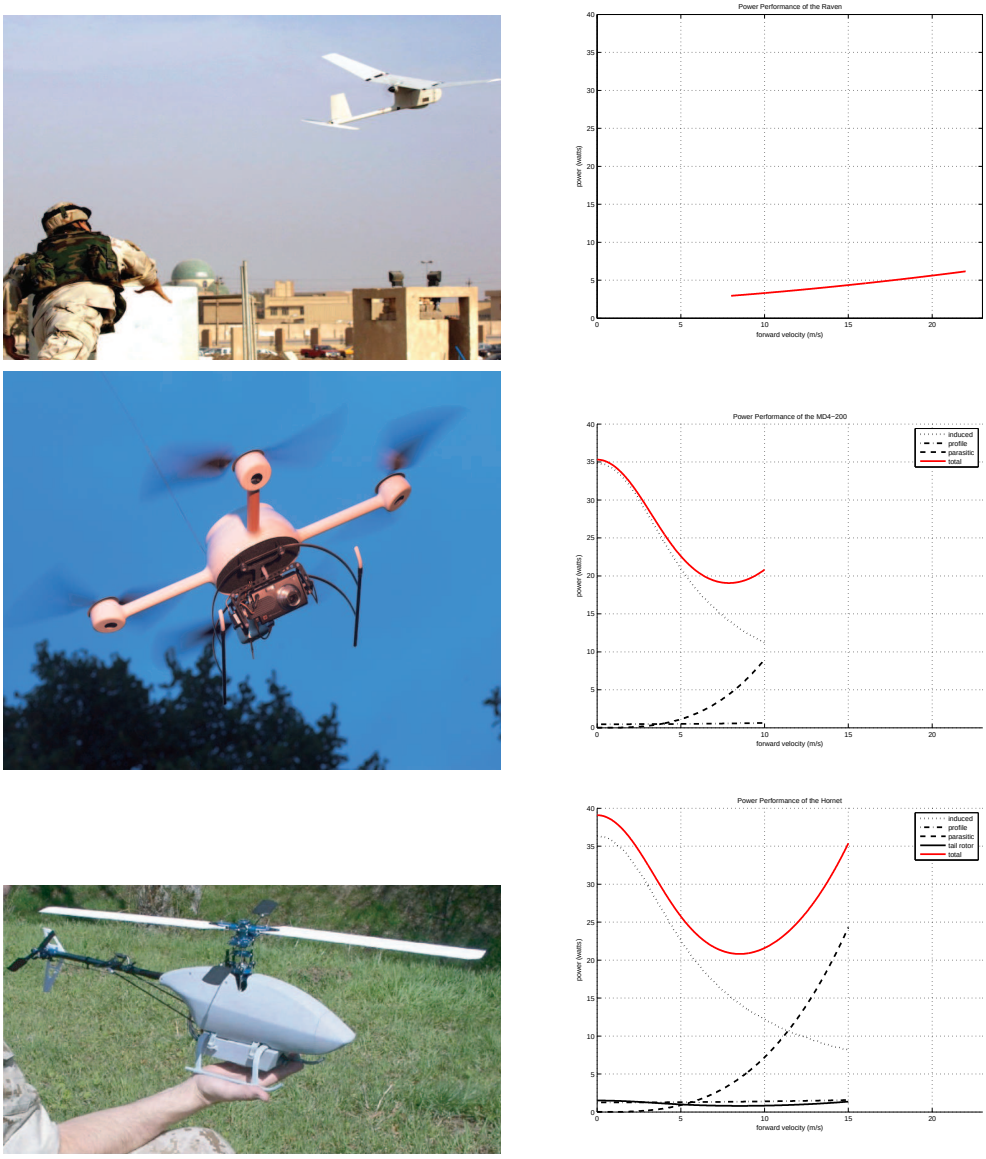


Figure 3.2: UAVs investigated (left) and their steady-state, level-flight power curves with respect to forward velocity (right). From top to bottom: the Raven UAV from AeroVironment [6], the Hornet UAV from Adaptive Flight [7] and the MD4-200 UAV from Microdrones GmbH [8] (pictures taken from gopaultech.com, microdrones.com and adaptiveflight.com, respectively).

The three UAVs also exhibit quite different power characteristics as shown in Fig. 3.2. These curves are computed from fixed and rotary wing flight mechanic principles described in [130, 131] based on the parameters in Table 3.1[2] . Power curves of the Hornet UAV and the MD4-200 UAV show the typical profiles common to rotorcrafts. The total power comes from two parts: induced power and parasitic power. Induced power is the byproduct power of lift and parasitic power is the power required to overcome the pressure and friction drags of the aircraft. While induced power decreases with increasing airspeed, parasitic power increases. For rotorcraft, the minimum power is usually achieved at a non-zero velocity. In another word, if we do not take power for vertical motion and acceleration into consideration, the most efficient flight configuration for the Hornet UAV or the MD4-200 UAV is level and steady flight at their corresponding optimal airspeeds. The power curve for Raven UAV is more straightforward. It increases monotonously with respect to airspeed. Notice that Fig. 3.2 does not include descending or climbing power and acceleration is set to zero. During our MPA construction, however, these two additional factors will be considered.

The three chosen UAVs also share some similarities. For instance, as shown in Table 3.1, they have similar weights and dimensions. They also have similar operating altitudes and endurances. Their light-weight construction and small sizes make them suitable for easy deployment and thus are ideal for missions that require high-mobility.

3.3.2 Grid-Based Flight Envelope Motion Primitive Automaton (MPA)

Based on the vehicles' flight envelopes, a quantization of their dynamics can be performed. The quantization is done by using the concept of grid-based, finite-state *motion primitive automaton* (MPA). A motion primitive corresponds to a motion[3] . This motion can of constant speed, in acceleration or in deceleration; it can be a straight line or a curve. Together, all the motion primitives provide a finite-state representation of the vehicles' guidance behavior, i.e., their closed-loop dynamics. In order to prevent from having to perform costly interpolations, these motion primitives are defined on a fixed spatial grid, which are described by the following three resolutions:

[2] Paper [117] provides details for the computation of the power curve of a standard helicopter.

[3] The concept of motion primitive is closely related to the symmetries that are inherent to vehicle's dynamics as those described in Section 5.3 and paper [4]. It is also deeply rooted in the concepts of motor primitive (as desribed in Section 1.2) and movement primitives (as described in Section 1.3).

- d_{xy}: set to the minimum turning radius for the lowest, non-zero speed;
- d_z: set to be consistent with the lowest, non-zero horizontal speed and the lowest non-zero ascent/descent speed;
- d_Ψ: set to be $\pi/4$. Such a resolution allows to take advantage of symmetry in the rotation.

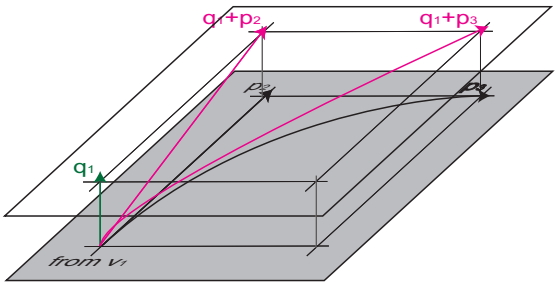


Figure 3.3: A motion primitive must be a combination of one horizontal motion primitive, such as p_2 and p_3, and one vertical motion primitive, such as q_1.

Table 3.2: Speed levels and turning directions for different motion primitives

Motion Primitive	Speed Level	Speed	Turning Direction*
Horizontal Motion Primitive	v_{h1}	0.5	$\pi/2, 0, -\pi/2$
	v_{h2}	3.5	$\pi/2, \pi/4, 0, -\pi/4, -\pi/2$
	v_{h3}	5.0	$\pi/2, \pi/4, 0, -\pi/4, -\pi/2$
	v_{h4}	6.0	$\pi/2, \pi/4, 0, -\pi/4, -\pi/2$
Vertical Motion Primitive	v_{v1}	0.0	level flight
	v_{v2}	1.0	up and down
	v_{v3}	2.0	up and down

* For horizontal motion primitives, positive turning directions mean right turns and negative turning directions mean left turns. For vertical motion primitives, there are three scenarios: level flight, descending and ascending.

As shown in Fig. 3.3, any motion primitive must be a combination of one horizontal motion primitive and one vertical motion primitive. To distinguish one from another, we will call such a combined motion primitive as a resultant motion primitive. Fig. 3.4 and Table 3.2 give an example of a motion primitive set. For horizontal motion primitives, their initial and final headings must be a multiple of $\pi/4$. Two forms of motion

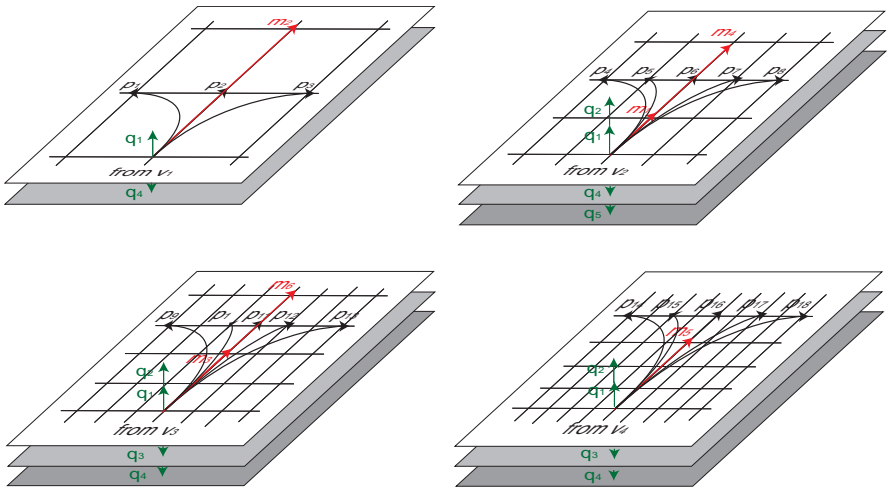


Figure 3.4: Motion primitives defined on a particular spatial grid.

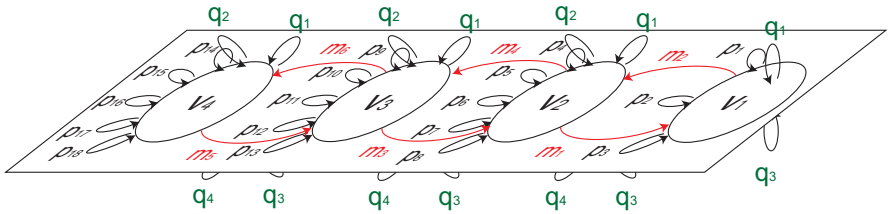


Figure 3.5: An example MPA model. If there is a arrow from state 'a' to state 'b', let's say from v_1 to v_2, it means that the UAV can make the transition from state 'a' to state 'b'. For this specified case, it means that the UAV can accelerate from v_1 to v_2.

primitives are used: rectilinear and turns. Each can be of constant speed or accelerating/decelerating. The horizontal motion primitives are described by the following attributes: translational and angular displacements (with respect to the discretized coordinates and heading) and the connectivity information. The vertical motion primitives are of constant speed and are described by the following attributes: displacements and connectivity information. A cost attribute can be assigned to each motion primitive, a horizontal one or a vertical one. It can be based on various performance criteria. Typical criteria include time and energy. Tactical criteria like visibility to a threat could also be used. Of course, a horizontal motion primitive can not be combined with any vertical motion primitive randomly. Their combination must be restricted by the flight dynamics of the UAV. For instance, the maximum vertical velocity of the Raven

UAV is a function of its horizontal velocity. An example of MPA model, after all these considerations have been accounted for, is shown in Fig. 3.5.

3.3.3 Digital Terrain Map

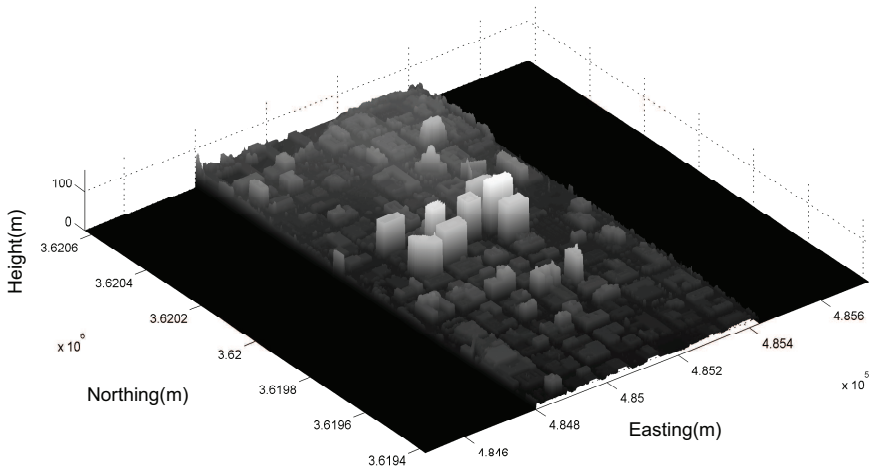


Figure 3.6: DTED map of the San Diego downtown area.

The terrain set used in this chapter and the next chapter is a 2.44 m resolution Digital Terrain Elevation Data (DTED) map covering a 0.6 km by 1.2 km section of the San Diego downtown area as shown in Fig. 3.6 [4] . The data was generated from a low altitude aircraft LIDAR scan (available from the USGS website [106]). The height map was extracted from the laser data by dividing the area covered by the data into square pixels and taking the maximum altitude sensed in each pixel. A very small number of outliers (fewer than 5) were removed by hand and replaced with 0s.

3.3.4 SCTG Computation Algorithm

This section provides a brief overview of the algorithm used to compute the SCTG function. For a given elevation map, a goal state and a MPA model, the algorithm

[4] Results with respects to other areas, such as AT&T park area of San Fransisco, can be found in our paper [117].

computes the cost-to-go value, velocities and parent cell at each cell in the map. The algorithm is based on Dijkstra shortest path algorithm. It uses a Fibonacci heap data structure for efficient retrieval and update of the cells (nodes) [132].

Suppose the entire terrain volume is represented as a graph and each free node is represented as $x_f \in F \subset \mathbb{R}^3$. Three sets of data can be obtained after the SCTG calculation: the first one $V_S(F)$ records the optimal CTG value $V_S(x_f) \in \mathbb{R}$ at each node x_f; the second one $v_S(F)$ records the optimal velocity $v_S(x_f) \in \mathbb{R}^3$ at each node x_f and finally the third one $P_S(F)$ records the parent node $P_S(x_f) \in \mathbb{R}^3$ of each node x_f. Notice that $V_S(F)$ is the traditional CTG function and the combination of $V_S(F)$ and $v_S(F)$ is what we call the SCTG map.

We formulate this problem into a graph search problem. Once the cells are initialized and the Fibonacci heap is constructed, in a loop, we fetch the unmarked cell v with the minimum CTG, label the cell v, explore its neighbors and update their CTGs. Neighbors of the v are cells from which the helicopter can reach v with one of the given motions specified by the MPA. Each cell can have at most m neighbors, where m is the size of the motion primitive set used in the MPA. The number of edges in the graph is at most mn, where n is the number of the free grid points. The complexity of the algorithm for SCTG computation corresponds to that of the Dijkstra's shortest path algorithm with Fibonacci heap on a graph of n vertices and mn edges, which is $O(mn + nln(n))$ (assuming edge weights are non-negative)[132].

Algorithm 1 Algorithm for SCTG Computation

Input: Environment dimensions S_x, S_y, S_z; terrain elevation map; motion primitives; cell dimensions D_x, D_y, D_z and goal G_x, G_y, G_z, GV_x, GV_y GV_z.

Output: CTG map of environment divided into Cells. Each Cell has properties: CTG value, 3 velocities along each coordinate axis and the parent cell.

begin
  divide the environment into cells and initialize to default values
  $N_x = \lceil S_x/D_x \rceil$ $N_y = \lceil S_y/D_y \rceil$ $N_z = \lceil S_z/D_z \rceil$ **for** $i \leftarrow 0$ **to** N_x **do**
    for $j \leftarrow 0$ **to** N_y **do**
      for $k \leftarrow 0$ **to** N_z **do**
        Cell[i][j][k].ctg $= \infty$ Cell[i][j][k].vx = Cell[i][j][k].vy = Cell[i][j][k].vz $= \infty$ Cell[i][j][k].parent = nil *based on elevation map decide whether the cell has an obstacle or free to fly* **if** *Cell[i][j][k] lies in an obstacle* **then**
          Cell[i][j][k].free = $false$ Cell[i][j][k].marked = $true$;
        else
          Cell[i][j][k].free = $true$ Cell[i][j][k].marked = $false$;
        end
      end
    end
  end
  initialize the cell which has the goal Set the goal cell velocties (vx, vy, vz) to goal velocities (GV_x, GV_y GV_z), and ctg to 0.0
  construct a fibonacci heap of unmarked and free cells, ctg value is the key $H \leftarrow$ Construct-Fibonacci-Heap(Cell)
  while *there are unmarked cells* **do**
    $v \leftarrow$ Delete-MinCTG-Cell(H) v.marked = $true$;
    the neighbors of v are determined through the motion primitives **while** *Each neighbor w of v* **do**
      if *$w.ctg > v.ctg$ + cost(v, w)* **then**
        Decrease-Key(H, w, w.ctg - (v.ctg + cost(v, w))) Compute the w.vx, w.vy and w.vz based on motion primitives w.parent = v
      end
    end
  end
  Destroy-Fibonacci-Heap(H)
end

3.4 Evaluation of MPA Framework

In this section, we use the solutions obtained by using nonlinear programming (NLP) as baselines to evaluate the performance of our MPA framework. We introduce the procedures of applying NLP to compute optimal solutions for some simple cases. Then, by comparing the results obtained by NLP with the results obtained by MPA, we are able to evaluate the performance of our MPA framework as well as to analyze the effect of quantization on the performance.

3.4.1 Baseline Solutions by Nonlinear Programming

For simple problems, it is possible to generate baseline solutions using a traditional numerical trajectory optimization method. The numerical trajectory optimization technique is based on explicit solutions to the two-point boundary value problem defined by the start and goal states. This method is one of the most accurate methods available, and is suitable for solving the optimization problems defined by guidance tasks over simple environment geometries (i.e. obstacles) and boundary conditions (initial and terminal state). The trajectory for a given spatial configuration, vehicle dynamics, and set of boundary conditions is computed using a nonlinear programming method (NLP) similar to that described in [133].

Nonlinear programming uses a finite difference method for simulating the vehicle dynamics in discrete time. In this method, trapezoidal discretization is used to approximate the dynamics of the vehicle. Taking horizontal motion in the x direction, for example, we have:

$$\begin{aligned} \frac{v_x[k+1]-v_x[k]}{2} &= \frac{N(x[k+1]-x[k])}{T}, \\ \frac{a_x[k+1]-a_x[k]}{2} &= \frac{N(v_x[k+1]-v_x[k])}{T}, \qquad k=0,...,N-1. \end{aligned} \tag{3.6}$$

N is the number of time intervals and T is the time duration, which is the quantity that needs to be minimized. With NLP, the constraints on the velocity and acceleration (3.6) are explicitly enforced. Since the T obtained from numerical method can be negative,

a slack variable zT is employed to ensure that we are minimizing a positive value:

$$\begin{aligned} T &\leq zT \\ -T &\leq zT \end{aligned} \tag{3.7}$$

Finally, the performance criterion that is minimized is represented as:

$$\begin{aligned} J =& zT + \epsilon \sum_{i=2}^{N-2} [(u_x[i+1] + u_x[i-1] - 2u_x[i])^2 + (u_y[i+1] + u_y[i-1] - 2u_y[i])^2 \\ &+ (u_z[i+1] + u_z[i-1] - 2u_z[i])^2](\frac{N}{T})^3 \end{aligned} \tag{3.8}$$

The purpose of the ϵ terms is to eliminate "ringing" in the control variables. The algorithm used to solve this nonlinear programming problem is called the interior point method [134].

3.4.2 Performance Evaluation of MPA Method

The concept of the MPA stems from the observation that general trajectories can be broken down into two types of segments: equilibria or coasting, and non-equilibria or maneuvering [78]. Therefore, the MPA must be sufficiently expressive to capture these two behaviors. The main design parameter in a MPA is the number of uniformly sampled velocity levels, i.e., the granularity of velocities discretization between the saturation levels. An appropriate set of motion primitives should capture the optimal performance of the continuous dynamic but should also be small enough to avoid excessive computation. Even though in benchmarking, the CTG is computed offline, it is important to understand the level of accuracy that is sufficient for this application. The accuracy of the MPA can be established based on the predicted CTG and velocity maps. The optimal value function calculated using NLP can be used to determine the optimality gap and vector field mismatch and determine the set of motion primitives that is accurate enough without being too large to use. For this purpose, the optimal solutions for an obstacle-free environment are generated for a number of points, as shown in Fig. 3.7. The optimal value function can be compared with the CTG maps resulting from different parameter choices using the mean squared error (MSE) criterion. The set of parameter choices that results in a CTG that most closely matches the optimal CTG is selected.

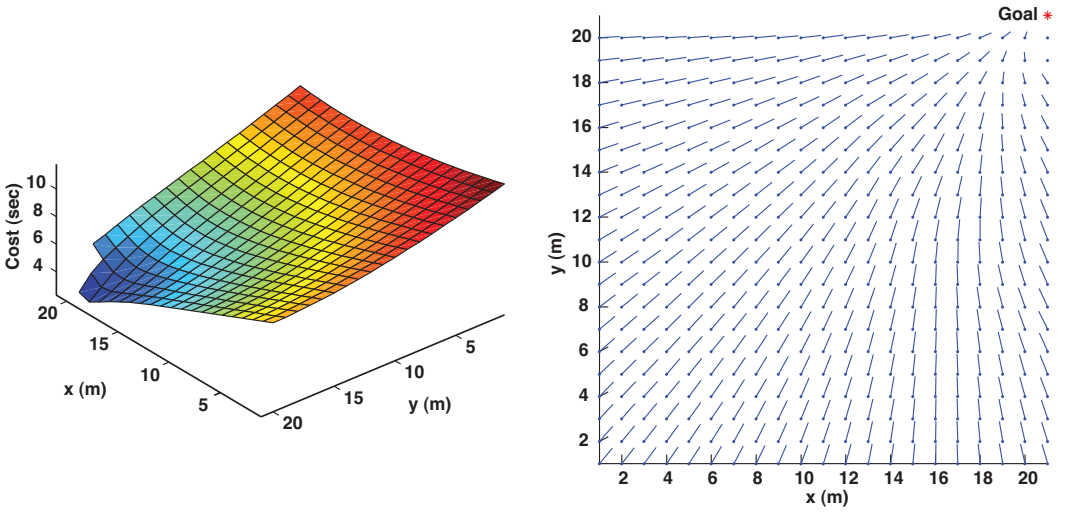


Figure 3.7: Optimal value function (left) and its corresponding vector field (right).

Fig. 3.8 shows how the mean relative cost difference between optimal cost and MPA cost changes with respect to different uniformly sampled velocity levels. It also illustrates that by including motion primitives that result from saturated accelerations, the overall MPA performance can be significantly improved. This is because saturated behaviors, or behaviors at the boundary of dynamic envelope, are the essential ingredients of time-optimal behavior. The most obvious case is the classical bang-bang control strategy.

As shown in Fig. 3.8, the lowest value of mean relative difference is obtained with 12 uniformly sampled velocities both for motion primitives with and without saturated behaviors. These settings result in a total number of 5615 motion primitives. Comparisons with respect to the optimal value function and optimal velocity vector field are shown in Fig. 3.9. The peak relative cost difference is 20.6% while the average relative difference is 4.2%. Larger relative differences are concentrated in the area surrounding the goal. This is primarily due to the coarse heading sampling within the state space quantization. Overall the accuracy of the MPA model is close to the continuous model.

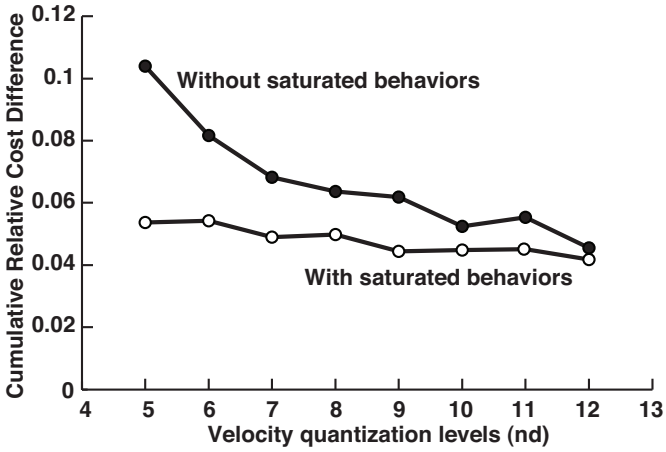


Figure 3.8: Averaged performance of MPA relative to the optimal value function for different levels of velocity quantization. Notice how saturated primitives improve the optimality across all levels.

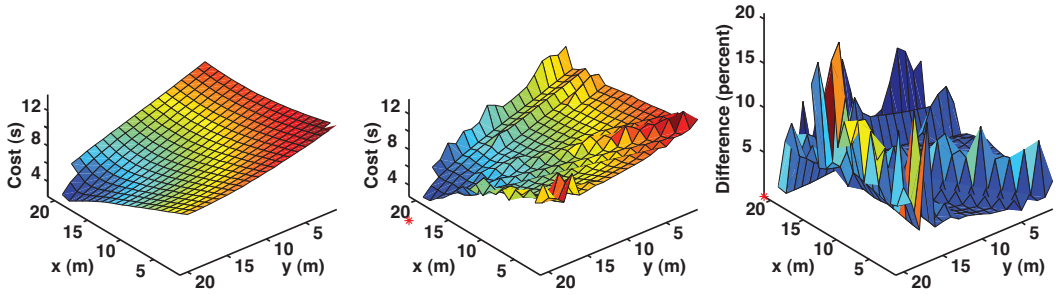


Figure 3.9: Validation of the MPA in free space: Optimal cost-to-go using the NLP solutions (left), and MPA (center), and the percent cost difference (right).

Chapter 4

Analysis of Optimal Guidance Behavior Based on Optimal Trajectories and SCTG Maps

4.1 Introduction

In this chapter, the optimal guidance behaviors generated for different UAV types and performance criteria by using the MPA framework described in the previous chapter are used to study the roles of individual factors, such as agent dynamics and performance criteria, on the guidance behavior and performance. The analysis is performed based on two different methods: individual trajectories, and spatial cost-to-go (SCTG) maps. The essential difference between the two methods is that the first one can only get insights about the influence of individual factors from a single run or multiple runs, while the second one can provide a global view of the guidance behavior as affected by the individual factors. If we take the analysis of guidance behavior as surveying an area, the first method corresponds to driving around and collecting data along the way and the second method corresponds to flying above the area with a helicopter and taking photos. One conclusion we can get from the guidance behaviors is that there are structures embedded in them, which result from the interactions between the agent and the environment. These structures will be further generalized and formally defined in

the next chapter.

The material presented in this chapter is first published in paper [110].

4.2 Analysis Based on Optimal Trajectories

4.2.1 Generating Optimal Trajectories from SCTG

The set $P_S(F)$ is stored in a tree data structure. The root of this tree is the goal and each branch represents a parent-child relationship between two nodes. In other words, if $P_S(x)$ is considered as a map $P_S(x_f) : \mathbb{R}^3 \mapsto \mathbb{R}^3$ that takes a node to its parent node, for any node x_0 in the free space, a route can be constructed by applying map P_S iteratively as $P_S^m(x_0)$ until it reaches the goal[1] .

Table 4.1: Time performance of different vehicles obtained with different performance criteria (unit: second)

vehicle	Hornet		MD4-200		Raven	
performance criterion	time	energy	time	energy	time	energy
A1	31.38	47.33	35.99	70.86	23.76	44.68
A2	21.27	26.60	25.10	51.83	18.84	50.86
A3	23.48	93.01	26.65	96.39	16.36	53.66
A4	18.73	41.22	23.41	51.97	11.26	28.78
A5	22.35	43.33	27.90	49.45	14.76	26.31
A6	22.09	47.59	26.04	70.03	21.32	40.05
mean	23.22	49.84	27.52	65.09	17.72	40.72
relative difference (%)	114.68		174.65		129.87	

The optimal trajectories can behave quite differently depending on the performance criteria used in the computation. These differences can be appreciated by inspecting their trajectories, associated trajectory characteristics and performances as measured by some chosen criteria. Fig. 4.1, Fig. 4.2 and Fig. 4.3 show the optimal trajectories for the three UAVs for the time and energy performance criteria. The figures also show the time histories of state variables and their corresponding histograms for one of the starting points. Their time and energy performance scores are given in Table 4.1 and

[1] Please refer to Section 3.3.4 for the meanings of $P_S(F)$, $P_S(x_f)$ and others.

