Title
Modeling of Human Decision Making via Direct and Optimization-based Methods for Semi-Autonomous Systems

Author
Lin, Theresa

Publication Date
2015

Peer reviewed|Thesis/dissertation

Modeling of Human Decision Making via Direct and Optimization-based Methods for Semi-Autonomous Systems

by

Theresa Lin

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

in

Engineering - Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Francesco Borrelli, Chair
Professor J. Karl Hedrick
Professor Thomas L. Griffiths

Fall 2015

Modeling of Human Decision Making via Direct and Optimization-based Methods for Semi-Autonomous Systems

Abstract

Modeling of Human Decision Making via Direct and Optimization-based Methods for Semi-Autonomous Systems

by

Theresa Lin

Doctor of Philosophy in Engineering - Mechanical Engineering

University of California, Berkeley

Professor Francesco Borrelli, Chair

In the generation where progression in technology have propelled the research of human-machine intelligent systems, it is becoming increasingly important to study the fundamental principles behind human behaviors from a computational point of view. This thesis aims to use advanced technologies, combined with advanced modeling methodologies and modern control algorithms, to study the principles behind the modeling of human decision making for two purposes. First, to use computational modeling frameworks to better understand the mechanisms and factors which affect decision making in different problem contexts, both from the controls design and psychology perspective. Second, to introduce a unifying framework for integrating human policies into controller design in order to improve the performance of human-machine intelligent systems. Models are grouped into either the direct method or the optimization-based method. Direct methods map observations to decisions directly through stored function maps and are associated with lower-level, reflexive and repetitive behaviors. Optimization-based methods, associated with higher-level planning behaviors, require extra cognitive effort to generate state predictions in search for a solution that satisfies a set of criterions.

To support the proposed modeling frameworks, both game-based and real-world experiments are conducted with the aid of advanced test apparatus and sensor technologies. Driving experiments on real roads and simulators explored driver behavior in everyday driving on the highway, extreme driving on slippery surfaces, and distracted driving with obstacles. Game-based experiments involving a projectile and a dual-task game were performed to collect consistent data in a controlled setting, and also designed to parallel the driving contexts in the real-world.

Results from the experiments showed that in extreme driving, a piecewise-affine switched model with two modes was used to differentiate the behavior in the linear and saturation region of the tire. Simulations from a model predictive approach also showed that drivers need to be aware of the nonlinearity in the tire dynamics in order to follow a learned reference trajectory. Similarly, results from the projectile game also revealed that subjects adopted switched strategies due to the nonlinearity and uncertainties involved in the problem. In particular, subjects used switched strategies depending on whether it was an old or new scenarios. In the old scenario, subjects likely

used a linear feedback strategy mapping errors directly to the change in control input. In the new scenario, subjects used an optimization-based strategy which minimized a combination of time to hit target and change in the control inputs in order to minimize the effect of uncertainty on the state trajectories.

To validate the conjecture that humans perform mental simulations of states for the optimization-based algorithms, eye-tracking glasses were used to better estimate the cognitive states of the subjects. Eye-tracking on the driver during curve negotiation showed switching between a far-point and a near-point, and eye-tracking on the juggler revealed an early shift in gaze to the predicted apex of the ball. Both results supported how humans will perform mental simulations with some form of a forward model.

In addition to the continuous decisions associated with the previously discussed applications, the discrete decision making of attention allocation in a dual task problem was also investigated and modeled as a Markov decision process (MDP). Simulation and inverse reinforcement learning showed that subjects first adopted a conservative approach and later converged to a riskier strategy as they interacted with better certainty in the game. A similar framework was applied to a real texting while driving context on the highway. Results from eye-tracking showed that attention duration on the phone decreased as vehicle speed increased, which agrees with the predictions of the MDP model.

Lastly, to integrate the advanced modeling methodologies into intelligent systems, the framework of model predictive control was modified to include driver models in the predictions. Controller intervention is minimized such that the semi-autonomous vehicle behaves more like the driver. This has led to shared control algorithms with better integration of human models.