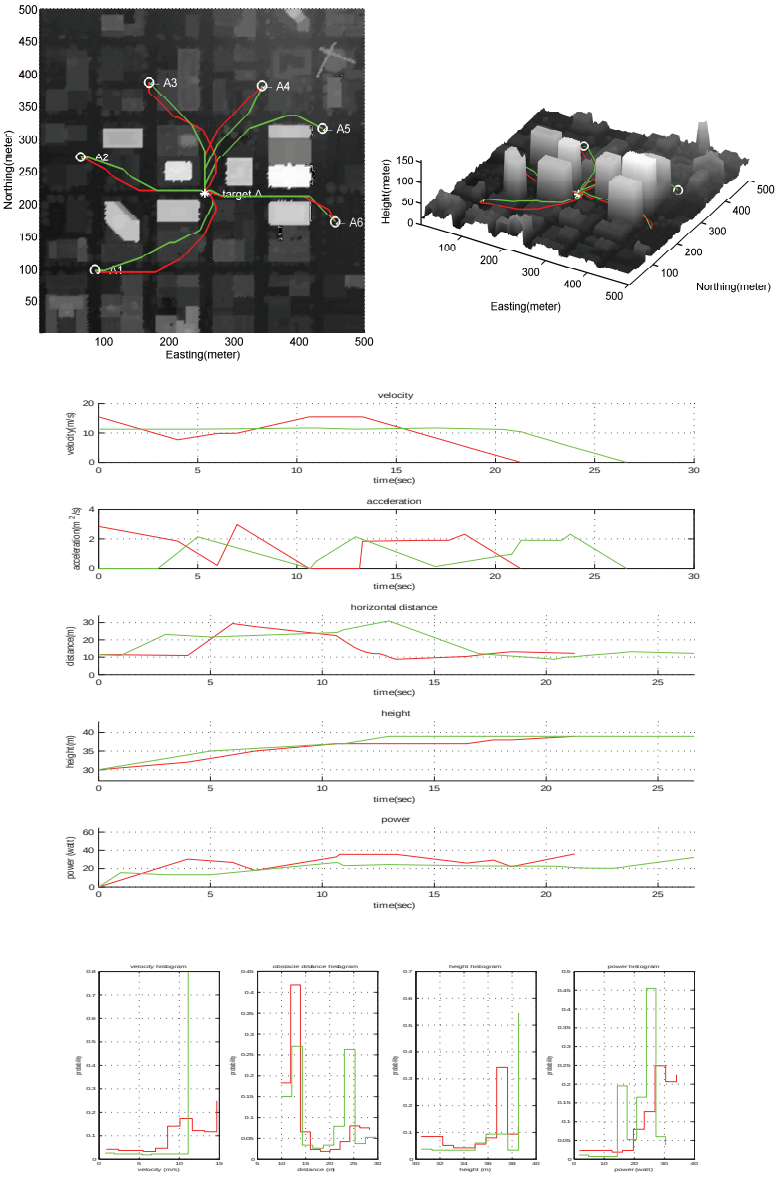


Figure 4.1: Optimal trajectories of the Hornet UAV (a standard helicopter) from six different points (A1 to A6). Time histories of a number of key variables (speed, acceleration, distance to the nearest building, height and power, from top to bottom) and their histograms (speed, distance to the nearest building, height and power, from left to right) for starting point A2 are shown on the second and third row. In this figure, Figure 4.2 and Figure 4.3, minimum-time trajectories, their corresponding time histories and histograms are illustrated as red and minimum-energy ones are illustrated as green.

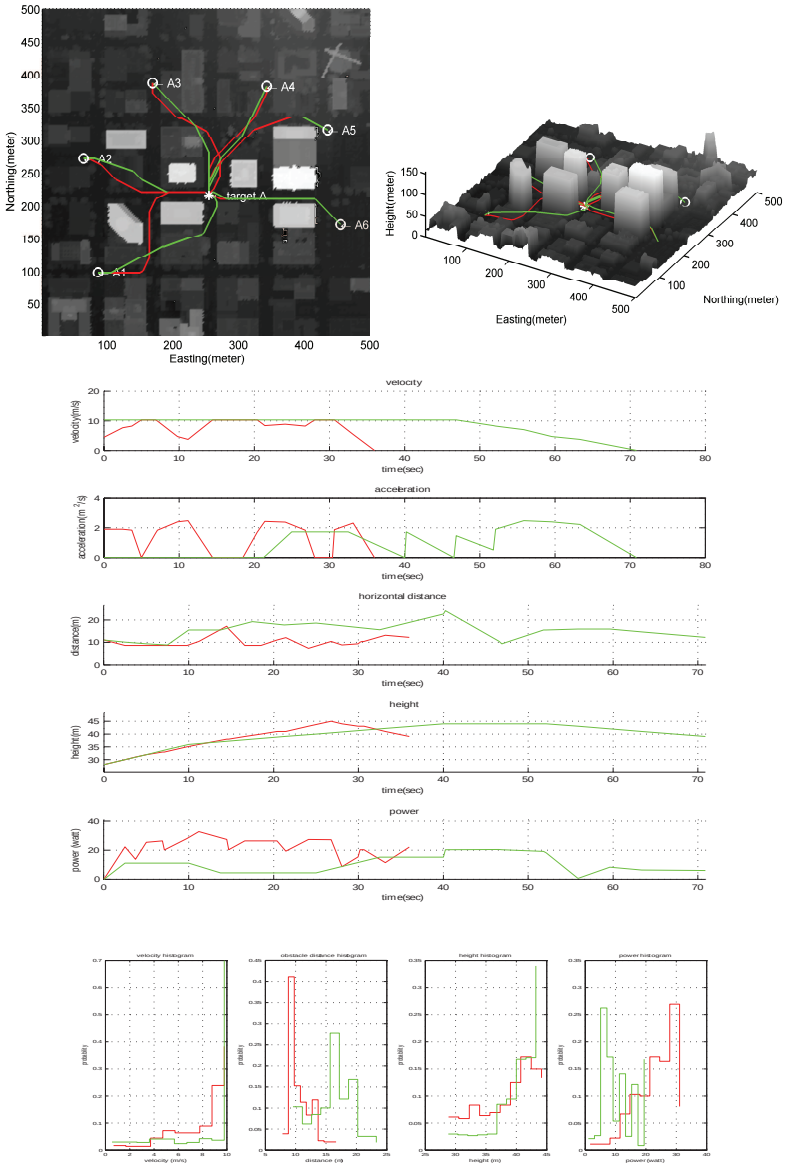


Figure 4.2: Optimal trajectories and their statistical characteristics for the MD4-200 UAV (a quadrotor).

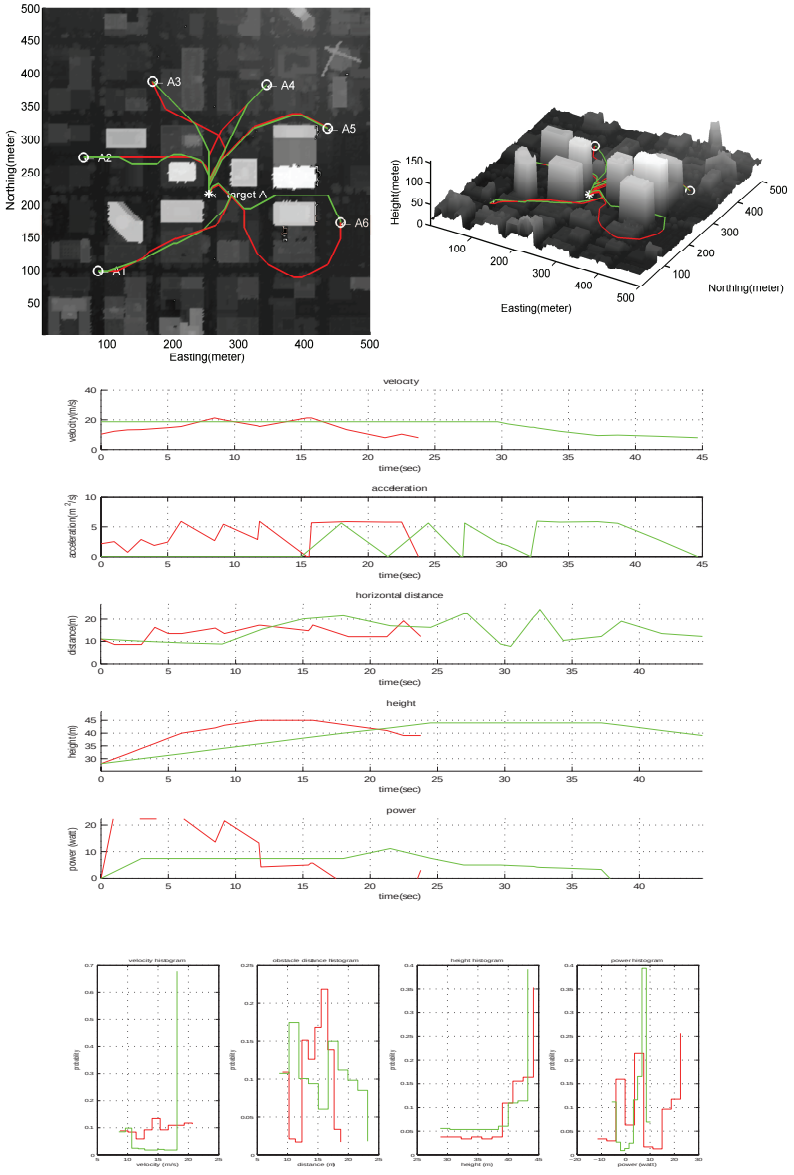


Figure 4.3: Optimal trajectories and their statistical characteristics for the Raven UAV (a fixed-wing aircraft).

Table 4.2: Energy performance of different vehicles obtained with different performance criteria (unit: J)

vehicle	Hornet		MD4-200		Raven	
performance criterion	time	energy	time	energy	time	energy
A1	916.53	772.22	835.60	747.90	249.66	240.04
A2	645.97	610.39	610.80	598.09	322.26	270.06
A3	573.48	473.15	515.79	467.54	322.26	270.06
A4	359.78	328.59	326.45	316.16	111.20	110.86
A5	529.31	488.35	499.34	476.52	62.93	43.80
A6	658.79	632.34	618.31	598.34	196.81	193.08
mean	613.98	550.84	567.72	534.09	188.67	160.15
relative difference (%)	11.46		6.3		17.8	

Table 4.2. In these two tables, different rows correspond to different starting point and different columns correspond to different vehicle and different performance criteria. Take Table 4.1 (which is based on the minimum-time trajectories) for example, if we want to find the MD4-200 UAV's performance score from point A4, we first go to the columns for the MD4-200 UAV, and then go to the column for time, and finally go to the row for A4, which is 23.4 seconds. These results provide the basis to investigate the influences of aircraft types and performance criteria on guidance performance.

4.2.2 Influence of Aircraft Types on Trajectories

While all three UAVs can reach the goal from the six starting locations, their trajectories can be very different depending on their dynamics. These difference illustrate how the flight-dynamic capabilities and energy profiles play out in the urban terrain environments.

The minimum-time trajectories of the Hornet UAV and the MD4-200 UAV have similar trajectory characteristics. They both stay 10 meters away from the nearest building 40% of the time (as shown in their distance histograms). Note that 10 meter is the safety distance that was specified and acts as a hard constraint for the minimum distance between the aircraft and the buildings. So in another word, the Hornet UAV

and the MD4-200 UAV stay as close to the buildings as possible to achieve minimum-time performance. A typical trajectory is the one achieved by the MD4-200 UAV from point A1: it essentially follows the streets, which are surrounded by tall buildings on both sides. As for the Raven UAV, due to its inability to perform sharp turns that would allow it to fly around buildings, its minimum-time trajectories stay further away from the buildings. A typical trajectory for this aircraft is the one from point A6: it can be divided into three phases - a right turn phase to steer around buildings, a coast flight phase and finally a left turn phase with minimum turning radius to reach the goal.

The minimum-energy trajectories behave quite similarly in the vertical direction for all three UAVs: they first climb until they clear buildings, then coast at a certain height and finally they descend. For the Hornet UAV and the MD4-200 UAV, in the final descending phase, they still need power; while for the Raven UAV, as shown from its power history in Fig. 4.3, in the last few seconds, there is no power expense. So in that final stage, it glides and uses the extra potential energy.

4.2.3 Influence of Performance Criteria on Trajectories

Individual differences can also be seen by inspecting their time and energy performances. For instance, for all the six starting points, as shown in Table 4.1, the Raven UAV outperforms the Hornet UAV, and the Hornet UAV outperforms the MD4-200 UAV. The ranking happens to be consistent with their maximum speeds. However, the relative performance differences achieved are quite smaller than the relative maximum speed differences. For example, the mean time performance is 23.22 second for the Hornet UAV and 27.52 second for the MD4-200 UAV, which corresponds to a difference of about 18%. The respective maximum speeds are 15.27 m/s and 10 m/s, corresponding to a relative maximum speed difference of about 53%. The fact that maximum speed does not translate entirely into a gain in performance highlights the importance of the interaction between the dynamic capabilities and the environment. As shown from their speed histories in Fig. 4.1 and Fig. 4.2, neither of these two UAVs is able to fly at maximum speed all the time.

We can also look at the effect that the different performance criteria have on behavior. For all three UAVs, as can be expected, their minimum-time trajectories have a general tendency to follow high speeds. In contrast, their minimum-energy trajectories

have a tendency to stay near a certain speed (11 m/s for the Hornet UAV, 10 m/s for the MD4-200 UAV and 18 m/s for the Raven UAV). Notice that these speeds are different from their respective minimum-power speeds. To achieve minimum-energy, a UAV has to find the right tradeoff between power and flight duration. Take the Raven UAV for example, while flying with its minimum-power speed (8.89 m/s) can save energy over a given flight duration, it also means that it has to spend more time to cover the same distance than it were flying at a higher speed. So after taking both of these factors into consideration, it settles to 18 m/s, which is neither the minimum-power speed nor the maximum flight speed.

Following the comparison based on UAV types, we can also gather interesting insights by comparing trajectories based on the performance criteria used to generate those. In Table 4.1 and Table 4.2, values that are underlined correspond to the minimum values obtained when that particular performance criterion was used in the optimization. The other values provide an indication of the range of values that can be attained and the particular influence of a performance criterion. As seen from the last row of Table 4.1, the average time performances of minimum-energy trajectories for all the three UAVs are twice of those of minimum-time trajectories. The differences are not that dramatic for energy performance, as shown in the last row of Table 4.2.

4.3 Analysis Based on SCTG Maps

In contrast to point-to-point trajectories, the SCTG maps, $V_S(F)$ and $v_S(F)$, can provide a global understanding of the way how the dynamic capabilities, the goal condition, the environment and performance criteria interact and give rise to the spatial distribution describing the optimal vehicle behavior. For instance, the vector fields $v_S(F)$ for the MD4-200 UAV and the Raven UAV are illustrated in Fig. 4.4. In the sub-figures, the vector at each free grid point x_f indicates the magnitude and direction of the optimal velocity $v_S(x_f)$.

To illustrate the idea behind vector field map $v_S(F)$, suppose at each location in the free environment, there is a particle. As time evolves, these particles follow the velocity indicated by the vector field map as shown in Fig. 4.5. Such a process is analogous to the concept of pathline in fluid mechanics [135]. Vector field map provides a snapshot

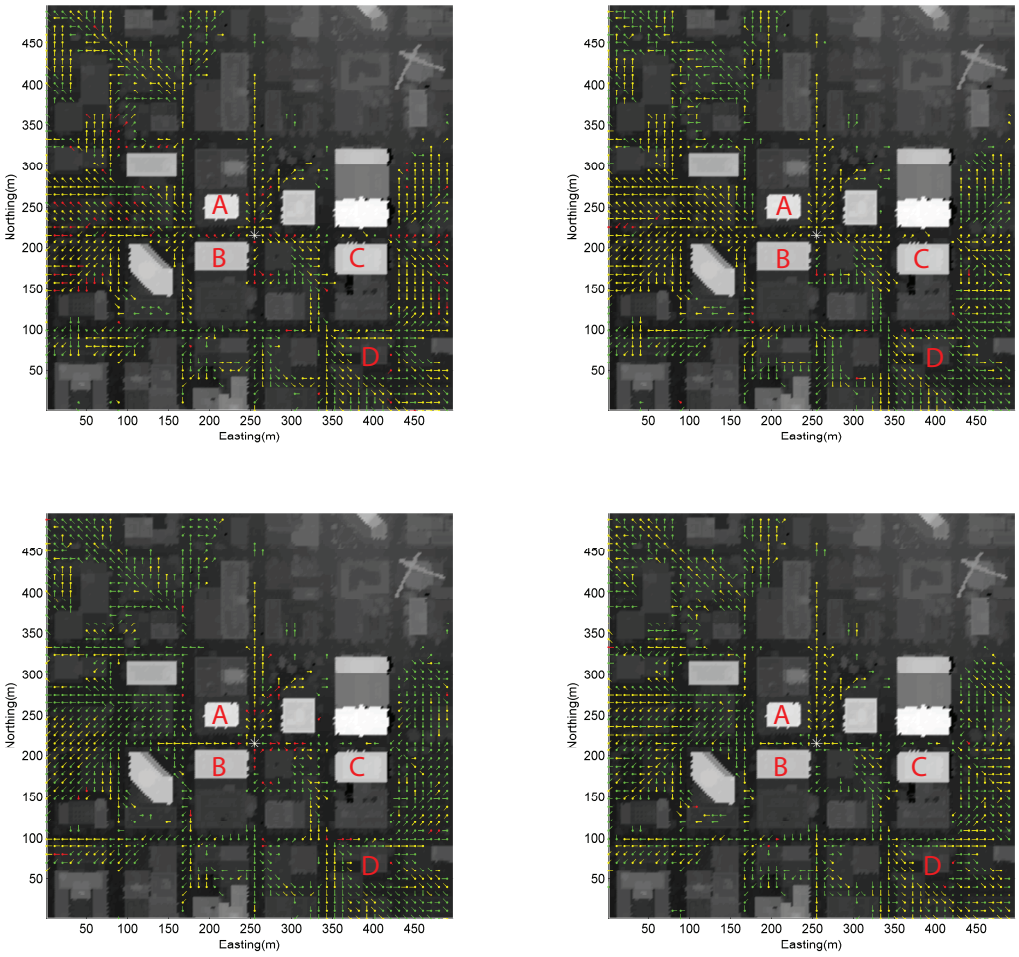


Figure 4.4: Vector fields for different UAVs and different performance criteria. First row corresponds to the MD4-200 UAV and second row corresponds to the Raven UAV. First column corresponds to minimum-time performance criterion and second column corresponds to minimum-energy performance criterion. Since the behaviors of the Hornet UAV and the MD4-200 UAV are similar, only the vector fields of the MD4-200 UAV are illustrated. In all these sub-figures, different colors mean different vertical velocity directions: yellow indicates level flight, red indicates descending and green indicates climbing. The goal is at the center of each figure and it is represented as a white star.

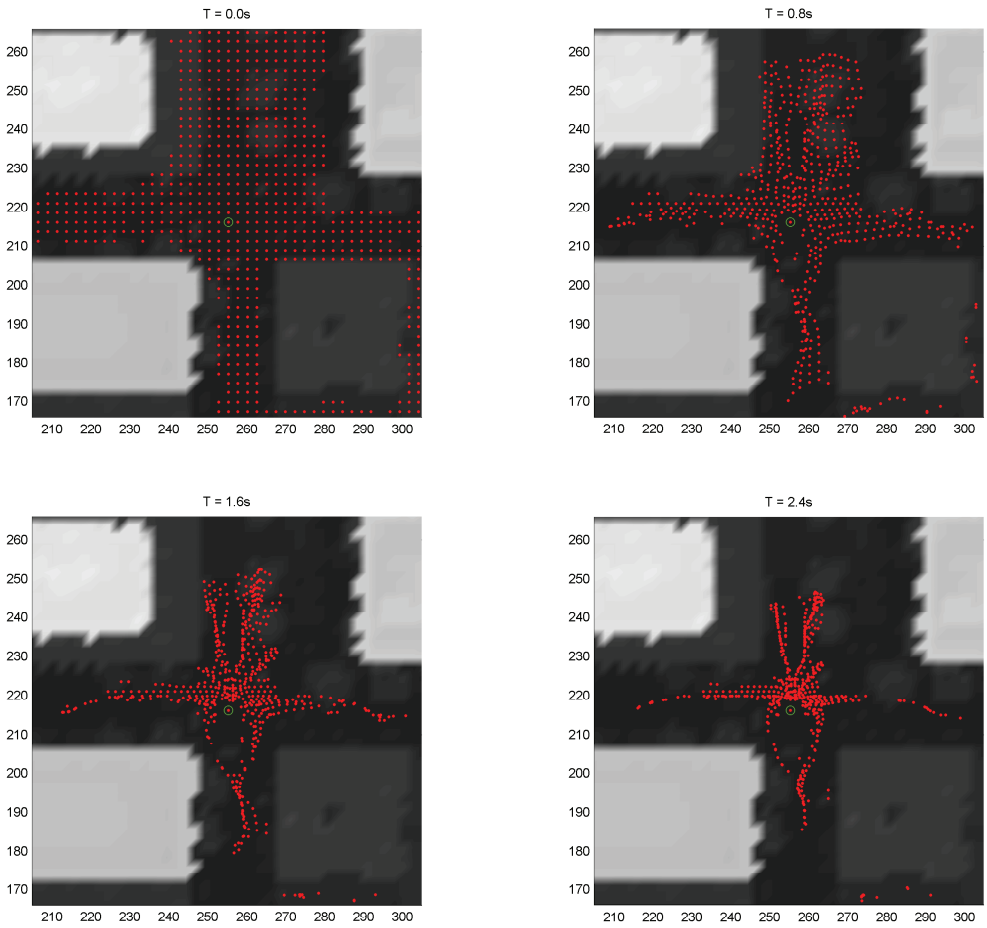


Figure 4.5: Explanation of the vector field map $v_S(F)$. Suppose a particle is put at each free location as shown in the upper-left sub-figure. The locations of these particles at time 0.8, 1.6 and 2.4 seconds are illustrated in the other sub-figures. The case here corresponds to the MD4-200 UAV with minimum time criterion. The corresponding vector field is in Fig. 4.4.

of the optimal behaviors, while figures such as the ones in Fig. 4.5 illustrate the full time history. Of course, here since the vector field is fixed, or steady in terms of fluid mechanics, the vector field in Fig. 4.4 and the particle locations in Fig. 4.5 convey equivalent information.

4.3.1 Characteristics of $V_S(F)$ and $v_S(F)$ Distributions

The map $v_S(F)$ can help to detect phenomena in a particular UAV behavior that may not be easily seen by investigating single trajectories. For example, for the area west of building 'A' (in Fig. 4.4), the general minimum-time behavior for the MD4-200 UAV is to move toward the street between building 'A' and building 'B'. For the Raven UAV, on the other hand, the general behavior is to move to the northeast and steer around building 'A'. The reason for the different patterns is that considering that the final heading is to the south, while the MD4-200 UAV can make a turn after it passes the street between 'A' and 'B', the Raven UAV can not. This explanation is also valid for what happens at the southwest corner of building 'B' for the Raven UAV. Instead of taking a more intuitive route, which is to fly around building 'B' clockwise, the Raven UAV has to take a longer route, which is to fly clockwise with a larger circle and around building 'A'.

The map $v_S(F)$ can also help better understand the effect of different performance criteria. The difference between minimum-time and minimum-energy behaviors for a same UAV, however, is not as dramatic as the difference seen between different UAVs with the same performance criterion, at least at the altitude shown in Fig. 4.4, which corresponds to the altitude of the goal. For many grid points x_f, the horizontal directions of velocity $v_S(x_f)$ for minimum-time and minimum-energy performance criteria are the same in this particular terrain. However, there exist many other locations where the horizontal directions are quite different, for instance, the southeast corner of building 'C'. Moreover, vertical behavior can be quite different. For the minimum-energy behavior, the UAV tends to stay at the same height to save energy. This requires climbing first to reach an altitude that allows clearing buildings. The stored potential energy can then be used. While for minimum-time behavior, the UAV has more freedom to move up and downs. Such behavior is clearly shown in the upper-left sub-figure.

4.3.2 Embedded Structure in SCTG: the Virtual Arteries

The velocity distribution $v_S(F)$ is relatively uniform within certain areas, such as the west of building 'A'. In another word, trajectories starting from this area share a similar route. In contrast, in some areas, $v_S(x_p)$ can be different even for two adjacent points. This is the case in the area at the southeast corner of building 'D'. As will be elaborated on in the next few chapters, such a phenomenon can be understood from a dynamical system standpoint. For instance, for the optimal behavior of Dubins' vehicle in 2D space (as described in Appendix A), the free space embeds a partitioning structure defined by separatrices and sub-goals. For all the starting points within the same partition, their trajectories are *equivalent* in the sense that they can be continuously deformed into each other without colliding with any obstacle[2] . Furthermore, their control laws are the same, which is basically to move away from repelling manifolds and to approach the sub-goal corresponding to the partition that they belong to. After passing the first sub-goal, any trajectory starting from a same partition will follow the same 1D path. The collection of all these 1D paths forms a artery-like structure, with its root at the goal and the tips at the sub-goals. Here, we call such a structure *virtual artery*. The virtual artery is embedded in the free space and reflects the salient structures within the guidance behavior, just as the role of the hierarchical generalized Voronoi graph (HGVG) in sensor-based exploration [136]. Only here the structure is a by-product of the interaction of the dynamics and the environment.

The concept of virtual artery can be extended to the 3D case. For instance, in Fig. 4.6, the minimum-time trajectories for a number of starting points behind building 'D' are plotted. Now based on the trajectories' behavior, the area behind the building can be divided into different partitions by 2D separatrices. And the following statements are still true: for all the starting points in a same partition, their optimal trajectories are equivalent and their local control laws are the same. The difference is that, for 3D cases, the trajectories leave a partition from a set of points instead of the single point in 2D cases; and after they pass the point set, they follow a tube of paths instead of the

[2] This equivalence will be called $\sim_s$ equivalence or equivalence with respect to subgoals in Chapter 5. One message we want to convey here is that these virtual arteries, once defined, can greatly reduce the complexity involved in the description of the UAV's guidance behavior. Such a feature, if used appropriately, would be potentially beneficial for the design of autonomous systems as well as human-machine interfaces.

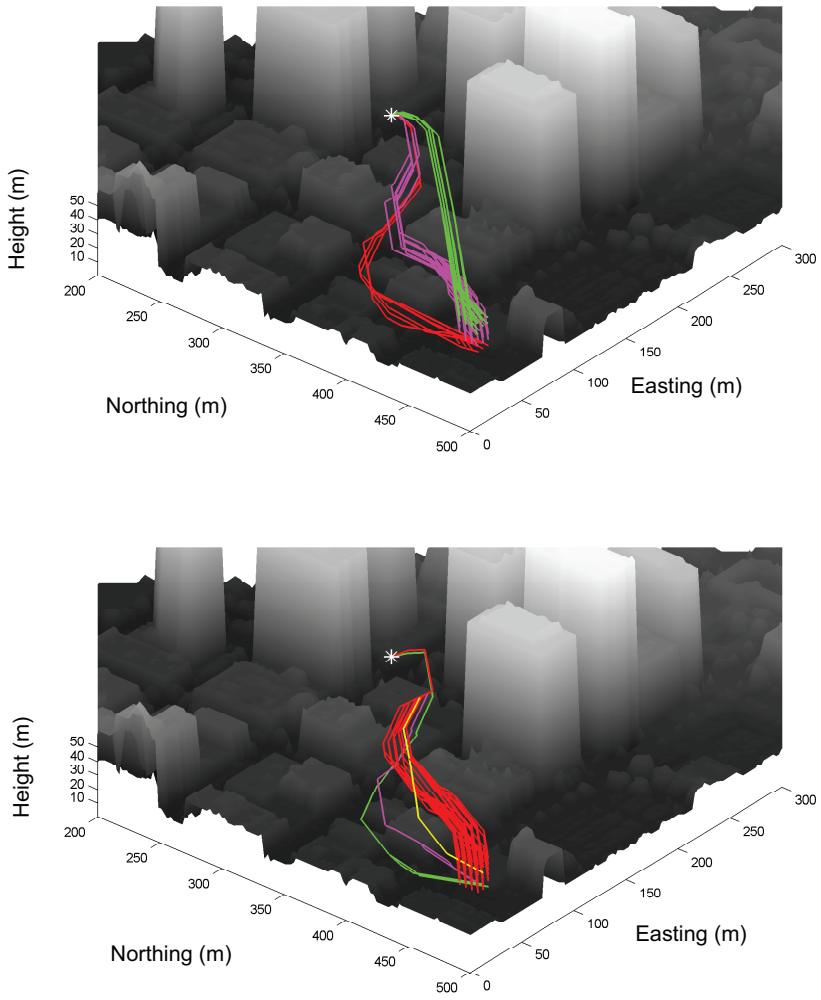


Figure 4.6: 'Virtual Artery' concept. These two figures illustrate the minimum-time trajectories from a number of points located at the southeast corner of building 'D'. The upper figure corresponds to the MD4-200 UAV and the lower one corresponds to the Raven UAV. Trajectories with the same color follow similar route. There are three different routes for MD4-200 UAV and four for Raven UAV.

single 1D path in 2D cases.

4.4 Conclusions

Guidance behavior results from the interplay of many factors, which include aspects related to flight-dynamic capabilities, UAV systems, guidance algorithm, performance criteria and the environments. In the previous chapter and this chapter, we described a framework to generate optimal guidance behavior for different UAV types and then to evaluate the important effects aircraft type and performance criteria have on guidance behavior.

The proposed framework formulates guidance problem as an optimal control problem. To enable computation of solutions over an entire geographical environment, the vehicle dynamics are approximated using a motion primitive automaton (MPA). This formulation makes it easy to compute the spatial distribution of optimal state and cost – the spatial cost-to-go (SCTG) maps – for any vehicle modeled as MPA, environment given as digital elevation map, and a pre-specified performance criterion.

4.4.1 Summary of Results

The results presented in the previous chapter and this chapter focused on three miniature UAV types: a fixed-wing aircraft, a standard helicopter and a quad-rotorcraft. Both time and energy performance criteria were investigated. SCTG map, computed for a section of the city of San Diego, provided a basis to analyze and compare the vehicle behavior. First, we compared individual differences in their trajectories and corresponding time histories for a number of starting locations in the environment. Second, we also analyzed the overall characteristics of their SCTG maps. The conclusions from these comparisons helped better understand the influences of UAV's dynamic capabilities and choice of performance criteria on UAV's overall guidance behavior. For instance, we were able to show that the Hornet UAV and the MD4-200 UAV share quite similar trajectory characteristics and overall spatial distribution of optimal behavior for the terrain investigated in this paper, while the Raven UAV behaves quite differently.

4.4.2 Situated Perspective

The SCTG maps offer a way to understand effects arising from the interaction between vehicle dynamics and the environment. How and to what degree a UAV can fulfill a guidance task depends on its embodiment, or what we could call dynamic 'fit', within its environment and how well the dynamic fit can take advantage of it to accomplish a task. For instance, the similarity in behaviors observed between the Hornet UAV and the MD4-200 UAV, do not simply arise because they have similar airframe configurations – actually they have quite different ones – but their similar ability of attaining their functionality within a specific urban environment. If these two platforms were used in another type of environment, this interaction may play out differently, i.e., they would have a different dynamic fit. We would then expect to see different guidance behaviors and thus different functional abilities. This example illustrates that, when speaking of guidance behavior, it is not sufficient to look at the vehicle alone; its operational environment and the dynamic interaction between the two also need to be taken into consideration[3] . This type of analysis is known as 'situated perspective' and has received significant attention in the robotics community [137].

4.4.3 Structures Existing in Guidance Behavior

Virtual artery illustrates that guidance behavior, when viewed as the results of agent-environment interactions, exhibit certain type of structures. If these structures could be appropriately characterized, first of all, they would provide a way of describing the guidance behavior with a much more compact representation; second, the existence of such a representation would mean the possibility of utilizing it in the design of autonomous guidance systems (since it provides a way of mitigating the representational or even computational complexities); last but not least, these structure, which are essentially inherent to the agent's interactions with the environment, can be used to the study of human guidance behavior. The last point will be elaborated on in the next few chapters. But the takeaway point here is that, whether humans are implementing some optimality principles or not, when organizing their guidance behavior, they face the same problems

[3] In Chapter 2, we spoke of guidance behavior as closed-loop dynamics resulting from the agent's interaction with the environment. The conclusions we get in this chapter further confirm the importance of taking a situated and embedded view when investigating guidance behavior.

as those of autonomous agents, e.g., the integration of different modules, the curse of dimensionality and others as described in Section 1.2. So if similar structures can be found in humans' guidance behaviors, it would be computationally efficient for them to utilize the structures to help organize their guidance behaviors, too.

Chapter 5

Interaction Patterns in Agent-Environment Dynamics

5.1 Introduction

In the previous chapter, we showed that certain structures emerge from the optimal guidance behavior of UAVs. These structures can be described as virtual arteries. And they can be defined with respect to the concept of equivalence and are essentially inherent to the guidance behavior. With the help of the structures, the description of the UAVs' guidance behavior can be largely reduced. This feature means that these structures could potentially be used to the understanding of human guidance behavior as well as the design of autonomous systems.

In this chapter, we conceptualize these structures and also generalize the concepts in such a way that they can be used to the study of both optimal and natural guidance behaviors. We first use the guidance behavior of Dubins' vehicle as an example to introduce key concepts and lay out the main research questions we will address in this chapter and the next two chapters. Then we identify two types of equivalences or symmetries that are inherent to guidance behavior. These equivalences provide the basic structural elements to formalize guidance behavior, and at the same time, are general enough to create a framework that applies both to natural and artificial agents. We then introduce the concept of *interaction pattern*, which is the core concept of the remaining chapters.

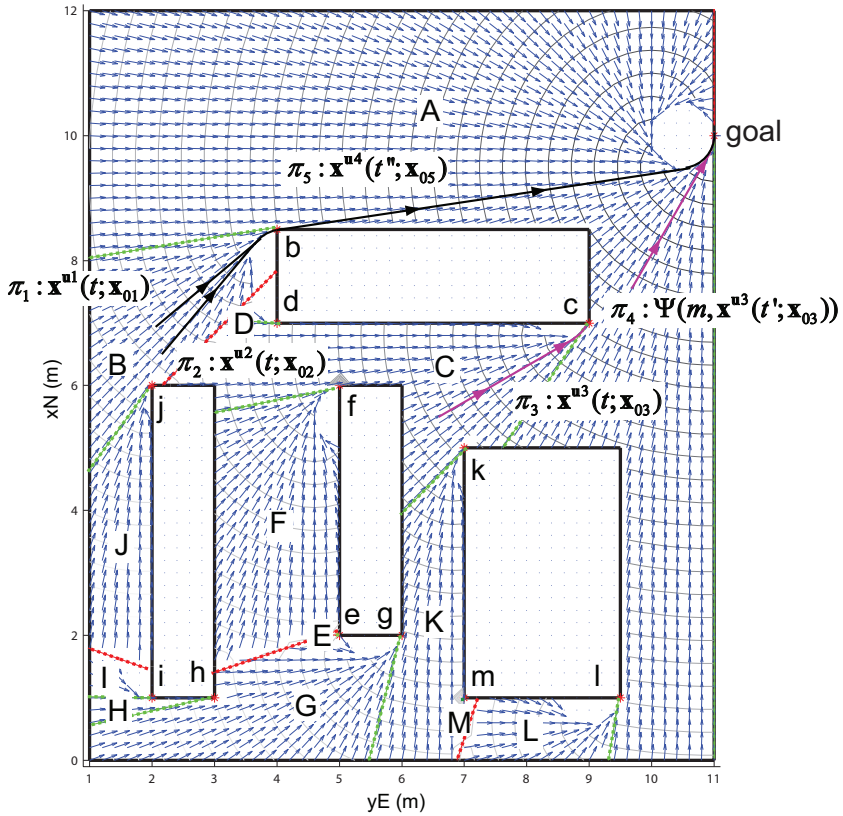


Figure 5.1: Time-optimal guidance behavior of Dubins' vehicle in an environment with multiple obstacles. Vectors represent the optimal velocity at each position, while contours represent the optimal value function. The velocity at the goal is constrained to be northbound (upward).

The material presented in this chapter is first published in papers [138, 113, 114].

5.2 A Toy Example: Guidance Behavior of Dubins' Vehicle

Before embarking directly into the details, we first give an example of the guidance behavior using a Dubins' vehicle. This simple example is meant to introduce the key concepts that will be discussed later.

Fig. 5.1 shows the time-optimal guidance behavior of a Dubins' vehicle for an environment with multiple obstacles. The behavior is generated using a continuous Dijkstra algorithm[1] . An optimal trajectory can be obtained by following the velocity associated with the vector field. One interesting observation we can make is that two regularities (later called equivalences or symmetries) can be found in the resulting guidance behavior. One regularity is illustrated by the two purple trajectories. These two trajectories start from different initial states and end at different goal states. However, the trajectories can be perfectly superimposed with one another following a translation and a rotation. Later we will say that these two trajectories are realizations of a same *guidance primitive*. The other regularity is illustrated by the two black trajectories. These two trajectories start from different initial states, but eventually, after they meet at point b, follow the same route to the destination. Later we will call the point (point b) where the two trajectories meet a *subgoal*[2] . It is important to realize that both of these two regularities are not introduced artificially for the the purpose of description and modeling. Instead they are fundamental properties of the agent-environment interaction. They will be explained and formalized in Section 5.3.

It can be clearly seen that, after the introduction of these two regularities, the descriptive complexity of guidance behavior can be greatly reduced. As can be observed in Fig. 5.1, combining all trajectories that meet at the same point and then follow the same route (as illustrated using the two black trajectories) results in a partitioning of the entire behavior into cells. These cells are denoted by capital characters and their corresponding subgoals are denoted by lowercase characters. Notice that since trajectory segment π_1 is equivalent to π_2 (the meanings of these symbols will be elaborated on in the next section), after this equivalence operation, any trajectory starting from partition B can simply be written as $\pi_1\pi_5$. Furthermore, equating the partitions or cells according to whether they can be superimposed with one another following some transformations (as illustrated using the two purple trajectories) results in an even more concise representation. Now each partition can be viewed as a sample "template" that

[1] The details are illustrated in Appendix A

[2] Strictly speaking, a subgoal belongs to the state space $\mathcal{X} \times \mathcal{E}$. So, in this case, it should include point b, which is defined by its two coordinates, x_b and y_b, the velocity at point b, which is defined by speed v_b and course angle ψ_b, and the environment state $\mathbf{0}^{4\times 1}$. The subgoal will be formally defined in Section 5.3.2.

we will later call an *interaction pattern*. Each trajectory segment can be considered as the realization of some specific template, and the templates themselves belong to some pre-defined library or "repertoire".

The practical and theoretic appeal of this scheme is obvious, considering that, for an agent operating in this or some similar environments, completing a guidance task does not require "remembering" or specifying how to negotiate each and every corner. Once a library of templates (interaction patterns) is known, the entire behavior can be organized according to the templates within this library. Being able to elucidate the principles dictating the emergence of these templates and their organization is valuable to study both human and animal behaviors as well as to inspire novel guidance algorithms.

Before elaborating further, some points require clarification. First of all, the vehicles involved in practical applications have more complex dynamics than a Dubins' vehicle. We also assume here that the agent has complete knowledge of the environment and perfect sensory and motor capabilities. Is it possible to generalize the conclusions derived from the behavior of a Dubins' vehicle to other vehicles operating under realistic sensory and motor capabilities? Second, the usage of "template" or "repertoire" here is rather qualitative. Is there a way we can formally characterize them, while keeping them as general as possible? Third, the Dubins' vehicle is also assumed to obey time-optimal behavior. Humans performing guidance tasks face the challenges related to high degrees of freedom, and, in many instances, are known to resort to satisfactory solutions instead of those that are optimal [139]. How is it possible to accommodate for these particular characteristics while preserving a rigorous and formal framework? Finally, we would like to deploy this framework to help us gain insights into the organization of human guidance behavior. How are we to design informative guidance experiments and subsequently process and analyze the data within this framework? This chapter and the next two chapters will address these questions.

5.3 Interaction Patterns

In this section, based on the formal definition of guidance behavior as closed-loop agent-environment dynamics, we define the interaction patterns that are inherent to guidance behavior. We achieve this by using the concept of equivalence or symmetry, which is

widely used in dynamical system theory [140] and geometric methods [141]. Investigating the symmetries that exist in the agent-environment dynamics leads to two key concepts: guidance primitives and subgoals, which combined, define the overall concept of interaction patterns. Taken together, these new concepts provide the foundations for a formal framework needed for the investigation and modeling of guidance behavior.

5.3.1 Guidance Primitives

In this subsection, we extend the concept of motion primitive, an already well-established concept in motor control [50], and more specifically its relation to the symmetry in vehicle dynamics, which was developed by Frazzoli et al. [4]. The outcome of our formulation is the concept of guidance primitive, which captures the symmetries in the dynamic interactions between the agent and the environment. In the following we adopt a similar notation as was used in [4].

One fundamental property of many vehicles, such as cars and aircrafts, is that their dynamics are invariant with respect to some transformations. This invariance can be formalized using symmetry groups [4]. Suppose there exists a finite-dimensional Lie group M. The action of this group M on the state manifold $\mathcal{X}$ can be written in the form of a mapping as follows:

$$\Psi : M \times \mathcal{X} \to \mathcal{X} \tag{5.1}$$

such that, for every $m, n \in M$, and $\mathbf{x} \in \mathcal{X}$,

$$\Psi(m, \Psi(n, \mathbf{x})) = \Psi(mn, \mathbf{x}) \tag{5.2}$$

In [4], the symmetry group M is defined for the agent S described by (2.1): for all $m \in M$, $\mathbf{x} \in \mathcal{X}$, $t \in [0, t_f]$ and all $\mathbf{u} \in \mathcal{U}$,

$$\Psi(m, \mathbf{x}^{\mathbf{u}}(t; \mathbf{x}_0)) = \mathbf{x}^{\mathbf{u}}(t; \Psi(m, \mathbf{x}_0)) \tag{5.3}$$

This equation implies that if $t \to (\mathbf{x}(t), \mathbf{u}(t))$ is an integral curve of (2.1), so is $t \to (\Psi(m, \mathbf{x}(t)), \mathbf{u}(t))$.

To account for the agent's interaction with the environment, we need to express the symmetry with respect to the closed-loop dynamics (2.8) instead of just the agent dynamics. Suppose the environment state can be written in the form of some relative

quantities, then (2.5) can be rewritten as:

$$\mathbf{e}(t) = \mathbf{x}(t) - \mathbf{x}_r \tag{5.4}$$

where $\mathbf{x}_r$ is some reference state. Under such an assumption, manifold $\mathcal{E}$, in which the environment state resides, is a subspace of the state manifold $\mathcal{X}$, and we can write the restriction of mapping Ψ in $\mathcal{E}$ as $\Psi|\mathcal{E}$:

$$\Psi|\mathcal{E} : M \times \mathcal{E} \to \mathcal{E} \tag{5.5}$$

And we say that $\mathcal{E}$ is invariant with respect to the group action $\Psi|\mathcal{E}$, or equivalently that M is a symmetry group for $\mathcal{E}$, if the following condition holds: for all $m \in M$, $\mathbf{e} \in \mathcal{E}$, $t \in [0, t_f]$ and all $\mathbf{u} \in \mathcal{U}$,

$$\Psi|\mathcal{E}(m, \mathbf{e}^{\mathbf{u}}(t; \mathbf{e}_0)) = \mathbf{e}^{\mathbf{u}}(t; \Psi|\mathcal{E}(m, \mathbf{e}_0)) \tag{5.6}$$

where $\mathbf{e}^{\mathbf{u}}(t; \mathbf{e}_0)$ is the environment state trajectory by plugging (2.2) into (5.4), and $\mathbf{e}_0$ is the initial environment state, which is $\mathbf{x}_0 - \mathbf{x}_r$. (5.6) implies that if $t \to (\mathbf{x}(t), \mathbf{u}(t), \mathbf{e}(t))$ is an integral curve of (2.1) and (5.4), so is $t \to (\Psi(m, \mathbf{x}(t)), \mathbf{u}(t), \Psi|\mathcal{E}(m, \mathbf{e}(t)))$.

Similar to [4], we say that two guidance behaviors, including both the agent state trajectory $\mathbf{x}(t)$ and the environment state trajectory $\mathbf{e}(t)$, are equivalent, if they can be superimposed through a time translation and an action from a group M, i.e., two behaviors, $\pi_1 : [t_{i1}, t_{f1}] \to (\mathbf{x}(t), \mathbf{u}(t), \mathbf{e}(t))$ and $\pi_2 : [t_{i2}, t_{f2}] \to (\mathbf{x}(t), \mathbf{u}(t), \mathbf{e}(t))$, are equivalent, if $t_{f1} - t_{i1} = t_{f2} - t_{i2}$ and there exist a time translation T and a group action $m \in M$ such that $(\mathbf{x}(t), \mathbf{u}(t), \mathbf{e}(t)) = (\Psi(m, \mathbf{x}(t+T)), \mathbf{u}(t+T), \Psi|\mathcal{E}(m, \mathbf{e}(t+T)))$ for all $t \in [t_{i1}, t_{f1}]$. For any valid trajectory $\pi : [0, t_f] \to (\mathbf{x}(t), \mathbf{u}(t), \mathbf{e}(t))$, a *guidance primitive* is defined as the class of all the trajectories that are equivalent to π.

The set of all guidance primitives for an agent S and an environment E with symmetry group M is called a guidance primitive library, which can be denoted as Π. Π can then be used as a representation of the entire extended state-space $\mathcal{X} \times \mathcal{U} \times \mathcal{E}$. Specifically, the guidance primitives allow us to divide the extended state-space into mutually exclusive and jointly comprehensive classes. Each trajectory class can be represented by a symbol $\pi_i \in \Pi$, i.e.,

$$\eta : \mathcal{X} \times \mathcal{U} \times \mathcal{E} \to \Pi := \{\pi_1, \pi_2, ...\} \tag{5.7}$$

A particular trajectory, one of the black trajectories shown in Fig. 5.1 for instance, can be divided into a sequence of guidance primitives and represented as a string of symbols, $\pi_1\pi_5$ for one of the black trajectories.

5.3.2 Subgoals and Interaction Patterns

Due to the inclusion of the environment state in the definition of guidance primitives, some symmetries in the sense of (5.3) are broken. For instance, in Fig. 5.1, if only considering the agent state $\mathbf{x}(t)$, the trajectories π_1 and π_2 are obviously equivalent to one another. However, when the environment state $\mathbf{e}(t)$ is included, the two are not equivalent anymore. Consequently, the number of guidance primitives needed to describe the guidance behavior increases, which in turn increases the computational load and storage involved in planning and other operations. Furthermore, as already pointed out in Section 5.2, even after the environment state is included, some equivalence relationships between the two trajectories, π_1 and π_2, still exist. Namely, as can be noted in Fig. 5.1, the two trajectories highlighted in partition B follow the same route after they pass point b. And second, within that partition B, one of the two trajectories can be continuously deformed into the other without intersecting the obstacles (this property is often related to the concept of homotopy [142]). In the following we will formalize this equivalence in the context of subgoal.

First, we introduce a symbolic state-space representation, which is obtained through quantization,

$$q : \mathcal{X} \times \mathcal{E} \to \mathcal{A} \tag{5.8}$$

where $\mathcal{A}$ is a finite set of symbols which is called the state alphabet [143]. Elements of $\mathcal{A}$ are called letters, each of which corresponds to a grid point of the quantized state space. With such a quantization, the set of all trajectories can then be written as:

$$\overleftarrow{S} = \{\overleftarrow{s}_i^L : s_{i-L+1}, ..., s_i, s \in \mathcal{A}, L \in \mathbb{Z}^+, i \in \mathbb{Z}\} \tag{5.9}$$

or

$$\overrightarrow{S} = \{\overrightarrow{s}_i^L : s_i, ..., s_{i+L-1}, s \in \mathcal{A}, L \in \mathbb{Z}^+, i \in \mathbb{Z}\} \tag{5.10}$$

where L is the length of the trajectory, and i is the index of the end state for definition (5.9) and the start state for definition (5.10). Notice that (5.9) and (5.10) are

actually equivalent. We introduce these two different representations for the purpose of distinguishing trajectory segments going to or coming from a particular state s_i.

Similarly, a quantization of the control space can be introduced,

$$q^u : \mathcal{U} \to \Sigma \tag{5.11}$$

where Σ is another finite set of symbols called the control alphabet. And the set of all control sequences can be written as:

$$\overrightarrow{U} = \{\overrightarrow{u}_i^L : u_i, ..., u_{i+L-1}, u \in \Sigma, L \in \mathbb{Z}^+, i \in \mathbb{Z}\} \tag{5.12}$$

The definition of symmetry groups, (5.3) and (5.6), and subsequently the definition of guidance primitive can be extended to this symbolic formulation and provide a more convenient way to define two important concepts: the subgoal and interaction patterns.

First, (5.3) and (5.6) can be re-formulated as:

$$(\Psi(m, \overrightarrow{s}^L(s_i)), \overrightarrow{u}_i^L) = (\overrightarrow{s}^L(\Psi(m, s_i)), \overrightarrow{u}_i^L) \tag{5.13}$$

where $\overrightarrow{s}^L(s_i)$ represents a trajectory starting from s_i and $\overrightarrow{s}^L(\Psi(m, s_i))$ represents a trajectory starting from $\Psi(m, s_i)$.

Then, using the symbolic representation, we can introduce two types of relationships over $\overleftarrow{S}$. One relationship $\sim_g$ is a re-statement of the definition of guidance primitives (5.13): two trajectories, $\overleftarrow{s_i}^L$ and $\overleftarrow{s_j}^L$, satisfy $\overleftarrow{s_i}^L \sim_g \overleftarrow{s_j}^L$ if there exist a $m \in M$ and a control sequence $\overrightarrow{u}_i^L$ such that:

$$(\Psi(m, \overleftarrow{s}_i^L), \overleftarrow{u}_i^L) = (\overleftarrow{s}_j^L, \overleftarrow{u}_i^L) \tag{5.14}$$

The other relationship $\sim_s$ is defined using the concept of causal state [107]:

$$\overleftarrow{s_i}^K \sim_s \overleftarrow{s_j}^L \Leftrightarrow P(\overrightarrow{s}|\overleftarrow{s_i}^K) = P(\overrightarrow{s}|\overleftarrow{s_j}^L) \tag{5.15}$$

for all semi-infinite $\overrightarrow{s} = s_0 s_1 ...$, where $K, L \in \mathbb{Z}^+$ and P stands for probability. Since in this paper we limit the treatment to deterministic systems, P equals one.

In this setting, the state s_0 is then called a *subgoal*, in the sense that, from this state on, the two trajectories, $\overleftarrow{s_i}^K$ and $\overleftarrow{s_j}^L$, will follow the same trajectories $\overrightarrow{s}$. We will drop the length variables K and L here and denote the members of any length in the set $\overleftarrow{S}$ of (5.9) by $\overleftarrow{s}$.

It can be easily verified that both $\sim_g$ and $\sim_s$ are equivalence relationships. Thus, if $\overleftarrow{s} \in \overleftarrow{S}$, one type of equivalence class over $\overleftarrow{S}$ can be defined in the following two steps (the order of these two operations is interchangeable): first,

$$[\overleftarrow{s}] = \{\overleftarrow{s}' \in \overleftarrow{S} : \overleftarrow{s}' \sim_s \overleftarrow{s}\} \tag{5.16}$$

and second,

$$[[\overleftarrow{s}]] = \{[\overleftarrow{s}]' \in [\overleftarrow{s}] : [\overleftarrow{s}]' \sim_g [\overleftarrow{s}]\} \tag{5.17}$$

Each equivalence class $[[\overleftarrow{s}]]$ defines an *interaction pattern* to reflect the fact that it captures invariances inherent in the agent-environment interactions.

And the set of all interaction patterns for an agent S and an environment E with symmetry group M is called an *interaction pattern library*, which can be denoted $\tilde{\Pi}$. Similar as Π, it provides a way of parsimoniously representing $\mathcal{X} \times \mathcal{U} \times \mathcal{E}$:

$$\tilde{\eta} : \mathcal{X} \times \mathcal{U} \times \mathcal{E} \to \tilde{\Pi} := \{\tilde{\pi}_1, \tilde{\pi}_2, ...\} \tag{5.18}$$

With these definitions, behavior can be described and representation using the symbols belonging to $\tilde{\Pi}$ instead of the entire state space. Actually, the cardinality of an interaction pattern library $\tilde{\Pi}$ is much smaller than the cardinality of its corresponding guidance primitive library Π. Together the different interaction patterns describe the range of guidance behaviors an agent can utilize when approaching subgoals.

5.3.3 Dubins' Vehicle Illustration

Going back to the guidance behavior of Dubins' vehicle for purpose of illustration, the agent state described by (2.1) and the environment state described by (5.4) are invariant with respect to a translation and a rotation about a vertical axis, or to the actions of the symmetry group $M = SE(2)$. Each element of M can be written in the form of a 3×3 matrix as follows:

$$m = \begin{bmatrix} \cos\psi & -\sin\psi & t_x \\ \sin\psi & \cos\psi & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{5.19}$$

where the rotation is characterized by angle ψ and the translation is characterized by a vector $\mathbf{t} = [t_x, t_y]'$. For the example trajectories shown in Fig. 5.1, according to

(5.14) and (5.15), we have $\pi_1 \sim_s \pi_2$ and $\pi_3 \sim_g \pi_4$. And taking the $\sim_s$ equivalence (5.16) results in the partitions of the state space shown in Fig. 5.1, and subsequently taking the $\sim_g$ equivalence (5.17) results in two interaction patterns: one corresponds to approaching the subgoals in the counterclockwise direction, as is the case for the guidance behaviors in partition A; the other corresponds to approaching the subgoals in the clockwise direction, as is the case for partition B. Actually, after a mirror reflection symmetry is added to group M, only a single interaction pattern is left.

5.4 A Few Words

Compared with other chapters of this dissertation, this chapter is slightly math-intensive. However, the essence of the concepts introduced in this chapter should not be difficulty to grasp. We found two symmetries or equivalences that are inherent to guidance behavior. One is defined with respect to symmetry groups, and the other is defined with respect to subgoals. The first symmetry implies that, no matter the locations or the times of two agents, as long as the agents have similar dynamics, the agents face similar environments, and the controls are the same, they will exhibit similar guidance behavior. This symmetry definition is rather intuitive. If it does not hold, we would have to learn how to drive every time we get to a new place. The second symmetry implies that there are certain spatial consistency within the guidance behavior. Under certain circumstance, the agent can use just one trajectory to represent a whole bundle of trajectories.

Taken together, these two symmetries (or the definition of interaction patterns) imply that they are the key to understand the mechanism that can be adopted by agents to reduce the complexities involved in planning and other operations (to solve the curse of dimensionality). Taking Dubins' vehicle's guidance behavior as an example, the concept of interaction pattern means that, if we were live in a world where every vehicle has the Dubins' dynamics and we had learned how to drive it at one place and one time, then we should be able to deal with every other situations. We only need to assign a sequence of subgoals, and then reuse the same control policy and regenerate the same guidance behavior again and again. That is to say, in order to navigate around that world, only one skill or interaction pattern in our repertoire is sufficient.

The concepts introduced in this chapter, guidance primitives, subgoals and interaction patterns, can be used to the description of both humans and artificial agents, such as Dubins' vehicle used in this chapter and UAVs used in the previous two chapters. In the next two chapters, we are going to focus on human guidance behavior. We are going to look at how these concepts will evolve in the context of human guidance. In Chapter 6, we are going to show some experiments we designed for the investigation of human guidance behavior. And then in Chapter 7, we are going to use the concepts and the framework developed in this chapter to analyze the experimental data.

Chapter 6

Lab Facility and Guidance Experiments

6.1 Introduction

Our formalization of guidance behavior as closed-loop agent-environment dynamics in Chapter 2 and our introduction of the concepts of guidance primitive, subgoal and interaction pattern in Chapter 5 provide the foundations for a *unified*, *formal* and *grounded* framework for the modeling and analysis of guidance behavior of both natural and artificial agents. As explained in Section 1.4, by *unified*, we mean that the guidance behavior definition we have encompasses a wide range of processes, such as perception, planning and control. And it also comes across the boundary between the agent and the environment. By *formal*, we mean that we are able to use the framework to quantify and codify guidance behaviors. The mathematical language we use is dynamical system theory [140] and formal methods [143]. By *grounded*, we mean that, as opposed to model-based approach (as described in Section 1.2.1) where the representations are prescribed by hand, the structures and representations we use in our modeling and analysis are inherent to guidance behavior and they essentially emerge from agent's interaction with the environment.

In this chapter and the next chapter, we use the analysis of human guidance behavior as an example to show details of how such a framework works. In this chapter, we first introduce the lab facility we use for the investigation of human guidance behavior. We

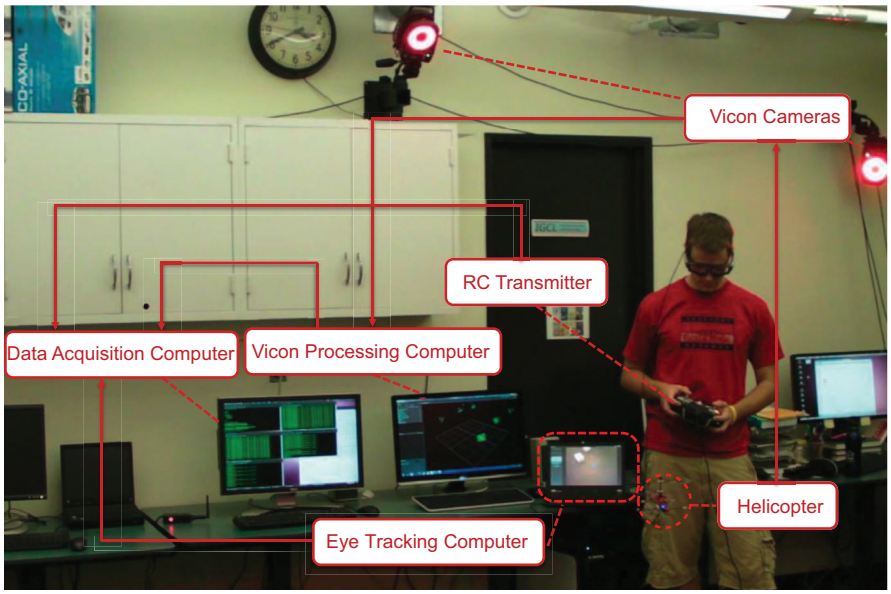


Figure 6.1: Architecture of our indoor flight test facility.