This thesis is dedicated to my parents,

who have patiently supported me during my 6+ years of absence.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to begin by thanking everyone who have helped me, motivated me, walked with me, and stood silently by me, on this doctorate journey. It certainly has been a long and winding journey with many ups and downs, and at times, with no end in sight. But I pulled through in the end! And I could not have achieved what I set out to accomplish without both the academic and emotional support of many.

First and foremost, I would like to express my deepest gratitude to my research advisor Prof. Francesco Borrelli, for continually showing support in my work. Thank you for keeping me motivated with your insightful comments and research ideas. Without it, I would not have been able to achieve the work in this thesis.

I would like to thank Prof. Tom Griffiths for inspiring me with research ideas connecting to cognitive psychology; Prof. Oliver O'Reilly for encouraging me to apply to graduate school and writing my letter of recommendation for Berkeley; Prof. Pieter Abbeel for being an inspiration and for being one of the nicest professors out there who shows a lot of care for the students; Prof. Karl Hedrick and Prof. Masayoshi Tomizuka for being welcoming when I first started grad school.

I would like to thank Miroslav for his mentoring at the beginning of my journey; Yiqi for starting this journey with me and helping me along the way; Andrew, Ashwin and Victor for all the work we have done together. Winter tests would have been even colder than it already is without your company.

I would like to thank Eric for all his help with the winter tests. Jacky, Peter, Alex and Nhi for your help setting up the virtual experiments, building the hardware and writing the software for it.

I would like to thank my lab mates Sarah, Frank and Jason for keeping me company and spicing up my grad student life; and Sergey for your words of encouragement on our juice breaks.

I would like to thank all my other friends at BATS and at UC Berkeley, Selina, Raechel, Chi-Shen, Oak, Mike and Shih-Yuan. We were all in this together. Thank you for being there. Grad school would not have been the same without our quals study sessions. Also, shout out to Max, who is one of the smartest person I know. You continually to inspire me with your brilliance and humbleness.

I would like to thank all my friends outside of grad school especially Ping, Trang and Ami. You have continually shown me support throughout all these years. Thank you for the emotional support, for keeping me sane, for understanding of my struggles, and for not asking those questions which every grad student fears to be asked.

I would like to thank Uncle and Auntie Lin, who have foster me with great care and love from the very first step I took in Berkeley. Thank you for being like a family to me.

Lastly, I would like to thank my parents, who have both sacrificed so much for me to be where I am today. Thank you for your unconditional love and care for the last 29 years. Thank you for the freedom you have given me. Thank you for putting up with my selfishness and temper. Thank you for letting me know you will still be proud of me no matter what I decide to do. Thank you for allowing me to delay my life by 6 years and patiently waiting for me to achieve my goals. I am truly grateful for all the support you have shown me all these years.

Chapter 1

Introduction

The topic of human behavioral modeling is a multidisciplinary research field ranging from cognitive psychology to computer science. With the advancement in technologies, there is an increasing emergence of human-machine intelligent systems, designed to performed human-centered tasks or to interact with humans. Therefore, it is important to understand human behaviors and integrate this knowledge into controller design. Many applications in computer science and engineering use machine learning algorithms to train robotic systems that are designed to either imitate human behavior or behave in a cooperative manner with humans. These data-driven approaches are designed for efficient identification and simulation of trained behavior, but does not relate to the fundamental psychological principles. To better understand the underlying principles behind human cognition, psychologists have used parametric methods, both from the top-down and bottom-up, to model how humans make inferences about the states of the uncertain world, and how the uncertain estimates about the world is used in decision making.

In the rest of this chapter, we will first discuss the aforementioned topics with detailed examples, and then summarize the issues associated with the engineering-oriented and psychology-oriented approaches to modeling human behavior. From these drawbacks, we will present our motivation and research directions associated with the studies in this thesis, which is to use advanced modeling frameworks to better understand the mechanisms and factors which affect decision making in different problem contexts, and attempt to connect psychological principles with engineering-oriented models. In addition, introduce a unifying framework for integrating human policies into controller design in order to improve the performance of human-machine intelligent systems.

1.1 Importance of behavioral modeling for human-machine intelligent systems

The progression in technology has propelled an increased research of human-centered intelligent systems to improve efficiency and quality of life. The advancements in computational power [1], sensor technologies [2], and computational algorithms such as sensor fusion [3] and machine learn-

ing [4] has enhanced the implementation of applications which involve human-machine interactions or machines designed to performed tasks that were traditionally done by the human [5].

Examples include the growing interest on intelligent vehicle systems, both from academic research and the automotive companies in the industry, as well as from the National Highway Traffic Safety Administration (NHTSA) for the purpose of reducing the number of fatal accidents on the road [6]. From the automotive industry, the production of intelligent vehicles and the integration of assistance features are also increasingly emerging into market [7]. Academic efforts can be seen in the numerous related conferences and symposiums, such as the IEEE Intelligent Vehicles Symposium, IEEE International Conference on Intelligent Transportation Systems, and International Symposium on Advanced Vehicle Control.

In particular, the DARPA Grand Challenge has encouraged numerous research institutions to build driverless vehicles for self-driving and navigation both in the dessert and in urban environments [8, 9, 10, 11]. In the task of environment mapping for urban self-driving vehicles, it is no longer sufficient to just estimate passive variables such as the road curvature [12] and surface conditions [13], but it is of great importance to include predictions of future actions of other drivers because it allows the control algorithm to execute a safer and more effective plan. For example, conventional adaptive cruise control algorithms follows the front vehicle by maintaining a safety time gap [14]. Total fuel cost could be reduced if the fuel consumption model is included in the control algorithm using model predictive control [15], and efficient following can be achieved if acceleration and deceleration of the front vehicle can be predicted in advance as a longitudinal driver model [16].

In some applications where the driver is considered as a disturbance [17, 18], with deterministically bounded motions, then the worst case scenario is always considered [19], i.e. steering behavior which could result in sudden lane departure, then vehicles will have to travel far apart from each other in order to be safe, which may not be the most effective method. Alternatively, if driver models are more carefully modeled and considered to have a probabilistic distribution [19, 20], then lane departure scenarios will reduce to a smaller percentage and a more effective algorithm can be implemented instead [19].

As mentioned prior, the original DARPA Grand Challenges has focused mainly on the task of driving, but has recently invested more interested in creating and funding challenges involving robotic system performing human-like tasks. In the 2012 DARPA Robotics Challenge, contestants were given the task to develop ground robots capable of executing complex tasks in dangerous, degraded, human-engineered environments [21]. Tasks that may be easy for the human but difficult for robots, such as walking, getting in and out of cars and manipulation of tools are some examples included in the challenge. The Atlas by Boston Dynamics is a high mobility, humanoid robot designed for emergency search and rescue operations in rough terrains [22]. Potentially, these robotic systems could be applied to substitute humans in dangerous military applications [23]. The features of performing human-like tasks, manipulating objects and navigating in a human-centered environment also exists in many humanoid robot applications [5]. For example, the Honda ASIMO is a humanoid robot designed to assist humans with everyday tasks in the most friendly and least intrusive manner [24].

The aforementioned applications all involve a robotic system where the behaviors of the human

are included in the design loop, whether it is to manipulate human-centered tasks or physically working in cooperation with the human. Therefore, it is important for the intelligent system to have knowledge of this interaction and understanding of human behaviors and cognitive states in various scenarios and contexts. In the context of driving, better understanding of driver models has multiple benefits including reliability issues and driver acceptance of intelligent vehicles [25]. In the next section, we will discuss the research on human modeling from computer science which has focused on the data-driven approach.