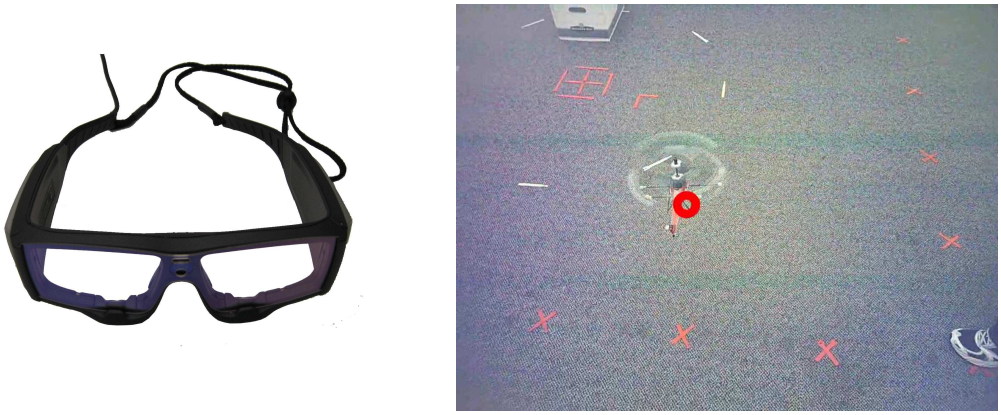
then describe the experiments and the data pre-processing procedures. Finally, we show some preliminary analysis results of the experimental data. The main purpose of this analysis is to compare intra-subjective performances. A more systematic analysis based on our formal framework will be carried out in the next chapter.

The material presented in this chapter is first published in papers [112, 144, 145].

6.2 Interactive Flight Test Facility

The experiments were conducted in the Interactive Guidance and Control Lab (IGCL). Fig. 6.1 gives an overview of the key hardware components that make up the lab facility. The hardware components include joystick user inputs, direct transmission of control signals to Spektrum RC receivers, real-time readings from a Vicon motion tracking system, an eye tracking device, and integration with onboard systems.

The measurement and recording of the position and orientation of the aerial vehicles (helicopters in our case) in flight is performed by a Vicon motion tracking system [146]. The motion tracking system consists of 6 high-speed MX-40 cameras that are affixed to the walls of the lab. The cameras are connected to a router and from there to a



(a) (b)

Figure 6.2: (a) SMI Eye Tracking Glasses (ETG) and (b) view from the scene camera showing the focus of visual attention as a red circle.

computer via an Ethernet link. Retro-reflective spherical markers are placed on the vehicle to form a shape that can be uniquely identified. Once the Vicon tracking system software (Vicon Tracker) has identified the helicopter as a tracked object, the position and orientation information can be streamed over a network connection. The system is capable of providing the vehicle's 6 DOF motion in real time with a sub-millimeter accuracy [147].

Operator visual gaze tracking is achieved using a pair of Eye Tracking Glasses (ETG) from SensoMotoric Instruments SMI [148]. (Fig. 6.2(a)). The ETG system consists of binocular eye tracking cameras that automatically compensate for parallax (making gaze tracking accurate at different distances). The eye tracking covers a range of 80° horizontally and 60° vertically with an accuracy of 0.5°. A forward facing scene camera (1280×960 resolution at 24 frames per second) records what the pilot observes (Fig. 6.2(b)). The scene camera has a field of view of 60° horizontal and 46° vertical.

The experiments are conducted with an off-the-shelve RC helicopter - Blade MCX (19 cm rotor diameter, and 35g with makers and battery on). It is controlled by a standard four-channel RC transmitter. The Vicon system only requires retro-reflective spherical markers to be affixed to the vehicle. On-board electronics are limited to the standard 4 axes receiver. This makes it possible to use highly miniaturized helicopters.

The data collected by the cameras are further processed by the ViconIQ software

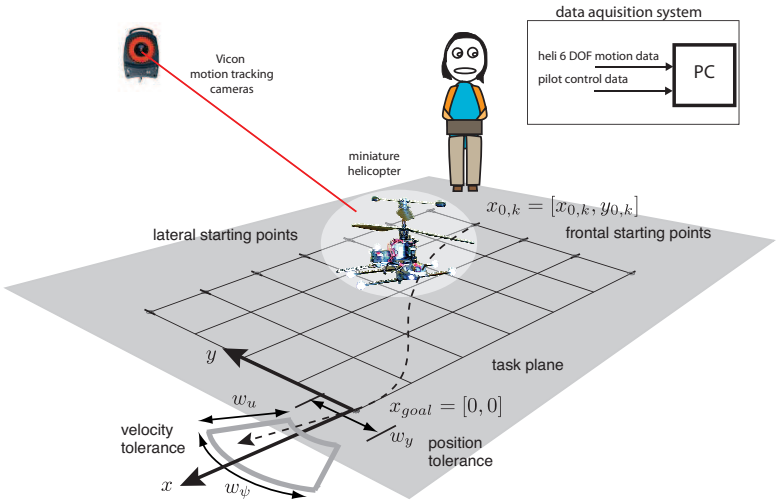


Figure 6.3: Illustration of the experimental setup. The pilot has to fly the helicopter from a number of initial starting positions to a goal set. The goal set is defined as the point $\mathbf{p}_0 = (0, 0)'$ with a lateral position tolerance $w_y = \pm.15$ meters and by a terminal state $v_{\text{goal}} = 1$ m/sec and course angle $\psi_{\text{goal}} = 0$ deg (subject to the tolerances $w_v = \pm 0.35$ m/sec and $w_\psi = \pm 15$ deg, respectively). The pilot control inputs are recorded simultaneously to the vehicle 6-DOF data.

running on a personal computer. The geometrical configuration of the markers is precisely calibrated by the tracking software, which then treats the configuration as a single rigid body. ViconIQ then generates the 6 degrees of freedom (DOF) motion tracking data. This data comprises the three translational DOF in a room-defined reference frame (x, y, z) and the three Euler angles (ϕ, θ, ψ).

The four pilot inputs (longitudinal and lateral cyclic inputs, collective motor speed, and differential motor speed) are recorded and integrated into a single data stream. The markers are tracked with an accuracy of approximately 0.02 pixels, which corresponds to about 40 μm in our experimental space. This level of resolution can be achieved at a sampling frequency up to approximately 160 Hz. The sampling time was set to $T_s = 0.01$ sec.

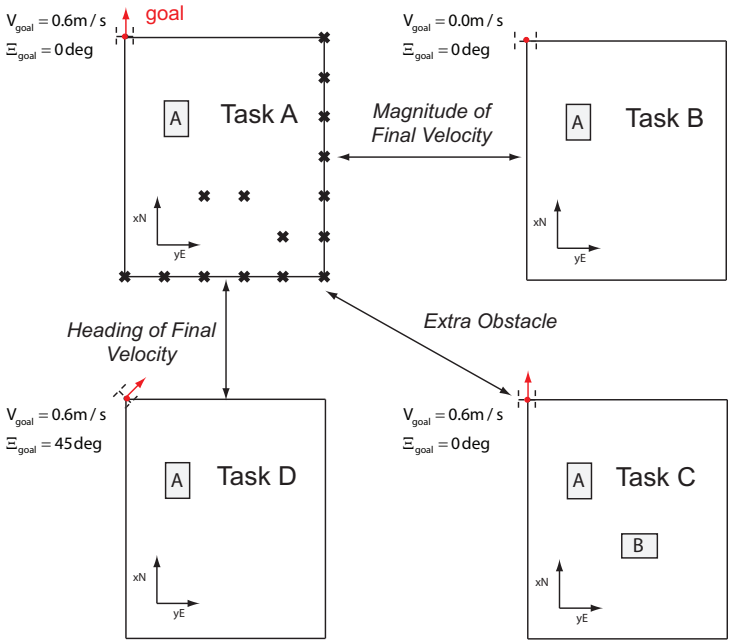
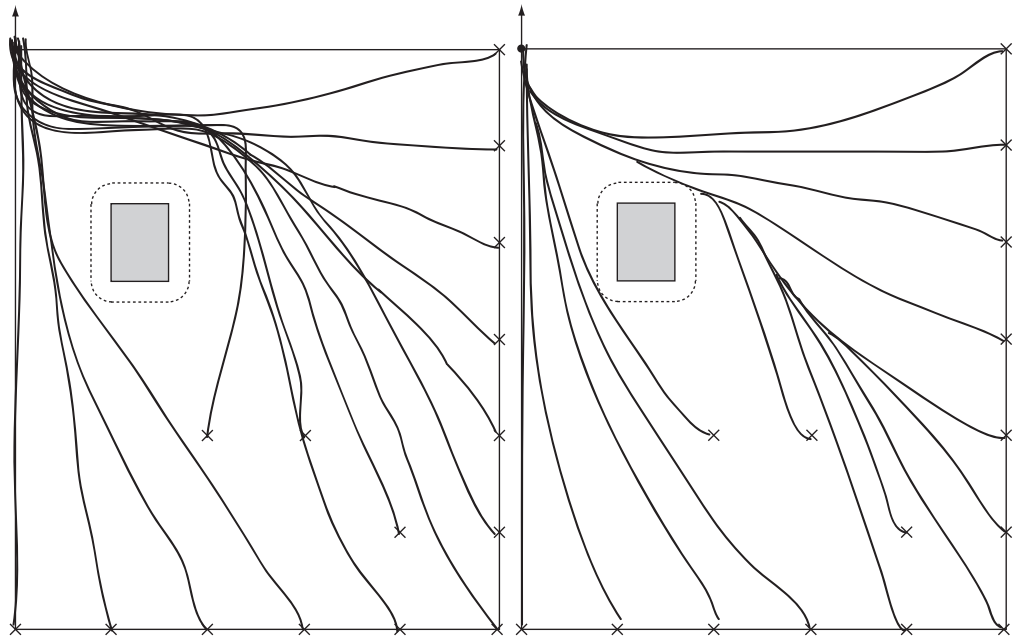


Figure 6.4: Illustration of the different task settings used in own investigation. Environment factors were manipulated so as to create opportunities to study how the pilots' spatial behaviors changes and identify invariants under different task settings. Three factors were chosen in our experiments: the i) magnitude of the final velocity, ii) the direction of the final velocity and iii) presence of an extra obstacle. The magnitudes of the final velocity was specified indirectly using an obstacle behind the goal location (forcing the pilot to stop within a finite distance). Starting locations are indicated by crosses in task A (staring locations in other tasks are not shown here).

6.3 Guidance Experiments

Two students from University of Minnesota were recruited as test pilots. The subjects have distinct skill levels. One subject was an advanced pilot with good facilities to fly the helicopter as well as maneuver in challenging task environments. The other was an intermediate pilot with sufficient skills to complete the task but more erratic performance. The general task setup, as illustrated in Fig. 6.3 and Fig. 6.4, is an extended version of one used in our previous work [112].

The pilot had to start from a stationary hover above one of the specified initial



(a) Intermediate Pilot (b) Advanced Pilot

Figure 6.5: Drawings for task A from the two pilots.

positions (within a circle of .25m radius) and guide the helicopter to a specified arrival location under different obstacle configurations and arrival conditions (total of four configurations A, B, C and D as depicted in Fig. 6.4). The initial positions were selected to span the task space. The initial heading was unconstrained, while the arrival heading and velocity were both constrained. Tolerances on the arrival conditions were used to insure adequate task feasibility (success rate above 90%). The goal set was defined by $W_y = \pm.3$ meters and $W_\psi = \pm 30$ degrees. To capture inter-subject variability, multiple trials were conducted for each starting location.

Training was conducted for each task configuration (A-D) before data collection. The pilots were asked to explore the task space freely and try as many routes as possible. They were also asked to try to fully exploit the helicopter's maneuvering capabilities in order to reach the goal in the shortest time while satisfying the end constraints and avoiding the obstacles[1] . Given the typical trade-off between arrival time and arrival

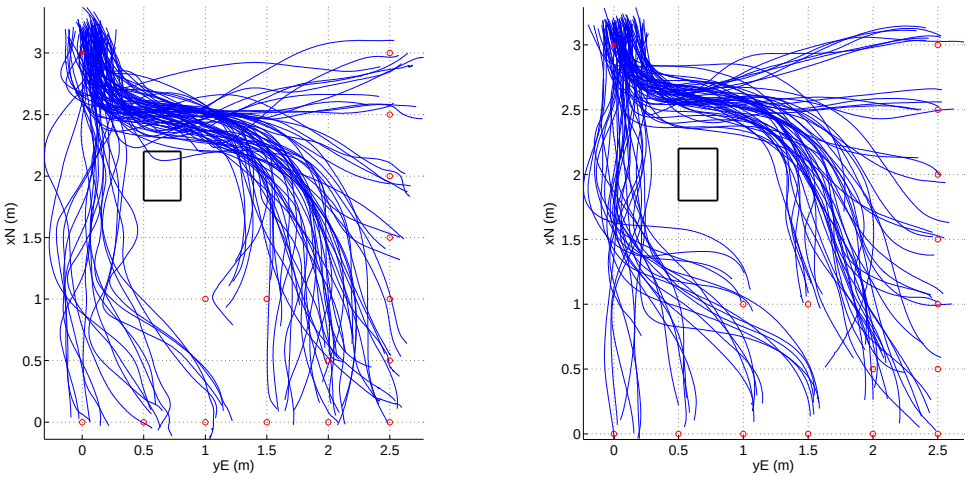
[1] Please notice that although in our experiment, as well as in the Dubins' vehicle's example, we use time as the cost, our definition of interaction pattern is not restricted to minimum-time behavior. The

precision, the pilots have to carefully adjust their strategies to fulfill guidance task requirements.

One potential confounding factor was the pilots' standing positions. To evaluate the effects of this factor, two separate experiments were conduced with task A. In one experiment, the pilots could stand where they wanted and move while flying the helicopter; in the other experiment, the pilots could only stand at a fixed position during the test. No significant difference was found in pilots' performances with respect to these two experiments. So in following analysis, the influence of this factor can be ignored.

After each task, the pilots were asked to draw on a piece of paper the "best" trajectory from each starting location. Their drawings were then scanned. And finally, pencil tool of Adobe Illustrator was used to trace the trajectories within each drawing. Such drawings for task A are shown in Fig. 6.5.

6.4 Data Pre-Processing Procedures



(a) Intermediate-Skilled Pilot

(b) Advanced-Skilled Pilot

Figure 6.6: Extracted trajectories for task A after pre-processing. Starting locations are indicated as red circles and the goal is indicated as a red star.

concepts of guidance primitive (5.13) and subgoal (5.15) are quite general. Therefore they can be used to describe and analyze a variety of behaviors, such as minimum-energy behavior.

The collected raw data was pre-processed to extract the set of relevant trajectory segments. The first step of pre-processing was to discard segments that did not satisfy the end constraints as well as the return segments (time interval between the goal interception and the beginning of the next trial). The pre-processing also involved interpolating intermittent drop outs due to loss of visual tracking. The drop-outs were isolated and only involved a few samples at a time, thus the interpolation did not alter the data to a significant degree.

The measurement data obtained after pre-processing is described by $\mathbf{x}^n(i), i = 1, ...N_n, n = 1, ..., N$ with N as the number of trajectories and N_n as the number of data point for trajectory n. The helicopter planar rigid-body motion is fully characterized by four variables $\mathbf{x}^n(i) := [x^n(i), y^n(i), v^n(i), \psi^n(i)]'$, where $[x^n(i), y^n(i)]'$ is the position, $v^n(i)$ is the speed and $\psi^n(i)$ is the course angle.

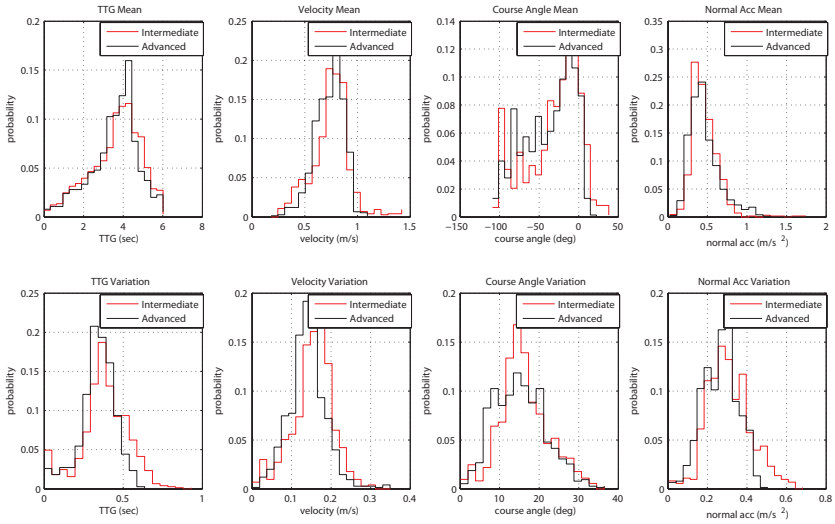
6.5 Preliminary Analysis: Intra-Subjective Comparison

Fig. 6.6 shows the collected trajectories for task A after pre-processing. Histograms and cumulative distribution functions for the mean time-to-go, speed, course angle and normal acceleration computed for these trajectories are illustrated in Fig. 6.7. The two pilots exhibit similar statistical profiles. However, the variability in these variables is much smaller for the advanced-skilled pilot than for the intermediate-skilled pilot. This observation can also be made directly from their trajectories. It can be seen that the fluctuation within a same ensemble of trajectories is not uniform along the course of a flight (taking the ensemble starting from (x = 3.0 m, y = 2.5 m) for instance). For both pilots, the trajectories are more broadly distributed in the middle than at the two ends. But the distribution envelopes for the advanced-skilled pilot are almost always narrower than those for the intermediate-skilled pilot.

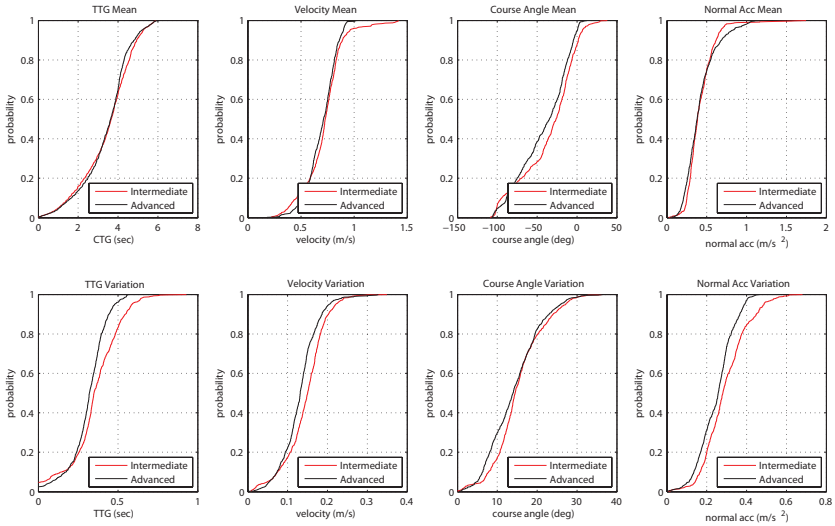
Most of these individual differences can be attributed to the pilots' different lower-level control skills, such as staying on track of a route or maneuvering smoothly around the corner of an obstacle. But it is our understanding that higher-level spatial comprehension and planning skills also play a role here. To verify this, after each task, the pilots were asked to draw the "best" trajectories on a a sheet of paper. Their drawings for task A are shown in Fig. 6.5. These drawings can be taken as a reflection of their planning

skills. In the intermediate-skilled pilot's drawing, it appears that not all the helicopter's dynamic capabilities are well respected. For instance, some of the sharp turns drawn at the northeastern corner of the obstacle would be hard to achieve. This is also the region where large fluctuations are observed in their executions. In the advanced-skilled pilot's drawing, on the other hand, the helicopter's dynamic capabilities are well respected. In addition, it could also be seen that, he organizes his guidance according to 'subgoals' (even though he is not aware of the concept), especially for those trajectories starting from southeast (lower-right).

In this section, we focused on finding the intra-subjective differences. In the next chapter, we are going to switch gear and focus on analyzing the similarities across different subjects. We are going to see what are the universal principles employed by humans even though they may have quite different skill levels. We especially are going to do this by using the concepts formalized in the previous chapter, such as guidance primitive, subgoal and interaction patterns.



(a) Histograms



(b) Cumulative Distribution Functions

Figure 6.7: Histograms and cumulative distribution functions of mean values and standard deviations of following variables extracted from trajectories of task A (Fig. 6.6): time-to-go (TTG) value, speed, course angle and normal acceleration (from left to right).

Chapter 7

Analysis Results and Discussion

7.1 Introduction

In this chapter, we use framework described in Chapter 2 and Chapter 5 to analyze data collected from experiments described in Chapter 6. We take six steps for the analysis:

- Step 1 (Section 7.2): Extract $\sim_s$ equivalence according to definition (5.15).
- Step 2 (Section 7.3): Extract $\sim_g$ equivalence according to definition (5.14). With these two equivalence extracted, according to the interaction pattern definition as described in Section 5.3.2, we are able to abstract out the patterns that are inherent to the pilots' guidance behavior.
- Step 3 (Section 7.4): Zoom in and analyze dynamical characteristics of each interaction pattern by using piecewise affine (PWA) model.
- Step 4 (Section 7.5): Zoom out and analyze how humans use interaction patterns to organize their guidance behavior.

All the analysis results are put together and used to build a formal model-a hierarchical hidden Markov (HMM) model-of human guidance behavior. This chapter ends with a discussion about the implications of the analysis results on the understanding of human guidance behavior and the potential applications of the results on the design of autonomous systems.

The material presented in this chapter is first published in papers [113, 144, 114, 149, 150].

7.2 $\sim_s$ Equivalence Extraction

Our analysis method extracts interaction patterns from the measurement data in two steps: First, transforming the measurement data into a symbolic representation and then partitioning experimental trajectories into segment clusters based on the $\sim_s$ equivalence (5.15). Second, examining the $\sim_g$ equivalences between the extracted segment clusters based on (5.14), which will be described in Section 7.3. The example used to illustrate our method in this and the following sections is mainly based on the data obtained from the advanced-skilled pilot's performing task A.

7.2.1 Symbolic Representation and Subgoal Identification

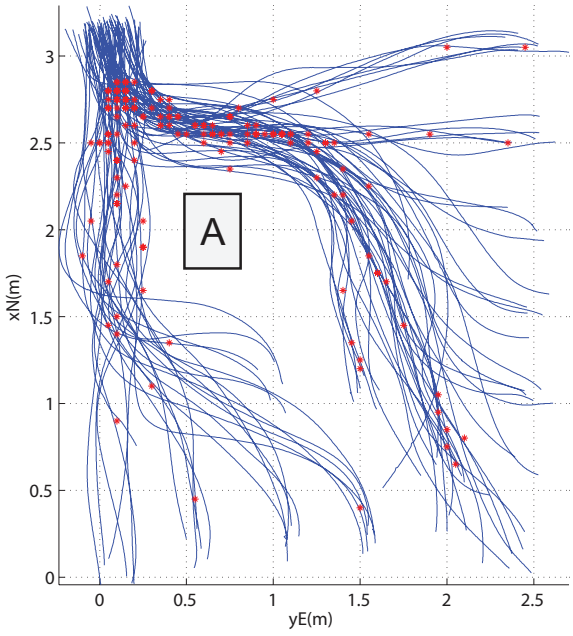


Figure 7.1: Identified subgoals marked by red stars for task A with the advanced-skilled pilot.

To be able to use $\sim_s$ equivalence definition (5.15), the measurement data must first be transformed into its symbolic representation. This transformation can be accomplished by quantifying the state space into mutually exclusive cells according to (5.8). Each cell corresponds to a letter of the state alphabet $\mathcal{A}$. If a data point $\mathbf{x}^n(i)$ falls within a cell, it is represented by the corresponding letter $s^n(i)$ with $s^n(i) \in \mathcal{A}$. Once the transformation is done for each data point, the original measurement data $\mathbf{x}^n(i), i = 1, ...N_n, n = 1, ..., N$ is then transformed into its corresponding symbolic form as $s^n(i), i = 1, ...N_n, n = 1, ..., N$. Please notice that the symbolic form is only used for this step. The other steps are implemented using the original measurement data in the form of time series.

The $\sim_s$ equivalence relationship (5.15) is then applied to the symbolic representation of the measurement data in a pairwise manner in order to find the common subgoal for every pair of trajectories (if the subgoal does exist). To be more specific, we assign s^* to be a subgoal of trajectories s and s' if they follow different trajectories before s^* and follow the same trajectory after s^*, i.e.,

$$s := \overleftarrow{s}_i^{K1}, \overrightarrow{s}_{i+1}^{L} \text{ and } s' := \overleftarrow{s'}_j^{K2}, \overrightarrow{s'}_{j+1}^{L}, \text{ with } \overrightarrow{s}_{i+1}^{L} = \overrightarrow{s'}_{j+1}^{L} \tag{7.1}$$

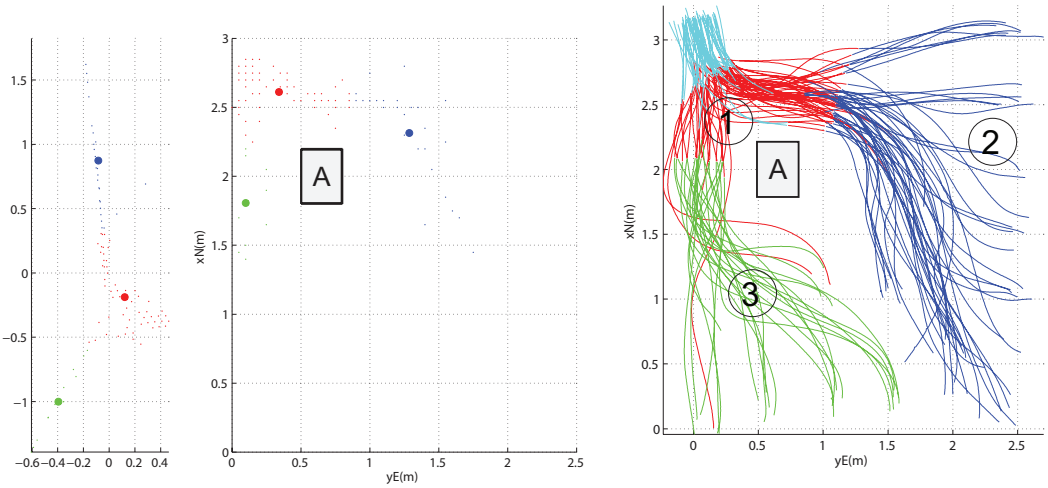
For trajectory s, $s^* = s_{i+1}$ and for trajectory s', $s^* = s'_{j+1}$. Please notice that, in (7.1), the superscript denotes the length of the trajectory segment not the index of the trajectory as in $s^n(i)$.

In practice, a threshold ϵ is added in order to tolerate small perturbations. The strict equality requirement $\overrightarrow{s}_{i+1}^{L} = \overrightarrow{s}_{j+1}^{L}$ is replaced with an inequality requirement as follows:

$$\sum_{k=1}^{L} ||s_{i+k} - {s'}_{j+k}||^2 <= \epsilon^2$$

This means that after s^*, even if one trajectory deviates from the other, as long as the cumulative distance between the two is smaller than a specified threshold (i.e., the deviation is transient), s^* is still considered as a subgoal. In our analysis, we choose ϵ to be 0.06 meter. The subgoals extracted for task A after evaluating all the trajectory pairs are illustrated in Fig. 7.1. The subgoal set can be represented as $s^*(i), i = 1, ..., K$ with K as the total number of identified subgoals.

7.2.2 Subgoal Clustering and Trajectory Segmentation



(a) (b)

Figure 7.2: (a) Subgoal clustering results for task A from the advanced-skilled pilot in the 2-dimensional Euclidean space Y (left) and the original coordinate (right). Different colors represent different clusters and the centers of the clusters are denoted by larger dots. (b) Trajectory segmentation result for task A with the advanced-skilled pilot. Segment cluster 1 is of color red; cluster 2 is of color blue; and cluster 3 is of color green.

It can be observed in Fig. 7.1 that the identified subgoals $s^*(i)$ are not uniformly distributed in the task space. They have a tendency to distribute in clusters, especially near the goal and the corners of the obstacle. For instance, for this particular case as shown in Fig. 7.1, the subgoals are concentrated at three areas: one near the lower-left corner of the obstacle, another one near the upper-right corner of the obstacle and a final one near the goal. This observation fits with the observation made for the guidance behavior in our Dubins' vehicle example. (The only difference is that, for the Dubins' vehicle, as shown in Fig. 5.1, the subgoals are located exactly at the corners of the obstacles.) Furthermore, for the sake of computational efficiency, i.e., it is costly for humans to compute, maintain and update these subgoals, the number of the *intended* subgoals assigned by humans should be limited. So it would be reasonable for us to make the following two hypotheses. First of all, the *observed* subgoals obtained in

Fig. 7.1 are just the expression of some *hidden* or *intended* subgoals. The number of these hidden subgoals should be much smaller than that of the observed ones. Second, the distribution of the observed subgoals around the hidden subgoals obey some nice property. Here we can assume that the distribution can be modeled by a mixture of Gaussians as follows [30]:

$$p(s^*) = \sum_{i=1}^{K^*} z_i \mathcal{N}(s^*|\mu_i, \sigma_i) \tag{7.2}$$

where K^* is the number of hidden subgoals, $\mathcal{N}(s^*|\mu_i, \sigma_i)$ is a normal distribution corresponding to the i-th hidden subgoal with μ_i as its mean and σ_i as its variance, and z_i is the mixing coefficient, which satisfies:

$$0 \leq z_i \leq 1 \text{ and } \sum_{i=1}^{K^*} z_i = 1. \tag{7.3}$$

Given this Gaussian mixture model, the hidden subgoals can then be identified by applying various clustering methods, such as K-means and EM [30].

But before the application of these clustering methods, there are two issues need to be taken care of. First of all, the state s as well as the subgoal $s^*(i)$ is embedded in a state space, which has a dimension that is larger than 2. For instance, for our case here, the state space is four dimensional (2 for the Euclidean coordinate, 1 for the speed and 1 for the heading). Thus, the clustering should be performed not in a 2 dimensional Euclidean space but in a 4 dimensional state space. Speaking of subgoals, this issue implies that we can not just cluster the subgoals based only on their coordinate information. Two subgoals may be not far from each other in spatial sense. But if the velocities at these two subgoals are quite different, it is unlikely that they are generated by the same hidden subgoal. Second, the influence of obstacle on the distribution of subgoals should also to be addressed. If only the distance between subgoals $s^*(i)$ is considered[1] , the identified subgoal clusters will likely cross over obstacles. Again, it is quite unlikely that the subgoals at different sides of an obstacle are generated by the same hidden subgoal. So, all in all, the clustering method should be able to deal with the topology originated from the high dimensional state space and the influence of the obstacle.

[1] This distance is computed in the four dimensional state space.

One way to solve these two issues is by using an isometric mapping technique such as Isomap [151], which involves constructing a lower-dimension manifold to embed the subgoals that preserves the geodesic distance between data points. For our problem, this process can be implemented as follows [152]:

(1) Construct a $K \times K$ matrix D^0:

$$D^0_{i,j} = \begin{cases} ||s^*(i) - s^*(j)|| & \text{if } ||s^*(i) - s^*(j)|| \leq \epsilon^* \\ \infty & \text{otherwise} \end{cases}$$

where ϵ^* is a fixed radius reflecting the local range of connectivity among the points. In our example, we chose 0.2 meters.

(2) Estimate the geodesic distance between all pairs of subgoals via Dijkstra's algorithm as follows:

$$D^m_{i,j} = \begin{cases} D^0_{i,j} & m = 0 \\ \min(D^{m-1}_{i,j}, D^{m-1}_{i,k} + D^{m-1}_{k,j})) & \text{otherwise} \end{cases}$$

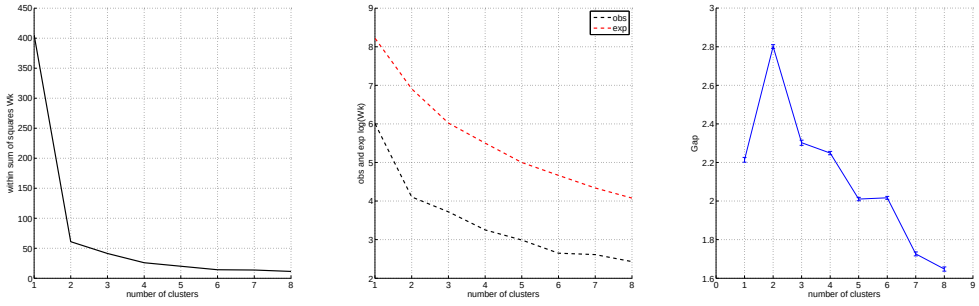
where $0 \leq m \leq K$.

(3) Apply multidimensional scaling (MDS) to matrix D^K in order to embed it in a d-dimensional Euclidean space Y that preserves the pairwise distance in D^K. In our example, we chose d to be 2.

The result of the embedding with respect to the original subgoals in Fig. 7.1 is shown on the left of Fig. 7.2(a). One thing that needs to be brought to attention is that, comparing the subgoals in Fig. 7.1 and Fig. 7.2(a), we see that some of the original subgoals were discarded by the Isomap process. These points are essentially outliers in the sense that, no matter how many iterations are applied in the Dijkstra's algorithm, the shortest distance between these points and the majority of other subgoal points will remain infinite. These outliers are dropped from the analysis and will not be used in the subsequent clustering process.

The clustering of the subgoals was performed using K-means in the Euclidean space Y. The number of clusters was determined via the Gap statistic as shown in Fig 7.3 (with a few human interventions[2]), which is based on how the dispersion measure

[2] Human intervention is needed when the optimal number of cluster is chosen to be 1 by the Gap statistic method, a scenario we try to avoid (unless there is strong evidence to support that really there is only one cluster).



(a) (b) (c)

Figure 7.3: Gap statistic method [9, 10]. (a) Within sum of squares function W_k; (b) Observed (black) and expected (red) values of $\log(W_k)$ and (c) Gap curve, which is the expected curve minus the observed curve. For this particular case, the optimal number of cluster k^* is chosen to be 2 based on the rule that: $k^* = \text{argmin}\{k|G(k) \geq G(k+1) - s'_{k+1}\}$, where $G(.)$ is the gap curve and $s'_K = s_k\sqrt{1+1/N_s}$ with s_k as the standard deviation of $\log(W_k)$ and N_s as the number of simulations. s'_k is shown as error bar in (c).

behaves with the number of clusters. Fig. 7.2(a) shows the clustering results for task A.

Finally, equivalent trajectory classes $[\overleftarrow{s}]$, according to equivalence relationship $\sim_s$ (essentially corresponding to the partitions in Fig. 5.1), can be obtained by dividing each trajectory into segments and labeling those based on their shared subgoals. After this operation, the original measurement data points are augmented by their cluster memberships as follows:

$$[\mathbf{x}^m(i)', \xi^m(i)]', i = 1, ...M_m, m = 1, ..., M$$

with M as the number of trajectory segments, M_m as the number of data points for trajectory segment m, and $\xi^m(i)$ as the membership of data point $\mathbf{x}^m(i)$ with $1 \leq \xi^m(i) \leq K^*$. The trajectory segmentation results for task A, task C and task D are shown in Fig. 7.4. Each color in these figures marks a cluster of trajectories that end at a subgoal belonging to the same subgoal cluster. For instance, in Fig. 7.4(d), all the green trajectory segments all end up at some subgoal belonging to a subgoal cluster that is near to the lower left corner of the obstacle.

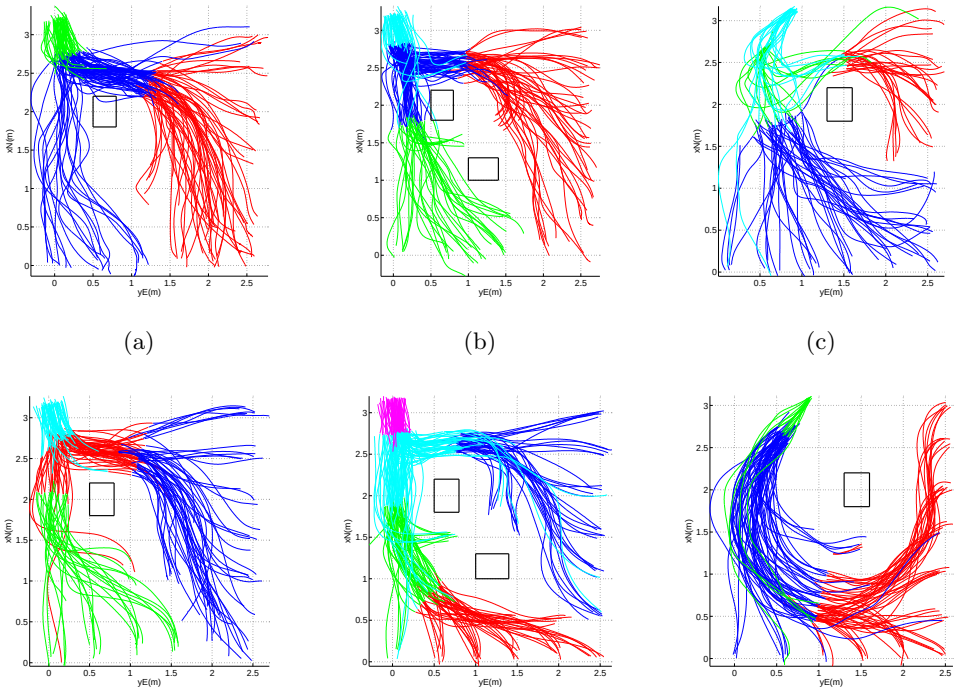
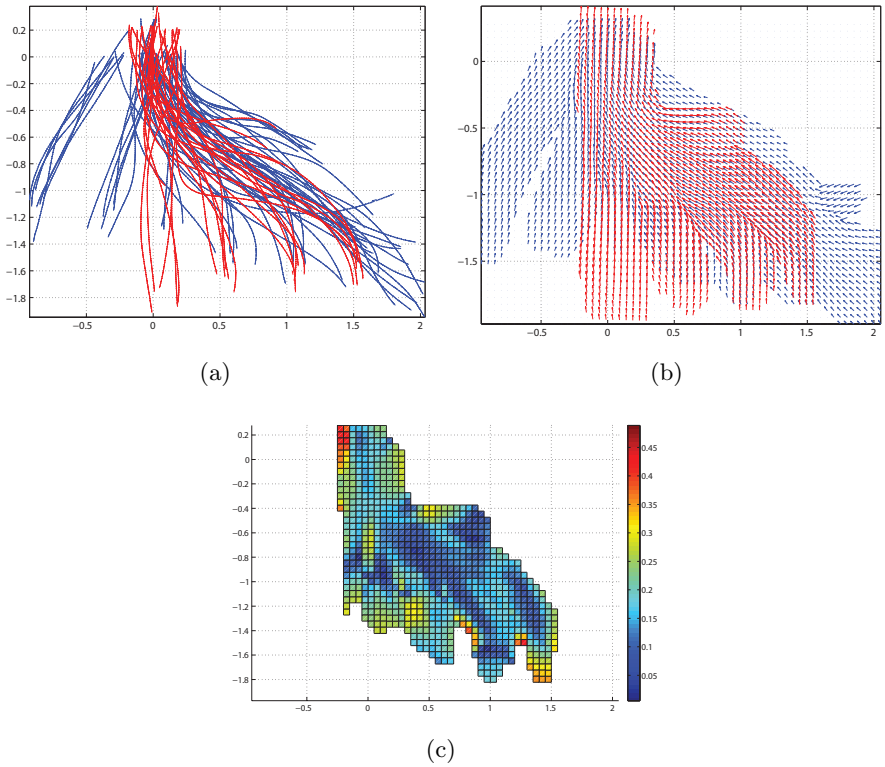


Figure 7.4: Trajectory segmentation results. First row corresponds to the intermediate-skilled pilot and the second row corresponds to the advanced-skilled pilot. Fist column corresponds to task A, second corresponds to task C and last corresponds to task D.

7.3 $\sim_g$ Equivalence Analysis

With the trajectory clusters as shown in Fig. 7.4 at hand, the next step is to find $\sim_g$ equivalences among these clusters. Combined with $\sim_s$ equivalences found in the previous section, we complete extracting interaction patterns from experimental data. In order to find $\sim_g$ equivalences, we first need to find a way of extracting equivalence between two trajectory clusters and then we need to find a way of extracting equivalences among all the trajectory clusters.



(a) (b)

(c)

Figure 7.5: Matching results for cluster 2 and cluster 3 of task A with the advanced-skilled pilot. (a) Superimposed trajectory segments of cluster ξ_1 and cluster $\Psi(m, \xi_2)$ with $\xi_1 = 2$ and $\xi_2 = 3$. (b) Superimposed velocity fields of cluster ξ_1 and cluster $\Psi(m, \xi_2)$ with $\xi_1 = 2$ and $\xi_2 = 3$. (c) The relative difference between correspondence points on a scale from 0 to 1 with 0 corresponds to perfect match.

7.3.1 $\sim_g$ Equivalence Between Two Trajectory Clusters

If cluster ξ_1 is symmetric to another cluster ξ_2 according to the definition of equivalence $\sim_g$ (5.14), they should both belong to a same interaction pattern $\tilde{\pi}_i$. We can evaluate $\sim_g$ equivalence between segment clusters through the following test: for any trajectory $\mathbf{x}^u(t; \mathbf{x}_0)$ from cluster ξ_1, if there exists an action m from the symmetry group M given by (5.19)[3] and a trajectory $\mathbf{x}^{u'}(t'; \mathbf{x}'_0)$ from cluster ξ_2 such that the following result

[3] The dynamics of the RC helicopter used in our experiments, as well as its effect on the environment as given by (5.4), is invariant with respect to actions of the group $SE(2) \times \mathbb{R}$, which is determined by four parameters t_x, t_y, t_z and ψ [4]. However, since we are only concerned with planar motion in this paper, the parameter t_z can be dropped and we end up with the same symmetry group $SE(2)$ as given by (5.19).

holds:

$$\mathbf{x}^{u\prime}(t';\mathbf{x}_0') = \Psi(m, \mathbf{x}^u(t;\mathbf{x}_0)) \tag{7.4}$$

and all the m's are the same, and vice versa, then the two clusters are equivalent in the sense of $\sim_g$.

This test can easily be reformulated as a data fitting or parameter optimization problem. Essentially, we are trying to find a transformation m, which is defined by three parameters t_x, t_y and ψ, such that the following cost is minimized:

$$\frac{1}{Z}\sum_{i=1}^{M_m}\sum_{m=1}^{M} z^m(i)||\Psi(m,\mathbf{x}^m(i)) - C(\Psi(m,\mathbf{x}^m(i)))||^2 \tag{7.5}$$

where

$$z^m(i) = \begin{cases} 1 & \text{if } \xi^m(i) = \xi_1 \text{ and } \delta^m(i) = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$Z = \sum_{i=1}^{M_m}\sum_{m=1}^{M} z^m(i)$$

and $C(\Psi(m,\mathbf{x}^m(i)))$ is the correspondence point within segment cluster ξ_2 for point $\Psi(m,\mathbf{x}^m(i))$. In order to establish a correspondence relationship between $\mathbf{x}_1 := \Psi(m,\mathbf{x}^m(i))$ and $\mathbf{x}_2 := C(\Psi(m,\mathbf{x}^m(i)))$, the points need to satisfy the following two conditions:

(1) $\xi^m(i) = \xi_2$ for $\mathbf{x}_2$, and the distance $||\mathbf{x}_2 - \mathbf{x}_1||$ between points $\mathbf{x}_1$ and $\mathbf{x}_2$ is smallest among all the points satisfying $\xi^m(i) = \xi_2$;

(2) $||\mathbf{x}_2 - \mathbf{x}_1|| \leq l_c$, which says that, if the shortest distance between $\mathbf{x}_1$ and any point in ξ_2 is greater than l_c, then no correspondence point exists in ξ_2 for $\mathbf{x}_1$.

If there exists a correspondence point for $\mathbf{x}_1$, $\delta^m(i) = 1$; otherwise, $\delta^m(i) = 0$.

One issue with using the cost function (7.5) is that the optimization will find a transformation m with small matching errors but will also tend to reduce the size of Z, leading to a smaller region where ξ_1 and ξ_2 overlap. To address this issue, a regularization term βZ is added to (7.5) as follows:

$$\frac{1}{Z}\sum_{i=1}^{M_m}\sum_{m=1}^{M} z^m(i)||\Psi(m,\mathbf{x}^m(i)) - C(\Psi(m,\mathbf{x}^m(i)))||^2 + \beta Z \tag{7.6}$$

where β is negative and takes care of the trade-off between matching error and size of the overlapping region.

The presence of the $C(.)$ term in (7.6) makes it a nonlinear optimization problem. Various methods can be applied. In this example, we used a greedy method. In order to avoid being trapped in local minima, the parameters t_x, t_y and ψ are initialized at different values, and the transformation corresponding to the best cost for all initial values is chosen as the final solution. To decrease the number of initial values needed, the two segment clusters are first aligned based on the directions given by their largest eigenvector and their respective centroids. The greedy method is implemented as follows:

(1) Randomize $m_i^0 := [t_x, t_y, \psi]'$.

(2) For each m_i^0, set $J_i^* = J(m_i^0)$ as computed via (7.6).

(3) For $j = 0, ..., s$, iterate:

(i) Find Δt_x, Δt_y and $\Delta\psi$, such that:

$$J([t_x + \Delta t_x, t_y + \Delta t_y, \psi + \Delta\psi]') < J_i^*$$

(ii) If such Δt_x, Δt_y and $\Delta\psi$ can be found, set:

- $m_i^j := [t_x + \Delta t_x, t_y + \Delta t_y, \psi + \Delta\psi]'$;
- $m_i^* := m_j^i$;
- $J_i^* := J(m_i^*)$.

(iii) Otherwise, end the iteration.

(4) Choose the i with the smallest J_i^*:

$$J^* := \min_i J_i^*$$

and

$$m^* := m_{i^*}^* \text{ with } i^* := \text{argmin}_i J_i^*.$$

Fig. 7.5 shows the matching result for cluster 2 and cluster 3 of task A with the

advanced-skilled pilot[4] . As can be seen, for most of the matched or correspondence points, the relative difference is well below 0.2 on a scale of 0 to 1, with 0 as the perfect matching. Most of the matched points with higher relative difference are those that are at the starting stage of the flight. Since, at the starting stage, the pilot's behavior is transient, and the comparison is done with one velocity at the starting stage (with velocity near zero) and the other one at the coasting stage (with velocity near maximum)[5] , large relative difference should be expected. However, for most of the other points, it is reasonable to say that the two trajectory clusters can be superimposed rather well via a transformation m. Therefore, according to the $\sim_g$ equivalence definition (5.14), they can be considered to belong to the same interaction pattern $[[\overleftarrow{s}]]$.

7.3.2 $\sim_g$ Equivalence Among Multiple Trajectory Clusters

To gain further insight into the organization of the trajectory clusters and interaction patterns, we proceed with an analysis of their mutual equivalence characteristics. The basic idea is the following: if two clusters exhibit strong equivalence, they can be combined into a new cluster, i.e., interaction pattern. This process can be conducted hierarchically as follows:

(1) Initialize the cluster list $L_\xi = [\xi_1, ..., \xi_{K^*}]$.

(2) Apply the greedy method to all $K^* \times (K^* - 1)/2$ cluster pairs in L_ξ.

(3) Choose the pair with the smallest J^*.

(4) Apply the corresponding m^* with respect to the smallest J^* to the first cluster of the pair. Delete the pair from L_ξ and add the new combined cluster to L_ξ.

(5) If there is only one pair left in L_ξ, end; otherwise, go to step (2).

Fig. 7.6(a) and 7.6(b) show the matching results for task A and C, respectively. And Fig. 7.6(c) and 7.6(d) show their corresponding averaged matching errors in the form of

[4] Please refer to paper [108] and paper [112] for a detailed description of the mapping method used to get the velocity fields as shown in Fig. 7.5(b). The velocity fields are obtained by using Gaussian kernel over a spatial grid [108]. For any grid point p, a window of radius R is first chosen. Any data point located outside this window does not contribute to the computation of the value at p. And the relative contribution for each point p_i within the window is weighted by a Gaussian function of its distance to p.

[5] The meanings of starting stage and coasting stage will be explained in Section 7.4.

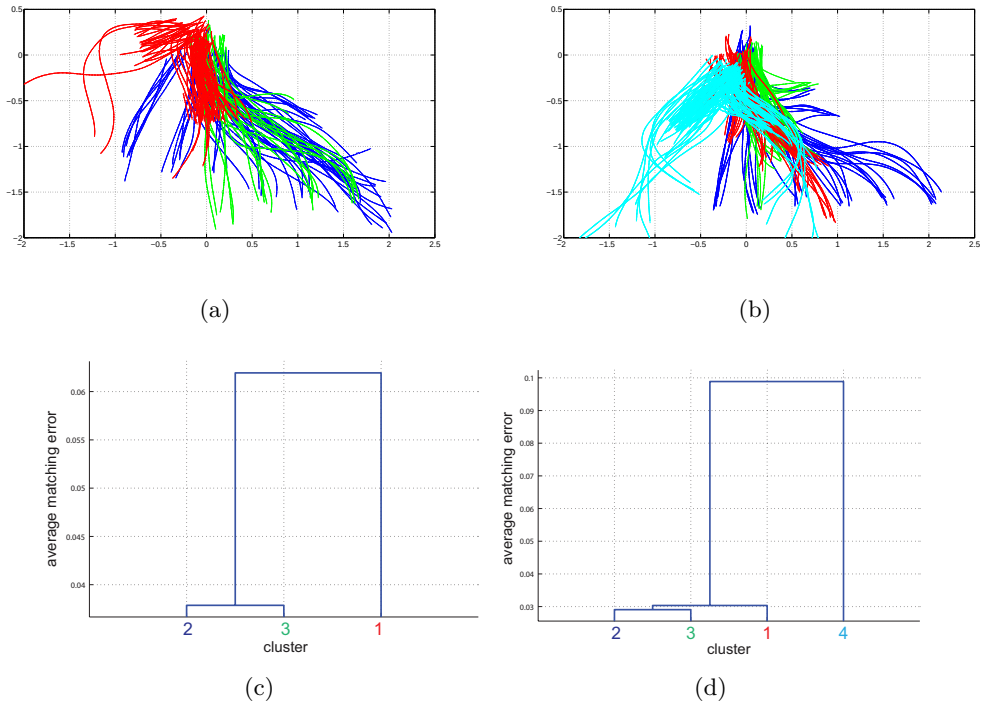


Figure 7.6: (a), (b) Matching results for all clusters of task A and task C, respectively, with the advanced-skilled pilot. (c), (d) Their corresponding averaged matching errors as shown in the form of dendrograms.

dendrograms. To demonstrate how to interpret these dendrograms, let's take Fig. 7.6(d) as an example. The original trajectory clusters are shown in Fig. 7.4(e). Fig. 7.6(d) shows that, in the first iteration, ξ_2 and ξ_3 are first combined into a new cluster ξ_5; and then, in the second iteration, ξ_1 and ξ_5 are combined into another new cluster ξ_6. The average matching errors for these two combining operations are relatively small (smaller than 0.04 m/s). So these three clusters, ξ_2, ξ_3, and ξ_4, can be considered as equivalent to one another with respect to $\sim_g$, or in other words, they belong the same interaction pattern $\tilde{\pi}_1$.

On the other hand, when ξ_1 is matched against ξ_6, their matching error is relatively large (about 0.1 m/s). Therefore, ξ_1 is considered to belong to a different interaction pattern $\tilde{\pi}_2$. The differences among interaction pattern can be explained in terms of the reference state $\mathbf{x}_r$ or subgoal $\mathbf{x}^*$ (the state corresponding to s^*). In fact, the reference

states for ξ_2, ξ_3 and ξ_4 have similar speed (around 0.8 m/s), which explains why we can find a transformations m (a translation, a rotation and/or a reflection) that will match these reference states. However, the speed for the reference state of ξ_3 (around 0.6 m/s) is quite different from those of ξ_2, ξ_3 and ξ_4, which explains why no transformations m can be found that will produce a reasonable match with the latters.

The dendrograms in Fig. 7.6 can also help us to identify the number of interaction patterns for each task and each pilot. Here, let's take task A as shown in Fig. 7.6(c) for instance. The interaction pattern is defined as the equivalence classes with $\sim_g$ and $\sim_s$ as the two equivalences. Trajectory cluster 1, 2 and 3 are obtained by using the concept of $\sim_g$ equivalence. Subsequently, in Fig. 7.6(c), we find that there exists $\sim_s$ equivalence between cluster 2 and 3. In the end, we end up with two interaction patterns, which can be conveniently understood as two skill sets, for task A and the advanced-skilled pilot: one corresponds to approaching a subgoal with a higher speed and includes trajectory cluster 2 and 3; and the other one corresponds to approaching a subgoal with a lower speed and includes trajectory cluster 1. The conclusion we can obtain here is quite similar to the one we obtained for Dubins' vehicle (Section 5.3.3). The only difference is that, for Dubins' vehicle, the velocity is constant and we have only one interaction pattern; but for this particular tasks and pilot, the velocity is not constant and we have two interaction patterns.

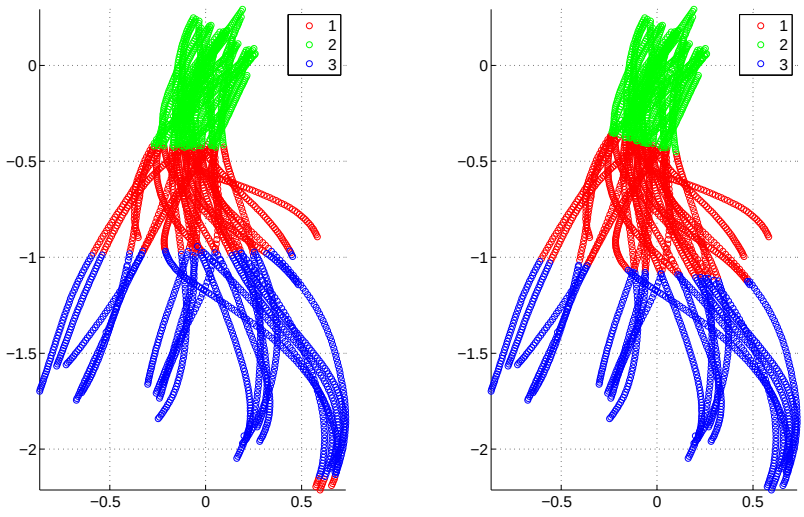
7.4 Dynamical Characteristics of Interaction Patterns

In Section 7.2 and Section 7.3, we are able to extract out the interaction patterns that are inherent to human guidance behavior from the experimental data based on $\sim_s$ and $\sim_g$ equivalence definitions. In this section, we look deeper and investigate the dynamical characteristics of each segment cluster (as a sample interaction pattern) by using Piecewise Affine (PWA) model identification techniques [153].

A PWA system is defined by the following state-space equations:

$$\mathbf{x}(t+1) = A_s\mathbf{x}(t) + B_s\mathbf{u}(t) + d_s, \text{ for } \mathbf{x} \in \mathcal{X}_s, \mathbf{u} \in \mathcal{U}_s \tag{7.7}$$

where $\{\mathcal{X}_s\}_{s=1}^{l_m}$ and $\{\mathcal{U}_s\}_{s=1}^{l_m}$ are polyhedral partitions of $\mathcal{X}$ and $\mathcal{U}$, and d_s is the noise term, which is assumed to be a Gaussian independent identically distributed random



(a) (b)

Figure 7.7: (a) Original data points of a typical segment cluster (advanced-skilled pilot, task A, cluster 3) according to their memberships before discriminant analysis is applied. (b) The data points according to their memberships after discriminant analysis is applied.

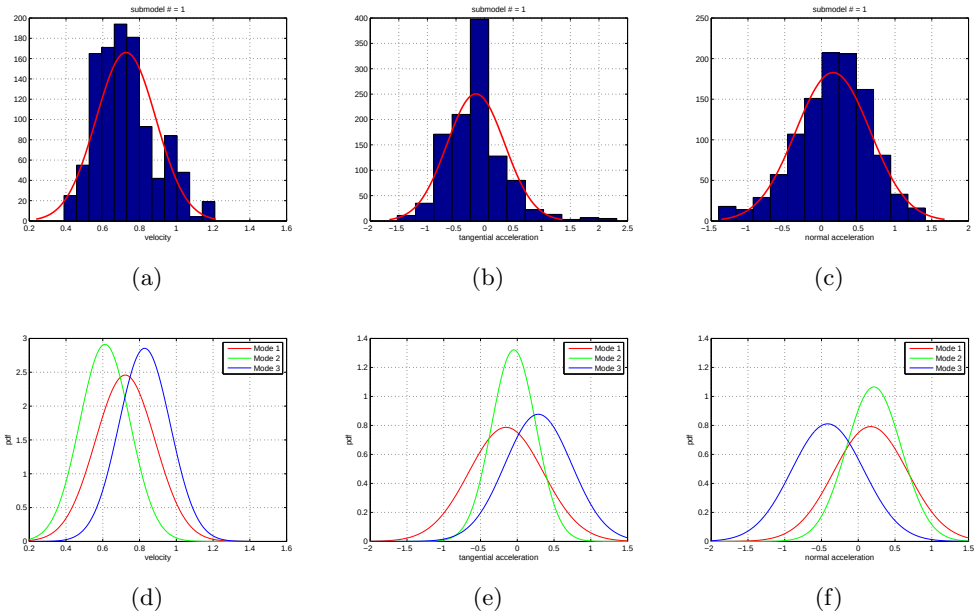
variable with zero mean and variance σ^2. We further assume that each mode of the PWA system can be written as a second-order model[6] :

$$\begin{bmatrix} \dot{x}(t+1) \\ \dot{v}_x(t+1) \\ \dot{y}(t+1) \\ \dot{v}_y(t+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ a_{1x} & a_{2x} & a_{3x} & a_{4x} \\ 0 & 0 & 0 & 1 \\ a_{1y} & a_{2y} & a_{3y} & a_{4y} \end{bmatrix} \begin{bmatrix} x(t) \\ v_x(t) \\ y(t) \\ v_y(t) \end{bmatrix} + \begin{bmatrix} 0 \\ b_x \\ 0 \\ b_y \end{bmatrix} + d \tag{7.8}$$

We found that using a coupled model as opposed to a decoupled model (with a_{3x}, a_{4x}, a_{1y} and a_{2y} all equal to zero) did not significantly affect the identification performance either with respect to the training error or the validation error.

The method proposed in paper [153] was used here to identify the decoupled PWA

[6] Here, we let the state $\mathbf{x}$ be $[x, v_v, y, v_y]'$, which is equivalent to our original definition of $\mathbf{x}$ as $[x, y, v, \psi]'$.



(a) (b) (c)

(d) (e) (f)

Figure 7.8: (a), (b), (c) The histograms of speed, tangential acceleration and normal acceleration (from left to right) and their corresponding fitted normal distributions (shown as red curves) for all the data points belonging to mode 1. (d), (e), (f) The fitted normal distributions of speed, tangential acceleration and normal acceleration (from left to right) for all the three modes.

model underlying each segment cluster $\xi_k, k = 1, ..., K^*$. Let $\mathcal{X}_{\xi_k}$ be given as

$$\mathcal{X}_{\xi_k} := \{\mathbf{x}^m(i) | \xi^m(i) = \xi_k, i = 1, ...M_m, m = 1, ..., M\}$$

which contains all the data points belonging to ξ_k, the vector of coefficients θ as

$$\theta := [a_{1x}, a_{2x}, b_x, a_{1y}, a_{2y}, b_y]' \tag{7.9}$$

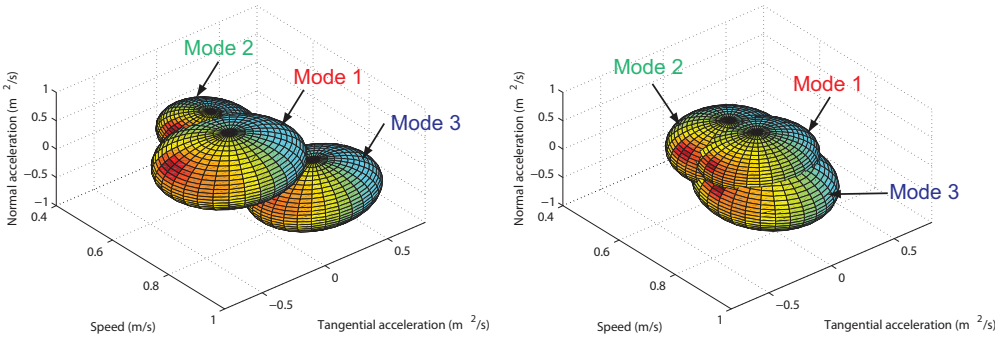
the vector of regressors $\mathbf{x}(k)$ as

$$\mathbf{x}(k) := [x(k), v_x(k), y(k), v_y(k)]' \tag{7.10}$$

and the vector of output samples $\mathbf{y}(k)$ as

$$\mathbf{y}(k) := [x(k+1), v_x(k+1), y(k+1), v_y(k+1)]' \tag{7.11}$$

All the data inside $\mathcal{X}_{\xi_k}$ is normalized before the identification procedures. The PWA identification follows the steps described in [153]:



(a) (b)

Figure 7.9: The fitted normal distributions of speed, tangential acceleration and normal acceleration as depicted with ellipsoids. The centers of these ellipsoids are the means of speed, tangential acceleration and normal acceleration. And the axes of these ellipsoids are the corresponding standard deviations. (a) shows a typical distribution for the advanced-skilled pilot and (b) shows a typical distribution for the intermediate-skilled pilot. Please refer to Figs. 7.8(a)-7.8(f) for some example histograms and their corresponding fitted normal distributions.

(1) For each $\mathbf{x}(i) \in \mathcal{X}_{\xi_k}$, find the $c-1$ nearest data points within $\mathcal{X}_{\xi_k}$. Together, they make up a point set $\mathcal{C}_i$ of c data.

(2) Estimate the vector coefficients θ_i for each $\mathcal{C}_i$ by using the linear regression method [30]:

$$\theta_i = (\Phi_i' \Phi_i)^{-1} \Phi_i' Y_i$$

where the output matrix Y_i is

$$Y_i = \left[\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_c\right]'$$

and the design matrix Φ_i is

$$\Phi_i = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & ... & \mathbf{x}_c \\ 1 & 1 & ... & 1 \end{bmatrix}'$$

The vectors $\mathbf{x}_i$ and $\mathbf{y}_i$ are in the form of (7.10) and (7.11), respectively. The covariance matrix of θ_i can be computed as:

$$V_i = \frac{1}{c-3}(Y_i - \theta_i \Phi_i)'(Y_i - \theta_i \Phi_i)(\Phi_i' \Phi_i)^{-1}$$

and the scatter matrix can be computed as:

$$Q_i = \sum_{\mathbf{x} \in \mathcal{C}_i} (\mathbf{x} - c_i)(\mathbf{x} - c_i)'$$

where c_i is the center of data set $\mathcal{C}_i$

$$c_i = \frac{1}{c} \sum_{\mathbf{x} \in \mathcal{C}_i} \mathbf{x}(\mathbf{x} - c_i)(\mathbf{x} - c_i)'$$

Choose the feature vector to be $\varsigma_i := [\theta_i', c_i']'$. Its corresponding variance is:

$$R_i = \begin{bmatrix} V_i & 0 \\ 0 & Q_i \end{bmatrix}$$

(3) Apply weighted K-means method to cluster the feature vectors in the feature space. The clustering cost functional is:

$$J(\{D_j\}_{j=1}^3, \{\mu_j\}_{j=1}^3) = \sum_{j=1}^{3} \sum \varsigma_i \in D_i ||\varsigma_i - \mu_i||^2_{R_i^{-1}}$$

(4) Finally, apply linear discriminant analysis (LDA) [30] to determine the partitions $\mathcal{X}_s, s = 1, ..., 3$ and apply linear regression method (2) to estimate the coefficients for each model $\theta_s, s = 1, ..., 3$.

The application of LDA here relies on the assumption that all the partitions $\mathcal{X}_s$ corresponding to different modes are polyhedral and convex. Following this then, the identification of the partitions reduces to the identification of the hyperplanes separating these partitions. And the choice of three as the number of modes is based on the observation that there is a significant improvement in fitting performance (with respect to validation errors) when the mode number is increased from two to three, while there is no significant improvement when the number is further increased.

Figs. 7.7(a)-7.7(b) show the results of the PWA identification for one of the segment clusters obtained from the advanced-skilled pilot data. From these two sub-figures, we see that the three modes provide a logical delineation of the dynamics. In particular, we see that the three modes map very consistently to three different operating stages: a starting stage (mode 3), a coasting stage (mode 1) and an approaching stage (mode 2).

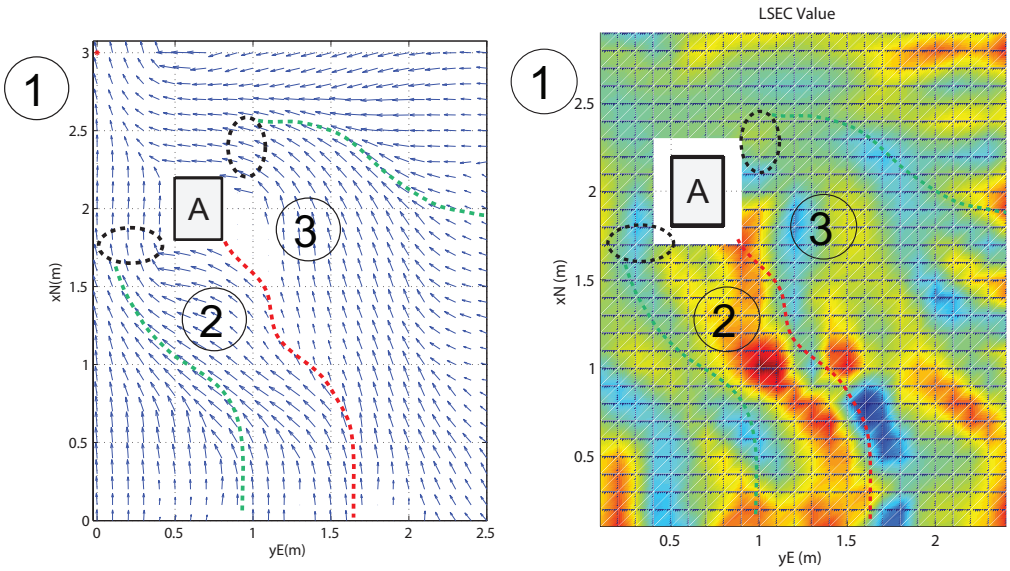
Figs. 7.8(a)-7.8(c) show the histograms and the fitted normal distributions of the speed, tangential acceleration and normal acceleration for just one mode. And Figs. 7.8(d)-7.8(f) show the fitted normal distributions for three modes. Notice that the three identified modes for this particular cluster have distinctive kinematic characteristics.

The distinctiveness of each mode can also be seen by plotting the fitted distributions as ellipsoids, as shown in Fig. 7.9(a). Other clusters and tasks from the advanced-skilled pilot show similar modes with distinctive kinematic characteristics. Fig. 7.9(b) illustrate the fitted distributions for a typical segment cluster from the intermediate pilot. It shows that although the intermediate-skilled pilot also operates according to three modes, the modes tend to overlap much more and are therefore less distinct. Some of this has to do with the overall larger variance in the intermediate-skilled pilot's modes compared to those of the advanced. We can conclude from these two sub-figures that, the advanced-skilled pilot is able to establish distinct operating modes; while the intermediate-skilled pilot is still struggling to establish the boundaries between different modes as well as maintain consistency within a single mode.

It is worth pointing out that the technique used in this section to analyze the dynamical characteristics of interaction patterns can also be used to evaluate the skill levels of different pilots. According to the conclusions obtained in this section and Section 6.5, it has been shown that the maturity of a pilot's skill should be reflected at two different levels. At the lower level, the pilot should be able to generate stable (spatially and temporally consistent) behaviors, which imply not only well-established interaction patterns or skill sets but also well-established operating modes within each interaction pattern. At the higher level, the pilot should be able to use these interaction patterns and modes to organize his/her behavior; and the switchings between these modes or interaction patterns should be clear-cut[7] . These two requirements can be evaluated together by the application of the ellipsoids as shown in Fig. 7.9. The requirement at the lower level means that the axes (corresponding to variances) of these ellipsoids should be short. And the requirement at the higher level means that the overlapping between the ellipsoids should be small.

[7] The organization and switchings will be analyzed in the next section.

7.5 Analysis of Organizational Principles

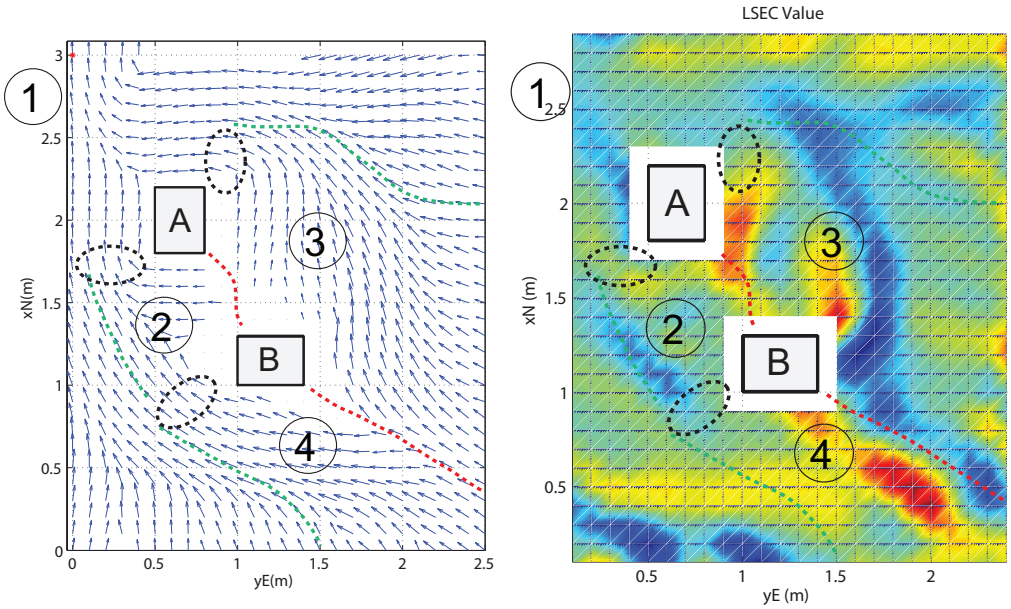


(a) (b)

Figure 7.10: (a) Velocity field and (b) LSEC field for task A with the advanced-skilled pilot. Black lines mark the locations of subgoals, red lines mark the locations of repelling manifolds (ridge lines), and green lines mark the locations of attracting manifolds (valley lines).

In the previous three sections, we are able to extract out the interaction patterns and also operating modes within each interaction pattern from the experimental data. When performing these analysis, the assumptions are kept at a minimal level. The structures, i.e., the interaction patterns and the modes, are not prescribed by humans like model-based approaches as described in Section 1.2.1. Instead, they emerge from the guidance behavior. The methods we developed to obtain these structures are *data-driven*, the structures as well as the details of the structures are derived from the experimental data not from some a priori knowledge. The data-driven nature of our methods is the reason that our framework is grounded.

In this section, given the collection of templates or repertories or interaction patterns, we answer the question of how the pilots organize their behaviors in order to complete a guidance task. Metaphorically, it is like we have all the interaction patterns



(a) (b)

Figure 7.11: (a) Velocity field and (b) LSEC field for task C with the advanced-skilled pilot. For the meanings of black lines, red lines and green lines, please refer to the captain of Fig. 7.10.

as puzzle pieces, and we want to figure out how the pilots put all these pieces together for the completion of guidance tasks. In principle, we should be able to identify the organizational principles by following the same data-driven method that derives the interaction patterns. For instance, we can first come up some biases of the models or representations regarding the organizational principles. Just like the definition of interaction patterns or modes, these biases should be inherent to guidance behavior, should be able to be described by mathematical languages, and should be compatible with the knowledge we have about human guidance. And then learning algorithms can be used to train these models or representations based on the experimental data [30, 24, 25, 28][8] . However, such an operation would require us to have a large amount of scenarios in order to train the algorithm[9] . Since we only have four scenarios as shown in Fig. 6.4, in

[8] This point will be further elaborated on in Section 7.6.2.

[9] Notice that here we are trying to learn some meta model, i.e., a model can be used to explain a wide range of scenarios. Interaction patterns might be shared by these scenarios but it is the organization of these patterns for the fulfillment of these scenarios that we are interested here, not the pattern

this section, we follow a different path. The key here is to figure out where the pattern switching happens and why the switching happens at that place. First of all, based on the experimental data, we observe the switchings; second, based on the observations, we make some hypotheses about the locations and the reason of the switching surfaces; finally, in order to check whether these hypotheses are reasonable or not, we use them to predict the switching locations and compare the predicted results with the experimental results.

7.5.1 Observation of Switching Surfaces

In the previous section, we use PWA system together with discriminant analysis to identify the partitions, including the boundaries, within each interaction pattern. The boundaries mark the switches from one mode to the next. In this subsection, in order to observe where the switching between different interaction happens, it would be helpful to find a way to characterized these switching surfaces first.

There are two ways we can characterize the boundaries or switching surfaces among these patterns.

(1) Attracting and repelling manifolds or Lagrangian coherent structures (LCSs) [154] of the velocity field. Qualitatively speaking, if we trace two points at either side of an repelling manifold forward in time, they will eventually diverge from each other; similarly, if we trace two points at either side of an attracting manifold backward in time, they will eventually diverge from each other [154]. These LCSs or manifolds can be extracted as the local maxima (ridges) of a finite-time Lyapunov exponents (FTLE) field, which in turn can be computed from the velocity field: repelling(attracting) manifolds as the ridges of the forward(backward)-time FTLE field. Fig. 7.10(a) and Fig. 7.11(a) show the computed velocity fields of task A and task C and the extracted attracting manifolds (green) and repelling manifolds (red), respectively. These manifolds are also marked in Fig. 5.1 for the Dubins' vehicle.

(2) Principal curvatures [155, 156] of the time-to-go (TTG) field. For a TTG function $T(x)$, its principal curvature at a point x can be defined as the extremal of its

themselves, which can be learned from the experimental data we already have, as shown in the previous section.

normal curvature. If point x has the largest maximum (positive) principal curvature in a neighborhood, it is called a ridge point. A ridge line can then be formed by connecting all the neighboring ridge points together. A valley line can be defined similarly, except that it connects points that assume the smallest minimum (negative) principal curvature. The principal curvatures can be extracted from a level-set extrinsic curvature (LSEC) field. A ridge line corresponds to a continuous region with values higher than a specified threshold and a valley line corresponds to a region with values lower than another threshold. Fig. 7.10(b) and Fig. 7.11(b) show the computed LSEC fields of task A and task C and the extracted valley lines (green) and ridge lines (red), respectively.

By comparing Fig. 7.10(a) to Fig. 7.11(b) with trajectory segments in Fig. 7.4(d) and Fig. 7.4(e), it can be seen that the extracted boundaries based on both of these two methods fit well with the boundaries of the trajectory segments or interaction patterns. One important switching surface that connects interaction patterns is the attracting manifold or valley. And it is closely related to the concept of subgoal[10] .

7.5.2 Hypotheses Regarding Organization of Patterns

In the previoius subsection, we showed that the switchings between different interaction patterns happen at two different kind of surfaces: repelling manifolds (ridges) and attracting manifolds (valleys). Then the key question is, based on these surfaces of different characteristics, for the purpose of organizing their guidance behavior, what are the principles implemented by humans to decide these surfaces?

Due to the limited number of scenarios, here we are only able to give one possible explanation. As stated in the description of our hierarchical guidance architecture (Fig. 1.6), one important benefit of using it is that, for higher-level operations, only certain features or information regarding the interaction patterns are needed. So here we assume that only a simple function for TTG (or velocity field) is retained in the higher level for each pattern. And then these TTGs are pieced together deliberately according to the optimality principle. Once the the locations of the interaction pattern

[10] By inspecting the concept of subgoal, which marks the point where trajectories that are equivalence to each other begin to converge, it is no wonder that subgoals coincide with attracting manifolds or valleys.

boundaries and subgoals are determined, the details regarding the patterns can finally be filled in at the lower level.

The following hypotheses can be made based on the above augments and our proposed hierarchical guidance model as shown in Fig. 1.6:

- In the bottom-up direction, only partial information regarding the interaction patterns are used in higher-level control. Here we assume that only the TTG information of these patterns is used. In another word, here we assume that, when humans are organizing their guidance behaviors in the higher level, they will only consider the TTG; other information, such as the exact functional form of each interaction pattern, can be ignored[11] . Furthermore, these features can be learned from experimental data. This fits the 'reusability' of the patterns. It is arrived at based on the results we obtained from step 1 and 2 (Section 7.2 and Section 7.2).

- In the top-down direction, human obeys optimality principles. Optimality principles are widely used in the analysis of human behavior, especially in motor control [45]. One main caveat of their application, though, is the computational complexity faced by humans, if optimal solutions are expected (the well-known "curse of dimensionality"). Our hierarchical model provides one way of handling this complexity issue. We hypothesize that what the brain actually organizes is the interaction patterns instead of the detailed low-level dynamics. In such a way, even optimality principles are still obeyed by humans, the computational complexity is still manageable. Our hypotheses are in line with the idea of *motor primitives*[12] , which are used to explain the spatial and temporal consistency existing in animal's motor behavior as well as animal's ability to tackle complexity involved in motor control problems.

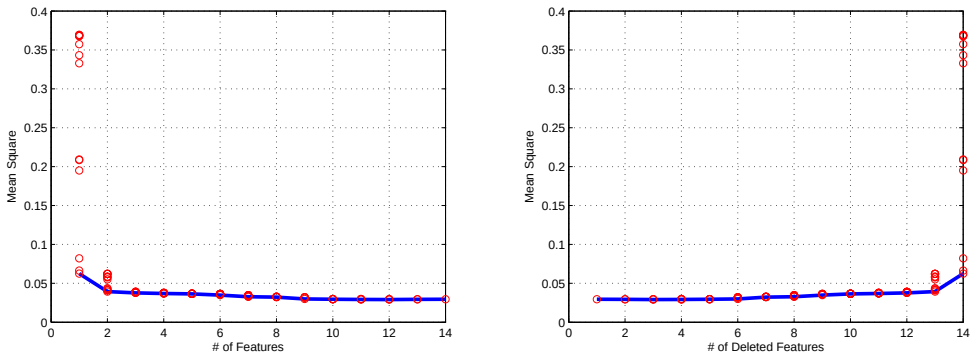
[11] You can think of this in the sense of hierarchical control, where only partial information is passed between the higher and lower modules. Or you can think of this as similar to ecological guidance [34] or behavioral dynamics [35], where partial information is sufficient for some specific tasks.

[12] The details regarding motor primitives can be found in Section 1.2.1 and papers [52, 157].

7.5.3 Hypotheses Validation

In this subsection, we show two pieces of evidences to support our hypotheses: one is by using the hypotheses to predict guidance behavior and the other one is by correlating the hypotheses with pilot's gaze locations.

Prediction of the Interaction Patterns Organization



(a) (b)

Figure 7.12: (a) Forward and (b) backward feature selection results. In both of these figures, blue lines indicate the minimum MSE, while the red circles indicate all the possible MSEs.

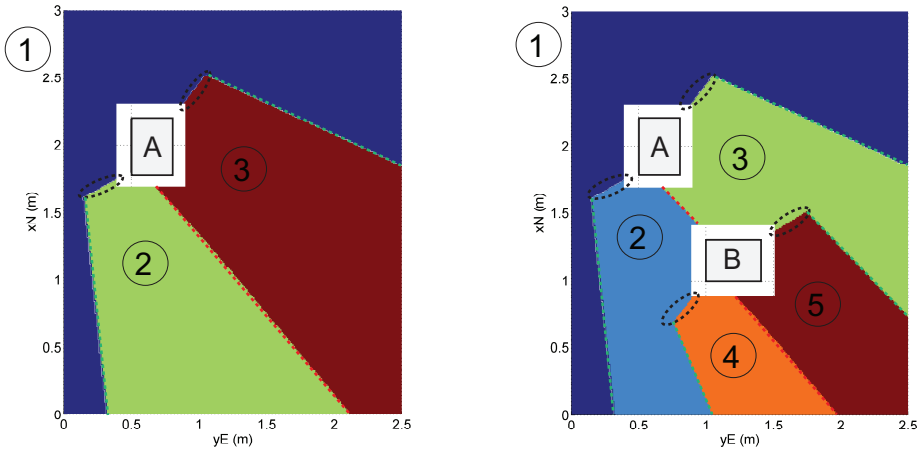
In order to validate these hypotheses, in this sub-subsection, they are going to be used to predict the interaction pattern organization for some particular tasks. The predicted behaviors are then compared with those obtained from experimental data.

The partial information about the interaction patterns chosen here is the time-to-go (TTG). It gives the time t needed to travel to a designated subgoal $\mathbf{x}^*$ for a particular interaction pattern given the current state $\mathbf{x}$:

$$t = G_{\mathbf{x}^*}(\mathbf{x}) \tag{7.12}$$

TTG is selected as the information passed between the low and high levels mainly because of its close relationship with the concept of tau[13] . Note that TTG could be

[13] Please refer to Section 1.2.2 and papers [2, 59, 60] for details about tau.



(a) (b)

Figure 7.13: (a), (b) The predicted partitions for task A and task C. In (a)-(b), black lines mark the locations of subgoals, red lines mark the locations of repelling manifolds (ridge lines), and green lines mark the locations of attracting manifolds (valley lines).

replaced with other information such as the interaction pattern's velocity field without compromising the outcome. But it should be noted that TTG is a more efficient representation than velocity field.

The shapes of TTG, i.e., $G_{x_g}(.)$, can be learned from experimental data. Forward method [10] is used to select the appropriate set of features which is sufficient to characterize the shape of TTG. The procedure is shown is shown in Fig. 7.12(a): we start with no variable; at each step, the feature resulting the most decrease in mean-squared-error (MSE) is added; the algorithm will be ended once all the features are added. A backward selection method can also be applied as shown in Fig. 7.12(b). These methods are applied to the whole data set, i.e., all the patterns. To prevent over-fitting, a fivefold cross validation is used to select the best feature that should be added or deleted at each step [12, 10].

It can be seen from Fig. 7.12 that using three features can achieve approximately the same performance as using the whole feature set, which has 14 features. The three features chosen by the forward method and the backward method for the advanced-skilled pilot are the same. They are: ρ, the distance from the current position to the

subgoal, v_r, velocity at the subgoal, and, $v_r\phi$, the product of v_r and ϕ the angle between the line connecting current point to the goal and the velocity direction at the subgoal. These three features can then be used to learn the functional form of the TTG from the experimental data. The selection result means that three independent features, ρ, v_r and ϕ, are sufficient to characterize the shape of the TTG. ρ and ψ determine the location of the agent with respect to the subgoal; while v_r determines the assigned arrival speed at the subgoal.

Once the functional form is obtained, a wavefront method based on optimality principles [158, 138] can then be used to derive the partitions. It works as follows:

- The subgoals are determined to be the locations where the wavefront defined by the learned TTG function meet the vertices of obstacles. Once a subgoal is initiated, a new wavefront is created at this new subgoal.
- The repelling manifolds are the places where two wavefronts originated either from a goal or a subgoal meet each other.
- The attracting manifolds are originated from subgoals and their directions are determined by the gradient of the wavefront.

The organization of patterns generated according to the above wavefront method for one task is shown in Fig. 7.13(b). By comparing the predicted switching locations with the empirical result as shown in Fig. 7.11, our wavefront method here predicts the locations of the subgoals and the pattern boundaries fairly well. The comparison between the prediction and the experimental data partially confirms our hierarchical organization hypotheses.

Correlation with Gaze Tracking

In our hypotheses, the switching between different interaction patterns happen at subgoals. If these subgoals do play a significant role in the behavioral organization, we should be able to find correspondence between the locations of subgoals with human's decision making and perception mechanisms. This can indeed be confirmed by looking at pilot's gaze locations, which are related to human's decision making and perception, during the guidance tasks as shown in Fig. 7.14. It can be observed that: the

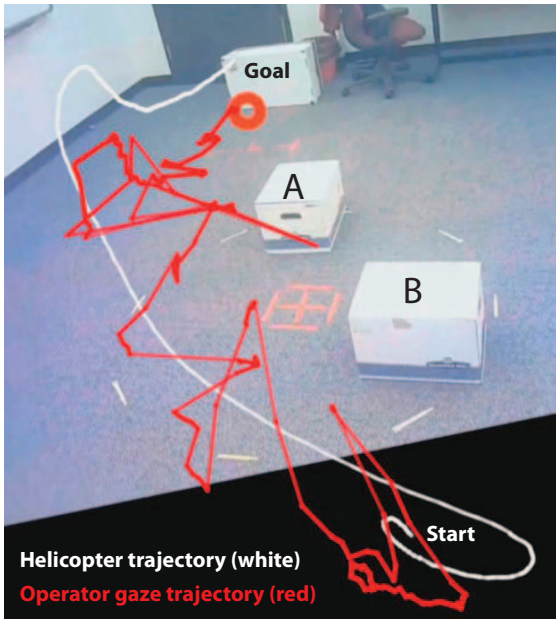


Figure 7.14: Pilot's gaze trajectory (red) and helicopter trajectory (white) superposed to video from field of view camera.

pilot spends most of his time attending to the helicopter; but occasionally the pilot switches his attention (which is correlated with his gaze location) to some locations that are ahead of the helicopter; and these locations are exactly where the subgoals are; once the helicopter reaches a subgoal, the next time the pilot switches his attention, he switches it to the next subgoal. For the task as shown in Fig. 7.14, the places where the pilot switches his attention to are the lower-left corners of obstacle A and B and also the destination. These are where the corresponding subgoals are as shown in Fig. 7.11.

7.6 Discussion

7.6.1 Implications for Human Guidance

Our formal framework provides a new perspective to understand human guidance behavior. In particular, the existence of interaction patterns guarantees that guidance behavior is consistent over time and has stereotypical characteristics in the sense that the same form of behavior will reoccur in other situations. For example, in the case

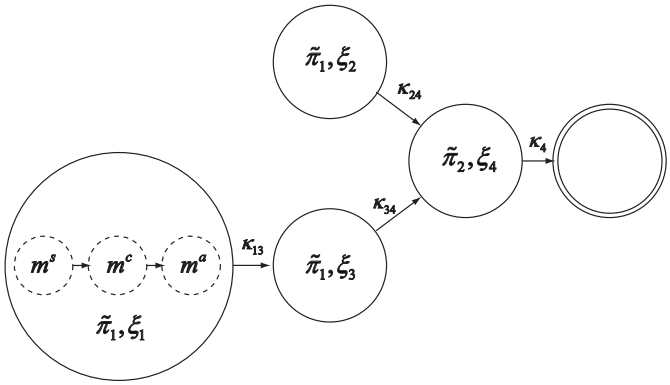


Figure 7.15: Automaton representation of the advanced-skilled pilot's guidance behavior for task C as shown in Fig. 7.4(e). Double circle denotes the final (or accepting) state. Only the mode transitions inside cluster ξ_1 are shown.

of two pilots operating two similar aircraft to complete two guidance tasks, no matter the layout of the environment and even the specific nature of the tasks, as long as both pilots employ the same information extraction law $h(.)$ and the same control law $k(.,.)$, they will exhibit similar guidance behaviors (The behavior includes both the dynamics of the aircrafts as described by (2.1) and their effects on the environment as described by the environment states in (5.4))[14] . This type of fundamental property has important practical implications both for human and autonomous systems. It also provides the key elements needed for a formal modeling language of guidance behavior. This section gives a brief description of some of these implications. They will be treated in detail in future papers.

The consistency of guidance behavior implies that there is no need for humans to generate a new information extraction and control strategy each time they face a new guidance task. Provided with an appropriate abstraction that is consistent with the invariances existing in the agent-environment interactions, i.e., interaction patterns, they can employ a similar strategy for a variety of different tasks.

The other implication is that the overall guidance behavior can be organized systematically. Humans or autonomous agents can utilize this natural organization in order to reduce the mental or computational burden needed when planning or reasoning.

[14] This obviously also assumes the atmosphere is isotropic and homogeneous and the gravitational field is constant.

The general features the guidance behavior are determined by the interaction patterns. These patterns are described in the library $\tilde{\Pi}$ (see (5.18)), which is parsimonious compared to the entire $\mathcal{X} \times \mathcal{U} \times \mathcal{E}$ but at the same time does not discard any significant component of the guidance behavior. Each letter drawn from this library (which is assumed to be of finite cardinality) corresponds to an interaction pattern $\tilde{\pi}$. And a finite string of such letters is called a word. The set of all such words, including the empty one $\tilde{\pi}_{\text{null}}$, can be denoted as $\tilde{\Pi}^\star$. Now guidance behavior can be organized as words from $\tilde{\Pi}^\star$.

Figure 7.15 depicts the entire guidance behavior of the advanced-skilled pilot as an automaton. Its descriptive complexity is much smaller than that of the original trajectory data shown in Fig. 7.4(e)[15] . The discrete states of this automaton are identified with individual segment clusters, which are samples of interaction patterns from $\tilde{\Pi}$. State-to-state transitions occur in response to some environmentally driven interrupt conditions κ, which are triggered at subgoals. With such an automaton representation, any trajectory starting from segment cluster ξ_1 can simply be generated as a word $\tilde{\pi}_1, \tilde{\pi}_1, \tilde{\pi}_2, \tilde{\pi}_{\text{null}}$ with the first letter corresponds to ξ_1 (shown as red in Fig. 7.4(e)), a sample of $\tilde{\pi}_1$, the second letter corresponds to ξ_3 (shown as green in Fig. 7.4(e)), also a sample of $\tilde{\pi}_1$, the third letter corresponds to ξ_4 (shown as blue in Fig. 7.4(e)), a sample of $\tilde{\pi}_2$, and the final latter corresponds to the final state or the goal.

Formal Model of Human Guidance Behavior

The concept of interaction patterns and formal system that follows from it can be used as foundation for an expressive guidance language similar to the motion description language (MDL) [74, 75, 159] or symbolic planning and control (SPL) [5]. It would work as follows. A letter in $\tilde{\Pi}$ is first paired with its corresponding subgoal $\mathbf{x}^*$. The information extraction law $h(.)$ and control law $k(.,.)$ encapsulate the low-level details regarding the agent dynamics and the agent-environment interactions. They are only accessed when necessary. The letters are then used to form a word $\tilde{\pi}_{i_1}, ..., \tilde{\pi}_{i_q}$ in $\tilde{\Pi}^\star$.

[15] Please notice that the automaton itself is not complete for the description of the entire guidance behavior. It needs to be augmented with details regarding each interaction pattern. Since, for this particular case, as described in Section 7.3, there are only two interaction patterns, the automaton together with these two interaction patterns is still far more parsimonious than the original data.

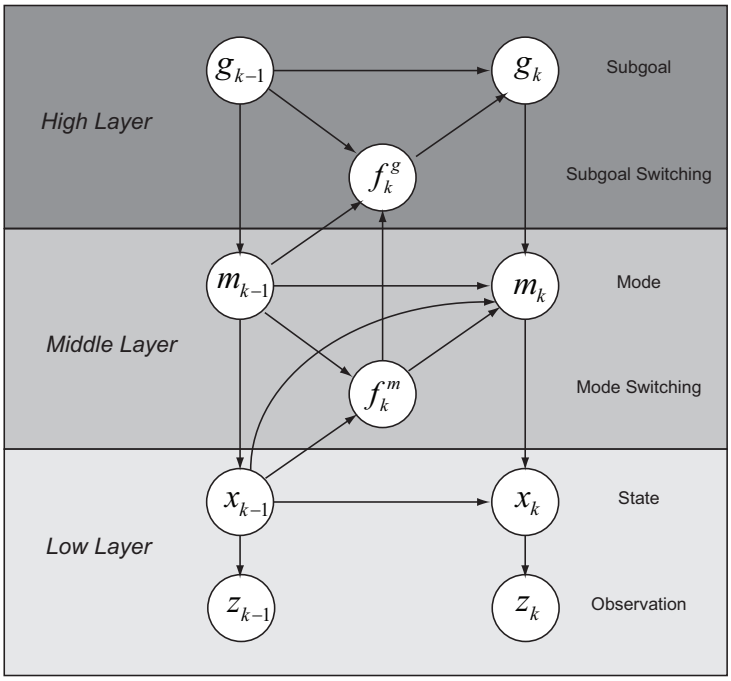


Figure 7.16: Hierarchical hidden Markov model (HHMM) of human guidance behavior.

This word can be augmented into a symbol string as:

$$(\tilde{\pi}_{i_1}, \mathbf{x}^*_{i_1}, h_{i_1}, k_{i_1}, \kappa_{i_1}), ..., (\tilde{\pi}_{i_q}, \mathbf{x}^*_{i_q}, h_{i_q}, k_{i_q}, \kappa_{i_q}) \tag{7.13}$$

The κs play the role of interrupts: every time the agent reaches a subgoal $\mathbf{x}^*_{i_j}$, κ_{i_j} is set to one and $\kappa_{i_{j+1}}$ is set to zero and a new behavior is initiated by setting the new subgoal $\mathbf{x}^*_{i_{j+1}}$, new information extraction law $h_{i_{j+1}}(.)$ and new control law $k_{i_{j+1}}(.,.)$. The same information extraction law $h_{i_{j+1}}(.)$ and control law $k_{i_{j+1}}(.,.)$ are implemented repeated as long as $\kappa_{i_{j+1}}$ remains zero, i.e., as long as the new subgoal $\mathbf{x}^*_{i_{j+1}}$ has not been reached.

It is expected that such a hybrid formulation can be generalized to more complex environments. The cardinality of $\tilde{\Pi}$, which can be seen as the repertoire of human guidance capabilities, should roughly stay the same; it is only the automaton representation corresponding to high-level planning that will become more complex. However, since the organization of interaction patterns is systematic, the complexity at high level is

still manageable.

Based on the results of the analysis of the dynamical characteristics in Section 7.4, it can be easily observed that each augmented letter of an interaction pattern string (7.13) can in turn be written as a mode string:

$$(m^s_{i_j}, h^s_{i_j}, k^s_{i_j}, \kappa^s_{i_j}), (m^c_{i_j}, h^c_{i_j}, k^c_{i_j}, \kappa^c_{i_j}), (m^a_{i_j}, \mathbf{x}^*_{i_j}, h^a_{i_j}, k^a_{i_j}, \kappa^a_{i_j})$$

where m's are the identified modes and the superscripts s, c, a stand for starting mode, coasting mode and approaching mode, respectively. Among those modes, only the approaching mode is truly an interaction pattern, since it is related to a subgoal $\mathbf{x}^*_{i_j}$; the other two modes are transient behaviors needed to accommodate for the 'boundary conditions' associated with the interaction patterns. Most importantly, the controls $k^{\cdot}_{i_j}$, as well as the interrupt conditions $\kappa^{\cdot}_{i_j}$, can be learned from trials and the former constitute the building elements for the k_{i_j} in (7.13).

Combined, these insights support a hierarchical model of human guidance behavior. An important way this model distinguishes itself from the models used in MDL and SPL (which construct the models in a top-down manner[16]) is that, it comes out in a bottom-up manner; it is built upon the invariances that are inherent to the agent-environment interaction. The model also clearly delineates and accounts for the key components of the hierarchy. It can be used to describe a pilot's guidance behavior, but could also be used to predict the behavior as the environment, mission or the operating system changes. The overall system can be formalized using a hierarchical hidden Markov model (HHMM) as shown in Fig. 7.16[17] . For the example in this paper, state $\mathbf{x}$ is taken as $[x, y, v, \psi]'$ and the measurement is taken to be the same as $\mathbf{x}$. Mode m can take on three values: m^s, m^c and m^e. The edges or dependencies among m and x at different times, along with the Boolean mode switching node f^m, are learned in step 3 (Section 7.4). Together, they correspond to the PWA systems learned from the interaction patterns. Similarly, in principle, the edges among subgoals g and the Boolean goal switching mode f^g can also be learned from experimental data in step 4 (Section 7.5).

What this hierarchical modes says is that, facing a complex guidance task, a pilot

[16] Issues with the top-down methods are discussed in Section 1.2.1 and Section 1.3.

[17] Please notice that the automaton representation as shown in Fig. 7.15 can be seen as the flat model corresponding to the HHMM model.

decomposes the task into a sequence of sub-tasks. Each of these sub-tasks belongs to and can be characterized by some interaction patterns or skill sets. And the switchings between these patterns happen at subgoals, where the pilot spends special attention to while flying. And the organization of the pilot's guidance behavior can be implemented in three levels. At the high level, the pilots assigns a sequence of subgoals; at the middle level, in order to fly from the current state to the assigned subgoal, the pilot follows a sequence of modes; and at the low level, within each mode, pilot's guidance behavior can be generated by a linear dynamical system.

Behavorial modeling has been used in human factors to describe human behavior as well as predict their performance following changes in the environment, mission or system (see e.g. [160, 161]). These models can be used to formally verify the human-automation interaction systems [162] or to design effective human-machine interfaces [100]. Our modeling philosophy is to limit the number of assumptions needed to determine these models. In the work presented in this chapter, the hierarchical structure and its components are mainly determined from experimental data. Such a practice would help achieve high-fidelity and comprehensive models without requiring human excessive iterations.

7.6.2 Implications for Engineering Applications

The framework and the results presented in this dissertation can potentially be used to the design of autonomous systems, especially in conjunction with the learning algorithms as described in Section 1.3. For instance, in the current state-of-the-arts learning from demonstration (LFD) algorithms, as described in Section 1.3, the models are largely prescribed by humans. Such a feature implies that the success of the algorithms is crucially determined by the 'appropriateness' of the models. The building of such models could be time-consuming and expensive, since the assumptions or the biases prescribed by humans may not be appropriate, which would result in execution results of low quality and lead to the need to modify the original assumptions. In addition, the models used in LFD are mostly ad hoc. They are developed for the purpose of solving particular problems. The learned algorithms may not be generalized well. They may work fine with certain tasks, especially those covered by the demonstrations. However, when the nature of the tasks change dramatically, the algorithms might fail ('generalization

problem').

One way of tackling the issues related to LFD algorithms is to integrate the algorithms with the formal method as described in this dissertation. For example, in order to design an autonomous guidance system for a UAV, we can ask the pilots to flight the UAV a number of times for different environments and task requirements. Then, based on the concept of interaction patterns or some other similar developed concepts, a model, such as the HHMM model built in this chapter, can be developed. The structures of such a model are derived in a bottom-up manner. Human interventions are kept at a minimum. So we are letting the data to speak for themselves; we are letting them to fill out the details. Finally, with the model at hand, LFD algorithm can be used to figure out the exact form of the guidance algorithm. It is our belief that the assumptions or biases prescribed by humans might be an important source that leads to the brittleness of robot's performance [18]. The gained knowledge of the principles that dictate the organization of spatial behavior will support our understanding of the adaptive guidance capabilities and in turn help design algorithms needed to operate in less structured and partially known environments.

Due to the descriptive and predictive capacities of the models built from our formal method (for instance, they can not only be used to describe pilot's guidance behavior and performance, but also potentially be used to predict the behavior and performance as the environment, mission or even the operating system changes), such high-fidelity models can also be used in other engineering applications other than the design of autonomous guidance systems. For example, the model can be used in the design of human-machine interfaces. It could be used as part of an active cuing system. Predicting behavior allows to identify potential failure states and then alerting the pilot and/or switching control modality. The model can also be used to help the engineers to quantitatively evaluate how changes in the system will affect the guidance behavior and performance, thus enable them to optimize the system configurations before going into expensive and time-consuming hardware test and field evaluation.

Chapter 8

Conclusion and Future Directions

8.1 Conclusion

In this dissertation, we presented a unified, grounded and formal framework to model and study guidance behavior of both humans and artificial systems. We showed that guidance behavior can be understood as the closed-loop dynamics of agent-environment systems. This definition of guidance behavior made it possible to consider the contributions of perception and action under a holistic framework. This definition also enabled us to focus on the investigation of the fundamental characteristics of the closed-loop guidance behavior. In particular, this definition made it possible to determine that there exist two types of symmetries or equivalences that are inherent to the agent-environment dynamics or guidance behavior: one related to the concept of guidance primitives and the other related to subgoals. The former symmetry guarantees that the guidance behavior, which includes both agent states and environment state, is consistent across time and space. The latter symmetry enables humans or autonomous agents to group their behavior according to some intermediate reference states and thus divide complex tasks into a series of simpler tasks. We introduced the concept of interaction pattern to unify these two symmetries.

The concept of interaction pattern was applied to the investigation of behavioral data collected from actual guidance tasks in conjunction with computational tools adapted from machine learning and system identification. The results confirmed the existence of interaction patterns in human guidance behavior. The results also suggested that

humans utilize these symmetries to organize and plan their behavior. More specifically, the results suggested that interaction patterns play the role of basic building blocks in the planning processes. Finally, these concepts also made it possible to describe human guidance behavior with hierarchical model without imposing too many potentially inappropriate constraints. We showed that our theoretical framework together with our computational tools provide the essential foundation for the formal modeling of human guidance behavior with a hierarchical hidden Markov model.

8.2 Future Directions

As discussed in Section 7.5, due to the limited number of scenarios, only the shapes of the time-to-go are learned while the organizational rule is assumed to follow some optimality principle. So even though the framework presented in this dissertation can be data-driven in principle, in practice, data-driven (bottom-up) methods are mixed with top-down methods. The prediction of the subgoals and other switching surfaces agrees reasonably well and we are able to establish correspondences between the pilots' gaze pattern with the subgoal locations. It would be interesting to see how the framework will work out when deployed on a larger scale. This can be achieved by providing the framework with more scenarios, such as those from real flight data with full scale helicopters.

The framework presented in this dissertation provides a new way to study humans' perceptual mechanisms. The organization of behavior based on the interaction patterns must be intimately linked to the perceptual mechanisms. In fact these patterns are the manifestation of the perception-action loop. In this dissertation, we use pilots' gaze data as a way to confirm the role of subgoal in the organization of guidance behavior. Such a methodology can be further extended to quantitative studies[1] , which can help us account for the specific perceptual mechanisms in the agent-environment model.

Following the same vein, the functional description provided by the model makes it possible to understand what potential measurements can be used to investigate operator workload and attention. Brain imaging and brain activity analysis is still often treating

[1] For instance, we can study how often pilots switch to the subgoals, and, under what circumstance, the switchings become more often.

the brain as a black box. Our hierarchic model provides a more precise picture of the type of activation levels (control, perceptual, planning) expected as a function of the stage in the task.

Other future directions in engineering, such as the application of the framework presented in this dissertation to design autonomous guidance systems and human-machine interfaces, can be found in Section 7.6.2.

References

[1] E. Todorov, W. Li, and X. Pan. From task parameters to motor synergies: A hierarchical framework for approximately optimal control of redundant manipulators. *Journal of Robotic Systems*, 22(11):691–710, 2005.

[2] D.N. Lee. Guiding movement by coupling taus. *Ecological Psychology*, 10(3-4):221–250, 1998.

[3] E. Frazzoli, MA Dahleh, and E. Feron. Robust hybrid control for autonomous vehicle motion planning. In *IEEE Conference on Decision and Control*, volume 1, 2000.

[4] E. Frazzoli, MA Dahleh, and E. Feron. Maneuver-based motion planning for nonlinear systems with symmetries. *IEEE Transactions on Robotics*, 21(6):1077–1091, 2005.

[5] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G.J. Pappas. Symbolic planning and control of robot motion. *IEEE Robotics & Automation Magazine*, 14(1):61–70, 2007.

[6] AeroVironment, Inc., 2010. `http://www.avinc.com/`.

[7] Adaptive Flight, Inc., 2006. `http://www.adaptiveflight.com/`.

[8] microdrones GmbH, 2009. `http://www.microdrones.com/`.

[9] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.

[10] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, NY, 2009.

[11] B. Mettler. Structure and organizational principles of agile behavior: challenges and opportunities in cognitive engineering. *Cognitive Critique*, 3, 2011.

[12] S. Russell and P. Norvig. *Artificial Intelligence A Modern Approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, 1995.

[13] M.L. Cummings and P.J. Mitchell. Predicting controller capacity in supervisory control of multiple uavs. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 38(2):451–460, 2008.

[14] Y. Boussemart and M.L. Cummings. Predictive models of human supervisory control behavioral patterns using hidden semi-markov models. *Engineering Applications of Artificial Intelligence*, 2011.

[15] H.A. Mallot and M.O. Franz. Biological approaches to spatial representation: a survey. *Robotics and Autonomous Systems*, 65, 1999.

[16] G. Buzsáki. Theta rhythm of navigation: link between path integration and landmark navigation, episodic and semantic memory. *Hippocampus*, 15(7):827–840, 2005.

[17] M.R. Endsley. Situation awareness in aviation systems. *Handbook of Aviation Human Factors*, pages 257–276, 1999.

[18] M. Campbell, M. Egerstedt, J.P. How, and R.M. Murray. Autonomous driving in urban environments: approaches, lessons and challenges. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1928):4649–4672, 2010.

[19] D. Mumford. Pattern theory: the mathematics of perception. *Arxiv preprint math/0212400*, 2002.

[20] K.R. Hammond, G.H. McClelland, and J. Mumpower. *Human Judgment and Decision Making: Theories, Methods, and Procedures*. Praeger Publishers, 1980.

[21] D. Kersten, P. Mamassian, and A. Yuille. Object perception as bayesian inference. *Annual Review Psychology*, 55:271–304, 2004.

[22] US Army. Us army roadmap for uas 2010-2035. *US Army UAS Center of Excellence*, 2010.

[23] C. Goerzen, Z. Kong, and B. Mettler. A survey of motion planning algorithms from the perspective of autonomous uav guidance. *Journal of Intelligent & Robotic Systems*, 57(1):65–100, 2010.

[24] A. Coates, P. Abbeel, and A.Y. Ng. Learning for control from multiple demonstrations. In *The 25th International Conference on Machine learning*, pages 144–151, 2008.

[25] B.D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.

[26] C.L. Nehaniv and K. Dautenhahn. Of hummingbirds and helicopter: an algebraic framework for interdisciplinary studies of imitation and its applications. *Interdisciplinary Approaches to Robot Learning*, 24:136, 2000.

[27] H.A. Simon. *The Sciences of the Artificial*. the MIT Press, Cambridge, MA, 1996.

[28] E.A. Billing and T. Hellström. A formalism for learning from demonstration. *Paladyn. Journal of Behavioral Robotics*, 1(1):1–13, 2010.

[29] S.M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, UK, 2006.

[30] C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, NY, 2006.

[31] U. Grenander and M.I. Miller. *Pattern Theory: from Representation to Inference*. Oxford University Press, New York, NY, 2007.

[32] D. Mumford and A.Ã. Desolneux. *Pattern Theory: The Stochastic Analysis Of Real-World Signals*. AK Peters Ltd, Natick, MA, 2010.

[33] T.M. Mitchell. *The Need for Biases in Learning Generalizations*. Department of Computer Science, Laboratory for Computer Science Research, Rutgers University, 1980.

[34] J.J. Gibson. *The Ecological Approach to Visual Perception*. Psychology Press, London, UK, 1986.

[35] W.H. Warren. The dynamics of perception and action. *Psychological Review*, 113(2):358, 2006.

[36] E.C. Tolman. Cognitive maps in rats and men. *Image and Environment: Cognitive Mapping and Spatial Behavior*, pages 27–50, 1973.

[37] J. O'Keefe and L. Nadel. *The Hippocampus as a Cognitive Map*. Oxford University Press, Oxford, UK, 1978.

[38] W.G. Chase. *Spatial representations of taxi drivers*. Defense Technical Information Center, 1982.

[39] JB Ranck Jr. Head-direction cells in the deep cell layers of dorsal presubiculum in freely moving rats. In *Soc Neurosci Abstr*, volume 10, 1984.

[40] T. Hafting, M. Fyhn, S. Molden, M.B. Moser, and E.I. Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806, 2005.

[41] F. Sargolini, M. Fyhn, T. Hafting, B. McNaughton, M. Witter, M.-B. Moser, and E. Moser. Conjunctive representation of position, direction, and velocity in entorhinal cortex. *Science*, 312(5774):758–762, 2006.

[42] G. Buzsáki. Theta rhythm of navigation: link between path integration and landmark navigation, episodic and semantic memory. *Hippocampus*, 15(7):827–840, 2005.

[43] P.W. Glimcher. *Decisions, Uncertainty, and the Brain: The Science of Neuroeconomics*. The MIT Press, Cambridge, MA, 2004.

[44] N. Chater and M. Oaksford. Ten years of the rational analysis of cognition. *Trends in Cognitive Sciences*, 3(2):57–65, 1999.

[45] E. Todorov. Optimality principles in sensorimotor control. *Nature Neuroscience*, 7(9):907–915, 2004.

[46] E. Todorov and M.I. Jordan. Optimal feedback control as a theory of motor coordination. *Nature Neuroscience*, 5(11):1226–1235, 2002.

[47] N. Bernstein. *The Co-ordination and Regulation of Movements*, volume 1. Pergamon Press, Oxford, UK, 1967.

[48] R.E. Bellman and S.E. Dreyfus. Applied Dynamic Programming. *Princeton University Press*, 1962.

[49] F.C. Anderson and M.G. Pandy. Dynamic optimization of human walking. *Journal of Biomechanical Engineering*, 123:381, 2001.

[50] S.F. Giszter, F.A. Mussa-Ivaldi, and E. Bizzi. Convergent force fields organized in the frog's spinal cord. *Journal of Neuroscience*, 13(2):467, 1993.

[51] F.A. Mussa-Ivaldi and E. Bizzi. Motor learning through the combination of primitives. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 355(1404):1755–1769, 2000.

[52] S. Schaal. Dynamic movement primitives-a framework for motor control in humans and humanoid robotics. In *2nd International Symposium on Adaptive Motion of Animals and Machines (AMAM)*, 2003.

[53] G. Rizzolatti and L. Craighero. The mirror-neuron system. *Annual Review Neuroscience*, 27:169–192, 2004.

[54] M.A. Arbib. From monkey-like action recognition to human language: An evolutionary framework for neurolinguistics. *Behavioral and Brain Sciences*, 28(2):105–124, 2005.

[55] M.C. Tresch, P. Saltiel, and E. Bizzi. The construction of movement by the spinal cord. *Nature Neuroscience*, 2:162–167, 1999.

[56] E. Todorov. Direct cortical control of muscle activation in voluntary arm movements: a model. *Nature Neuroscience*, 3(4):391–398, 2000.

[57] A. Chemero. *Radical Embodied Cognitive Science*. The MIT Press, Cambridge, MA, 2009.

[58] James J. Gibson. *The Senses Considered as Perceptual Systems*. Houghton Mifflin, Boston, MA, 1966.

[59] D. N Lee, C. M. Craig, and M. A. Grealy. Sensory and intrinsic coordination of movement. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 266(1432):2029–2035, 1999.

[60] G.D. Padfield, D.N. Lee, and R. Bradley. How do helicopter pilots know when to stop, turn or pull up? *Journal of the American Helicopter Society*, 48(2):108–119, 2003.

[61] A.P. Georgopoulos. A tribute to tau. *Closing the Gap the Scientific Writings of David N. Lee*, pages 157–161, 2007.

[62] A.P. Georgopoulos. Cognitive motor control: spatial and temporal aspects. *Current Opinion in Neurobiology*, 12(6):678–683, 2002.

[63] J.A.S. Kelso. *Dynamic Patterns: The Self-organization of Brain and Behavior*. The MIT Press, Cambridge, MA, 1995.

[64] R.C. Schmidt, C. Carello, and M.T. Turvey. Phase transitions and critical fluctuations in the visual coordination of rhythmic movements between people. *Journal of Experimental Psychology: Human Perception and Performance*, 16(2):227, 1990.

[65] P.A. Corning. The re-emergence of emergence: A venerable concept in search of a theory. *Complexity*, 7(6):18–30, 2002.

[66] B.R. Fajen, W.H. Warren, S. Temizer, and L.P. Kaelbling. A dynamical model of visually-guided steering, obstacle avoidance, and route selection. *International Journal of Computer Vision*, 54(1):13–34, 2003.

[67] T. Van Gelder. The dynamical hypothesis in cognitive science. *Behavioral and Brain Sciences*, 21(5):615–628, 1998.

[68] R.D. Beer. Dynamical approaches to cognitive science. *Trends in Cognitive Sciences*, 4(3):91–99, 2000.

[69] DC Conner, AA Rizzi, and H. Choset. Composition of local potential functions for global robot control and navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, 2003.

[70] C. Belta, V. Isler, and G.J. Pappas. Discrete abstractions for robot motion planning and control in polygonal environments. *IEEE Transactions on Robotics*, 21(5):864–874, 2005.

[71] H. Kress-Gazit, G.E. Fainekos, and G.J. Pappas. From structured english to robot motion. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2717–2722, 2007.

[72] H. Kress-Gazit, G.E. Fainekos, and G.J. Pappas. Where's waldo? sensor-based temporal logic motion planning. In *IEEE International Conference on Robotics and Automation*, pages 3116–3121, 2007.

[73] G. Batt, C. Belta, and R. Weiss. Temporal logic analysis of gene networks under parameter uncertainty. *IEEE Transactions on Automatic Control*, 53(Special Issue):215–229, 2008.

[74] R.W. Brockett. On the computer control of movement. In *IEEE International Conference on Robotics and Automation*, pages 534–540, 1988.

[75] V. Manikonda, P.S. Krishnaprasad, and J. Hendler. Languages, behaviors, hybrid architectures and motion control. *Mathematical Control Theory*, pages 199–226, 1998.

[76] P. Martin and M. Egerstedt. Motion description language-based topological maps for robot navigation. *Communications in Information & Systems*, 8(2):171–184, 2008.

[77] V. Gavrilets, E. Frazzoli, B. Mettler, M. Piedmonte, and E. Feron. Aggressive maneuvering of small autonomous helicopters: A human-centered approach. *The International Journal of Robotics Research*, 20(10):795–807, 2001.

[78] V. Gavrilets, B. Mettler, and E. Feron. Human-inspired control logic for automated maneuvering of miniature helicopter. *Journal of Guidance, Control, and Dynamics*, 27(5):752–759, 2004.

[79] H.A. Simon. The architecture of complexity. *Proceedings of the American Philosophical Society*, 106(6):467–482, 1962.

[80] R. Amit and M. Matari. Learning movement sequences from demonstration. In *The 2nd International Conference on Development and Learning*, pages 203–208, 2002.

[81] S. Schaal, S. Kotosaka, and D. Sternad. Nonlinear dynamical systems as movement primitives. In *IEEE International Conference on Humanoid Robotics*, 2000.

[82] D.H. Park, H. Hoffmann, P. Pastor, and S. Schaal. Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields. In *8th IEEE-RAS International Conference on Humanoid Robots*, pages 91–98, 2008.

[83] A. Fod, M.J. Matarić, and O.C. Jenkins. Automated derivation of primitives for movement classification. *Autonomous Robots*, 12(1):39–54, 2002.

[84] D. Del Vecchio, R.M. Murray, and P. Perona. Decomposition of human motion into dynamics-based primitives with application to drawing tasks. *Automatica*, 39(12):2085–2098, 2003.

[85] K.R. Dixon and P.K. Khosla. Trajectory representation using sequenced linear dynamical systems. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3925–3930, 2004.

[86] D. Kulic, D. Kragic, and V. Krüger. Learning action primitives. *Visual Analysis of Humans: Looking at People*, page 333, 2011.

[87] J.K. Aggarwal and Q. Cai. Human motion analysis: A review. In *IEEE Nonrigid and Articulated Motion Workshop*, pages 90–102, 1997.

[88] D.A. Forsyth, O. Arikan, and L. Ikemoto. *Computational Studies of Human Motion: Tracking and Motion Synthesis*. Now Publishers, Boston, MA, 2006.

[89] R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976–990, 2010.

[90] A. Pentland and A. Liu. Modeling and prediction of human behavior. *Neural Computation*, 11(1):229–242, 1999.

[91] M. Cummings, C. Nehme, J. Crandall, and P. Mitchell. Predicting operator capacity for supervisory control of multiple uavs. *Innovations in Intelligent Machines-1*, pages 11–37, 2007.

[92] M.J. Kochenderfer, L.P. Espindle, J.K. Kuchar, and J.D. Griffith. A comprehensive aircraft encounter model of the national airspace system. *Lincoln Laboratory Journal*, 17(2):41–53, 2008.

[93] R.A. Redner and H.F. Walker. Mixture densities, maximum likelihood and the em algorithm. *SIAM Review*, pages 195–239, 1984.

[94] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[95] S. Calinon, F. Guenter, and A. Billard. On learning, representing, and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 37(2):286–298, 2007.

[96] M. Muhlig, M. Gienger, S. Hellbach, J.J. Steil, and C. Goerick. Task-level imitation learning using variance-based movement optimization. In *IEEE International Conference on Robotics and Automation*, pages 1177–1184, 2009.

[97] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden markov model. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 379–385, 1992.

[98] N.T. Nguyen, D.Q. Phung, S. Venkatesh, and H. Bui. Learning and detecting activities from movement trajectories using the hierarchical hidden markov model. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 955–960, 2005.

[99] S. Osentoski, V. Manfredi, and S. Mahadevan. Learning hierarchical models of activity. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 891–896, 2004.

[100] L. Liao, D.J. Patterson, D. Fox, and H. Kautz. Learning and inferring transportation routines. *Artificial Intelligence*, 171(5):311–331, 2007.

[101] Y. Boussemart, JL Fargeas, ML Cummings, and N. Roy. Comparing learning techniques for hidden markov models of human supervisory control behavior. In *Aerospace Conference*, New York, NY, 2009.

[102] R.A. Brooks. Intelligence without representation. *Artificial intelligence*, 47(1):139–159, 1991.

[103] N. Lynch, R. Segala, and F. Vaandrager. Hybrid I/O automata revisited. *Lecture Notes in Computer Science*, pages 403–417, 2001.

[104] B. Mettler and O. Toupet. Receding horizon trajectory planning with an environment-based cost-to-go function. In *IEEE Conference on Decision and Control*, pages 4071–4076, 2005.

[105] B. Mettler and Z. Kong. Receding horizon trajectory optimization with a finite-state value function approximation. In *American Control Conference*, pages 3810–3816, 2008.

[106] United States Geological Survey, 2006. `http://lidar.cr.usgs.gov/`.

[107] C.R. Shalizi and J.P. Crutchfield. Computational mechanics: pattern and prediction, structure and simplicity. *Journal of Statistical Physics*, 104(3):817–879, 2001.

[108] B. Mettler and Z. Kong. An extremal fields approach for the analysis of human planning and control performance. In *IEEE International Conference on Robotics and Automation*, Pasadena, CA, 2008.

[109] N. Dadkhah, R.V. Korukanti, Z. Kong, and B. Mettler. Experimental evaluation of an online trajectory optimization with spatial value functions for autonomous

guidance in challenging environments. In *48th IEEE Conference on Decision and Control*, Shanghai, China, 2009.

[110] Z. Kong and B. Mettler. Evaluation of guidance performance in urban terrains for different UAV classes and performance criteria. In *International Conference on Unmanned Aerial Vehicles (ICUAS)*, Dubai, UAE, 2010.

[111] B. Mettler, N. Dadkhah, and Z. Kong. Agile autonomous guidance using spatial value functions. *Control Engineering Practice*, 18(7):773–788, 2010.

[112] B. Mettler and Z. Kong. Mapping and analysis of human guidance performance from trajectory ensembles. *IEEE Transactions on Systems, Man, and Cybernetics*, accepted.

[113] Z. Kong and B. Mettler. Foundations of formal language for humans and artificial systems based on intrinsic structure in spatial behavior. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, CA, 2011.

[114] Z. Kong and B. Mettler. Guidance primitives and interaction patterns: foundations for formal modeling of human guidance behavior. *IEEE Transactions on Systems, Man, and Cybernetics*, submitted.

[115] S. Schaal, P. Mohajerian, and A. Ijspeert. Dynamics systems vs. optimal control: a unifying view. *Progress in Brain Research*, 165:425–445, 2007.

[116] D.M. Wolpert and Z. Ghahramani. Computational principles of movement neuroscience. *Nature Neuroscience*, 3:1212–1217, 2000.

[117] Z. Kong, V.R. Korukanti, and B. Mettler. Mapping 3D guidance performance using approximate optimal cost-to-go function. In *AIAA Guidance Navigation and Control Conference*, Chicago, IL, 2009.

[118] B. Mettler, Z. Kong, C. Goerzen, and M. Whalley. Benchmarking of Obstacle Field Navigation Algorithms for Autonomous Helicopters. In *American Helicopter Society 66th Annual Forum*, Phoenix, AZ, 2010.

[119] Z. Kong and B. Mettler. Evaluation of guidance performance in urban terrains for different uav types and performance criteria using spatial ctg maps. *Journal of Intelligent & Robotic Systems*, 61(1):135–156, 2011.

[120] B. Mettler, Z. Kong, C. Goerzen, and M. Whalley. Benchmarking guidance performance for autonomous rotorcraft. *Journal of the American Helicopter Society*, accepted.

[121] J.M. Evans and E.R. Messina. Performance metrics for intelligent systems. *NIST Special Publication*, pages 101–104, 2001.

[122] G.S. Sukhatme and G.A. Bekey. Multicriteria evaluation of a planetary rover. In *IEEE International Conference on Robotics and Automation*, pages 22–28, 1996.

[123] S.C. Wong, L. Middleton, B.A. MacDonald, and N.Z. Auckland. Performance metrics for robot coverage tasks. In *Australasian Conference on Robotics and Automation*, volume 27, page 29, 2002.

[124] E. Tunstel. Operational performance metrics for Mars Exploration Rovers: field reports. *Journal of Field Robotics*, 24(8-9):651–670, 2007.

[125] D.J. Pines and F. Bohorquez. Challenges facing future Micro-Air-Vehicle development. *Journal of Aircraft*, 43(2):290–305, 2006.

[126] M. Costello. Challenges facing Micro Air Vehicle: flight dynamics and controls engineers. In *46 th AIAA Aerospace Sciences Meeting and Exhibit*, 2008.

[127] J.H. Reif. Complexity of the Mover's Problem and generalizations extended abstract. In *The 20th Annual IEEE Conference on Foundations of Computer Science*, pages 421–427, 1979.

[128] E. Hutchins and G. Lintern. *Cognition in the Wild*. MIT Press, Cambridge, MA, 1996.

[129] A.E. Bryson and Y.C. Ho. *Applied Optimal Control*. Wiley, New York, NY, 1975.

[130] J.G. Leishman. *Principles of Helicopter Aerodynamics*. Cambridge University Press, Cambridge, UK, 2006.

[131] T.R. Yechout and S.L. Morris. *Introduction to Aircraft Flight Mechanics: Performance, Static Stability, Dynamic Stability, and Classical Feedback Control*. AIAA, Reston, VA, 2003.

[132] M.L. Fredman and R.E. Tarjan. Fibonacci Heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3):596–615, 1987.

[133] J.T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2010.

[134] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer Verlag, New York, NY, 1999.

[135] R.W. Fox, A.T. McDonald, and P.J. Pritchard. *Introduction to Fluid Mechanics*. John Wiley & Sons, New York, NY, 1985.

[136] H. Choset and J. Burdick. Sensor-based exploration: The hierarchical generalized Voronoi graph. *The International Journal of Robotics Research*, 19(2):96, 2000.

[137] H. Zender, P. Jensfelt, O.M. Mozos, G. Kruijff, and W. Burgard. An integrated robotic system for spatial understanding and situated interaction in indoor environments. In *National Conference on Artificial Intelligence*, volume 22, pages 1584–1589, 2007.

[138] Z. Kong and B. Mettler. On the general characteristics of 2D optimal obstacle-field guidance solution. In *IEEE Conference on Decision and Control*, Shanghai, China, 2009.

[139] H.A. Simon. Invariants of uman behavior. *Annual Review of Psychology*, 41(1):1–20, 1990.

[140] A. Katok and B. Hasselblatt. *Introduction to the Modern Theory of Dynamical Systems*, volume 54. Cambridge University Press, Cambridge, UK, 1996.

[141] J.E. Marsden and T.S. Raiu. *Introduction to Mechanics and Symmetry: A Basic Exposition of Classical Mechanical Systems*, volume 17. Springer Verlag, New York, NY, 1999.

[142] S. Bhattacharya, M. Likhachev, and V. Kumar. Identification and representation of homotopy classes of trajectories for search-based path planning in 3d. In *Proc of Robotics: Science and Systems*, 2011.

[143] D.A. Lind and B. Marcus. *An Introduction to Symbolic Dynamics and Coding*. Cambridge University Press, Cambridge, UK, 1995.

[144] Z. Kong and B. Mettler. An investigation of spatial behavior in agile guidance tasks. In *IEEE International Conference on Systems, Man, and Cybernetics*, Anchorage, AK, 2011.

[145] B. Mettler, N. Dadkhah, Z. Kong, and J. Andersh. Research infrastructure for interactive human and autonomous guidance. *Journal of Intelligent and Robotic Systems*, accepted.

[146] Vicon. *Vicon MX Systems*, October 2006.

[147] N. Dadkhah and B. Mettler. System identification modelling and flight characteristics analysis of miniature co-axial helicopter. *Journal of American Helicopter Society*, accepted.

[148] SensoMotoric Instruments GmbH, Warthestraße 21 , D-14513 Teltow/Berlin , Germany. *iView ETG User Manual*, version 1.1 edition, April 2012.

[149] B. Mettler and Z. Kong. Hierarchical model of human guidance performance based on interaction patterns in behavior. In *2nd International Conference on Application and Theory of Automation in Command and Control Systems*, London, UK, 2012.

[150] Z. Kong and B. Mettler. Modeling and prediction of human guidance behavior based on agent-environment interaction patterns. *The 2013 IEEE International Conference on Robotics and Automation*, submitted.

[151] J.B. Tenenbaum, V. De Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[152] O.C. Jenkins and M.J. Mataric. Performance-derived behavior vocabularies: data-driven acqusition of skills from motion. *International Journal of Humanoid Robotics*, 1(2):237–288, 2004.

[153] G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari. A clustering technique for the identification of piecewise affine systems. *Automatica*, 39(2):205–217, 2003.

[154] S.C. Shadden, F. Lekien, and J.E. Marsden. Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3-4):271–304, 2005.

[155] M.P. Do Carmo and M.P. Do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Englewood Cliffs, NJ, 1976.

[156] A.M. López, D. Lloret, J. Serrat, and J.J. Villanueva. Multilocal creaseness based on the level-set extrinsic curvature. *Computer Vision and Image Understanding*, 77(2):111–144, 2000.

[157] K.A. Thoroughman and R. Shadmehr. Learning of action through adaptive combination of motor primitives. *Nature*, 407(6805):742, 2000.

[158] J.S.B. Mitchell. A new algorithm for shortest paths among obstacles in the plane. *Annals of Mathematics and Artificial Intelligence*, 3(1):83–105, 1991.

[159] M. Egerstedt. Some complexity aspects of the control of mobile robots. In *American Control Conference*, volume 1, pages 710–715, 2002.

[160] B. Donmez, C. Nehme, and M.L. Cummings. Modeling workload impact in multiple unmanned vehicle supervisory control. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 40(6):1180–1190, 2010.

[161] ML Cummings, S. Bruni, S. Mercier, PJ Mitchell, et al. Automation architecture for single operator, multiple UAV command and control. *The International C2 Journal*, 1(2):1–24, 2007.

[162] M.L. Bolton, R.I. Siminiceanu, and E.J. Bass. A systematic approach to model checking human-automation interaction using task analytic models. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, (99):1–16, 2011.

Appendix A

Optimal Guidance Behavior of Dubins' Vehicle

In this appendix, we introduce the Dubins' vehicle model and the procedures to arrive at the optimal guidance behavior of this vehicle.

A.1 Dubins' Vehicle

Dubins' vehicle can be viewed as an idealized aircraft which satisfies following conditions: the aircraft flies in a stationary air mass with small flight path angle; and the flight path angle rate and is coordinated with the yaw rate to prevent side-slip. We assume the vehicle's dimension is reduced to a mass point. The vehicle state, g, is represented as $g = (x, y, \theta)$, where (x, y) is the location of the vehicle with respect to a global coordinate frame, and θ is the orientation of its velocity with respect to the global x-axis. Furthermore, the constant forward speed is v and the minimum turning radius is ρ. Let the only input of the system be κ, $\kappa \in [-\omega, \omega]$ and ω be v/ρ. Then the dynamics of the Dubins' vehicle can be given as follows:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \kappa \end{pmatrix} \tag{A.1}$$

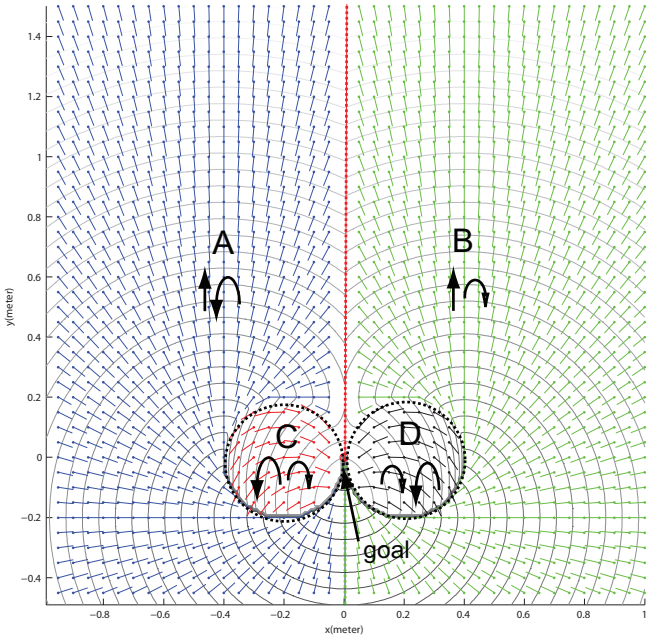


Figure A.1: Local optimal control strategy (represented by the velocity field) for Dubins' vehicle and contours of corresponding value function. The final configuration is $(0, 0, \pi/2)$. Different types of separatrices are represented by different colors: green dotted line is an attracting manifold; red dotted line is a repelling manifold. The symbols below partition names are the optimal strings.

A.2 Local Optimal Controller

For Dubins' vehicle, since the speed is constant, the minimum-time problem is equivalent to the minimum-length problem. We first find the optimal solution to Dubins' vehicle in an obstacle-free environment and then show that this solution can be extended to any environment that is convex.

Suppose the goal configuration is represented as a triple (x_g, y_g, θ_g). We use symbols $\uparrow$, $\curvearrowleft$ and $\curvearrowright$, to denote forward, left turn with radius ρ and right turn with radius radius ρ. Then a trajectory can be indicated by a string of such symbols, with a duration attached to each symbol. For example, a string $\uparrow\ (t_1)\ \curvearrowleft\ (t_2)$ means a forward flight

with t_1 followed by a left turn with t_2. It is shown that every optimal trajectory consists of two segments and it is among one of the four types:

- $\curvearrowleft (t_1) \curvearrowright (t_2)$, where t_2 should satisfy $\omega \times t_1 > \pi$;
- $\curvearrowright (t_1) \curvearrowleft (t_2)$, where t_2 should satisfy $\omega \times t_1 > \pi$;
- $\uparrow (t_1) \curvearrowright (t_2)$;
- $\uparrow (t_1) \curvearrowleft (t_2)$.

For a certain final configuration and an obstacle-free environment, each of these four strings is the optimal control strategy for a certain partition as illustrated in Fig. A.1. Then the solution space can be divided into partitions according to their optimal control strings. For instance, in domain A the optimal control string is $\uparrow\curvearrowright$. It can be easily proved that, as long as the boundaries of a local convex domain does not intersect the circles introduced by the goal $g_f(x_g, y_g, \theta_g)$, the optimal solution space $T(g_f)$ for this goal within this domain can be constructed from solution space $T(g_0)$ for configuration $(0, 0, 0)$ by two consecutive steps: first translate $T(g_0)$ by (x_g, y_g) and then rotate the resulted solution space by angle θ_g[1] .

A.3 Global Optimal Controller

To guarantee global optimality, an environment with obstacles can not be partitioned by just considering its geometrical features. This section first proves the existence of sub-goals and then describes the method to partition the space in terms of the dynamic behavior. We also show that the optimal solution space for an environment with obstacles shares similar structures as in the free environment case.

A.3.1 Existence of Sub-goals

Before we prove the existence of sub-goals[2] , the definition of visibility should be modified at first to fit our framework. A point (x, y) is defined as *left visible* with

[1] Such a property is the direct result of the symmetry existing in Dubins' vehicle's dynamics as described in Section 5.3.3.

[2] A more general definition of subgoal is provided in Section 5.3.2.

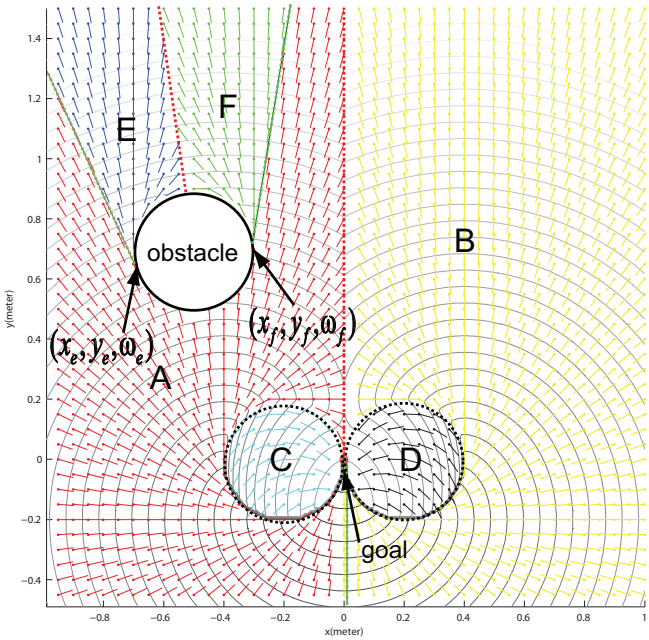


Figure A.2: Partitions for the same environment as Fig. A.1 when an obstacle is added. The influence region of the obstacle is the union of E and F.

respect to (x_0, y_0, ω_0) if there exists a $\uparrow (t_1)\ \curvearrowleft (t_2)$ type path, which starts at (x, y), ends at (x_0, y_0, ω_0) and does not penetrate any obstacle. *Right visibility* can be defined similarly. For instance, according to these definitions, for Fig. A.2, domain E is right visible with respect to (x_e, y_e, ω_e) and domain F is left visible with respect to (x_f, y_f, ω_f). And union of these two regions is called the *influence region of the obstacle*.

The sub-goal is the place where an optimal trajectory departs a domain. By applying continuous Dijkstra concept, we can prove that: the optimal trajectory starting from a point inside the influence region of an obstacle must consists of the following two segments: a sub-path generated by local optimal control law, which starts from this point and ends at a visible vertex of the corresponding obstacle, and a sub-path which starts from that vertex and ends at the global goal. This vertex is called a sub-goal, such as points (x_e, y_e, ω_e) and (x_f, y_f, ω_f) in Fig. A.2.

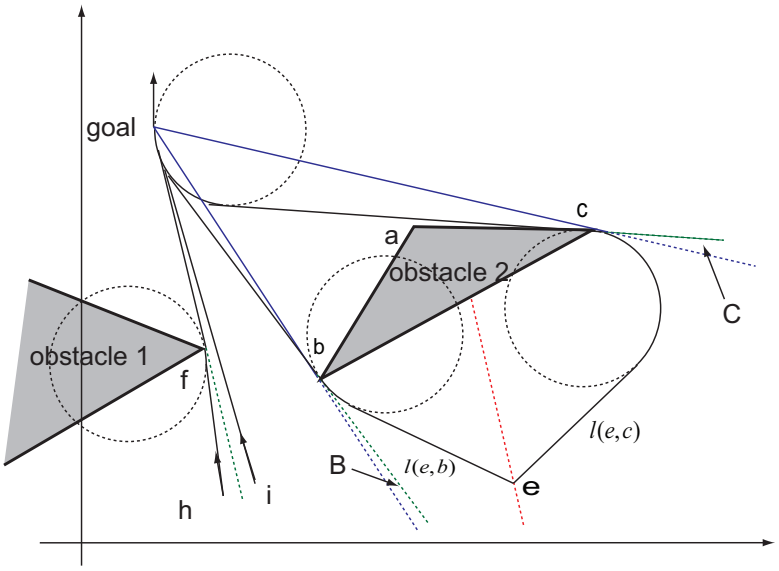


Figure A.3: Existence of attracting manifolds (green) and repelling manifolds (red) for the optimal solution space of an environment with obstacles. The extended segment is of color blue. Obstacle 1 is infinite to the left.

A.3.2 Partitioning of Optimal Solution Space

Due to the characteristics of Dubins' vehicle, its optimal control problem can be transformed to a computational geometry problem. There exist one-to-one correspondences between structures of optimal solution space for Dubins' vehicle and structures of shortest Euclidean path map. And these correspondences are: the sub-goal of a local domain to the root of a cell, the attracting manifold to the extended segment, the repelling manifold to the bisector, the left or right visibility to the ordinary sense visibility.

A similar structure to extended segments can be constructed as shown in Fig. A.3: for example, vertex f is right visible with respect to the goal, so a ray can be extended from the right circle induced by the goal to vertex f. The extended part of this ray, illustrated as green in Fig. A.3, is the structure. Now suppose h and i are two points around this structure, one of them h is within the influence region and the other i is not. The optimal solution to i is the same as the one when there are no obstacles. But the optimal solution to h is "influenced", since it is not visible with respect to goal. According to the existence of sub-goal, the optimal solution of this point is first to

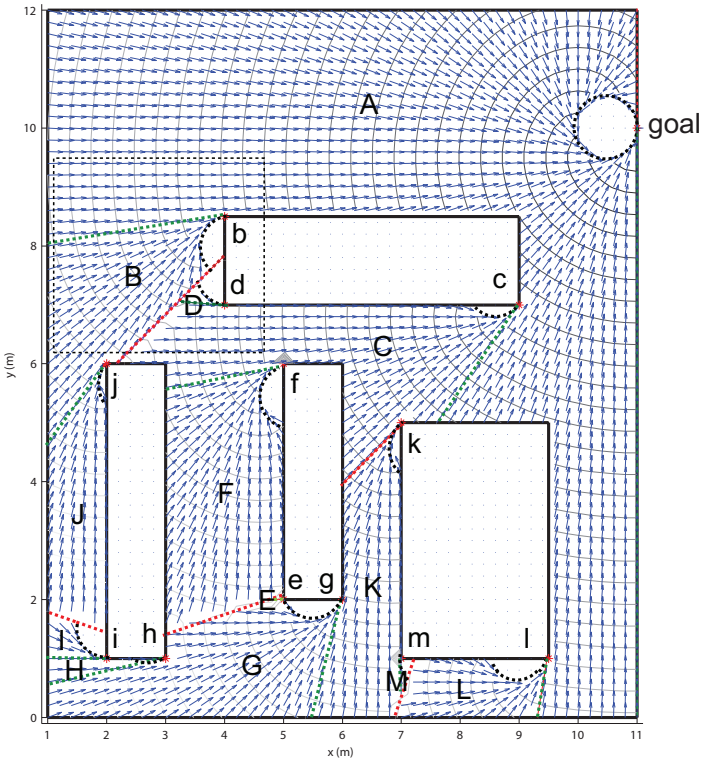


Figure A.4: Partition of the optimal solution space for an environment with mutiple obstacles. The meaning of each color is the same as Fig. A.1. The sub-goal for each domain is illustrated as a red star. A close view of the enclosed area is shown in Fig. A.5.

follow local optimal strategy to vertex f and then follow another local law to the goal. For this kind of structure, both the optimal value function and the optimal velocity are continuous. It has the same property as an attracting manifold. To verify that it is indeed an attracting manifold, we can trace the optimal trajectories from points h and i backward in time, these two trajectories will diverge. Fig. A.3 also shows the difference between ordinary visibility and the visibility used in this paper. Points in domain B is right visible with respect to goal. But they are not visible to the goal in the ordinary sense. On the other hand, points in domain C is not visible with respect to the goal. But they are visible to the goal.

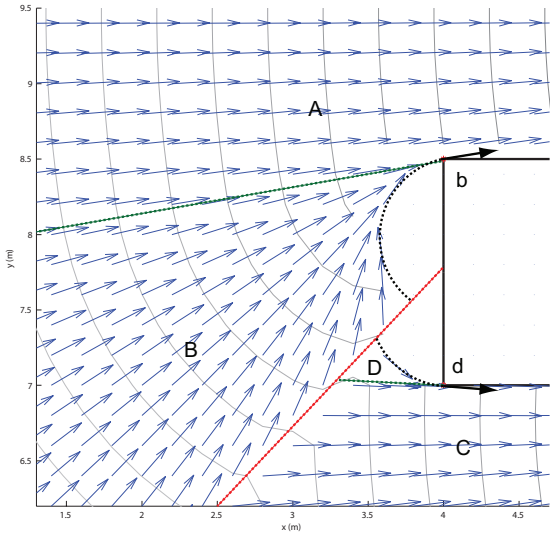


Figure A.5: Close view of the optimal solution space in Fig. A.4. The configurations at sub-goals are given.

For Fig. A.3, suppose the costs at vertices (x_b, y_b) and (x_c, y_c) are $l(x_b, y_b)$ and $l(x_c, y_c)$, respectively. And their corresponding optimal velocities are ω_b and ω_c. For point e, which is inside the influence region of obstacle 2, since it is left visible with respect to (x_c, y_c, ω_c) and right visible with respect to (x_b, y_b, ω_b), its optimal solution will be: $arg_u(\min(l(e,c) + l(x_c, y_c, \omega_c), l(e,b) + l(x_b, y_b, \omega_b)))$. If the two terms are the same, the vehicle can either head to the vertex b or the vertex c. Both of these two strategies are optimal solutions. Within the influence region, the set of all the points having this property forms a structure similar to bisector in computational geometry. Around this structure, the optimal value is continuous while the optimal velocity is not. Follow the same argument above, we can verify that this structure is a repelling manifold.

Based on the similarities between the structures of optimal solution spaces and the structures of shortest path map, similar procedures as in computational geometry can be taken to partition the whole free space [3] . An example of such a partition is shown

[3] Under the assumption that the minimum turning radius ρ is moderate with respect to the size of

in Fig. A.4 and Fig. A.5.

In conclusion, this section shows that, for Dubins' vehicle, it is possible to partition the space in a particular way such that sequential execution of local optimal control laws will lead to a global optimal solution. The effects of vehicle dynamics on the solution space can be reflected by different types of manifolds, which are grown from obstacles. (Here the goal can be seen as a degenerate obstacle.)

---

the obstacles.