1.2 Human modeling applications using data-driven methods

In the field of computer science and engineering, many human-centered applications are designed to either imitate human behavior, or behave in a cooperative manner while working with humans. With this mindset, control algorithms are treated as a black box and the focus of the model is placed on the output of the black box. The inner structure of the algorithm and how it relates to human cognition is not addressed. Emphasis is placed on using a model structure that can be efficiently identified from extensive data, provide good predictions and simulate the behavior of the human effectively within the trained context. We generalize these as modeling from a data-driven approach.

Various machine learning algorithms, such as learning from demonstration [26], have been used to reproduce human behavior on robotic systems. Demonstrations from both human motor skills and trajectory planning were used to teach robots how to perform manipulation tasks. For example, nonlinear differential equations were trained to replicate hand grasping motions on the Sarcos dexterous robot arm [27], and trajectory transfer through non-rigid registration was used to teach a PR2 robot how to tie knots [28]. Another example uses dynamical movement primitives, learned from demonstrated trajectories, as a central pattern generator of a biped robot to reproduce human locomotion [29]. Humanoid robots were used to imitate the strokes of tennis players, taiko drummers and robotic juggling [30].

In driving applications, data-driven methods can be used to identify driver behaviors in different traffic conditions. For example, using Markov models, data from vehicle states, driver input and traffic conditions can be used to identify and predict lane change behavior [31] and turning at intersections [32]. Recursive least squares is used to identify driver steering during lane keeping on the highway in real-time [33]. Gaussian mixture models can also be used to predict lane changing [34] and longitudinal behaviors with car following [35] on the highway. Potentially these models can be implemented in autonomous algorithms for driver preference matching [36]. A cluster-based probabilistic model is used to identify and predict the behavior of a distracted driver [19]. Apprenticeship learning via inverse reinforcement learning was used to imitate driver behavior on a highway in a virtual game [37].

The above examples of using data-driven methods to imitate human behavior is appropriate if we only care about the predicted output of the model. However, unlike the consistent performance of machines, human behavior is highly variable due to uncertainty and multiple factors that are not account for. Often times the trained models will work well for specific scenarios but cannot be

generalized to new situations. Therefore, instead of using efficient data-driven methods to match the output of a specific task, psychologists strive to better understand the general principles that govern human behavior through the use of parametric methods.

1.3 Understanding human behavior from psychology using parametric methods

To answer the fundamental questions of human behavior, psychologists have long studied the associated components which makes us human: our senses, our decisions, our actions, and our social interactions. Neuroscientists have also mapped out the corresponding components in our brain using brain imaging techniques. The language and modeling structures used to describe our understanding of human behaviors have evolved from descriptive theory to computational models.

To be able to simulate human behaviors, the cognitive architectures proposed in [38] attempts to unify the various results of cognitive psychology into a comprehensive computer model. The proposed modeling language is suitable to simulate both simple and complex behaviors which involve multiple psychological phenomena such as memory and cognitive attention bottleneck in applications like highway driving [39] and driving under distraction [40]. The individual modules dealing with different cognitive aspects of human behavior can either be descriptive or computational and represented at different levels of abstraction, primarily categorized into the bottom-up approach and the top-down approach.

Of particular interest in this thesis, is behaviors which involve dynamic relationships of variables within the scope of the problem. The cognitive processes involved in a dynamic task include making inferences about the world and making a decision based on these inferences. In the following sections, we will first introduce the modeling approaches used at different levels of abstraction, discuss examples of models used for inference problems, and how uncertain knowledge about the world are incorporated into the decision making process. In particular we will draw on examples of sensorimotor learning and control, which has been well-researched in the context of hand reaching.

1.3.1 Level of abstraction: bottom-up vs. top-down

Similar to Marr's hardware level of human vision [41], the lowest level of abstraction focuses on the mechanisms at the neuronal level and models how the physical connections between the neurons can lead to macro-level functional phenomena [42]. This bottom-up approach is different to the top-down approach which bypasses the physical implementations of our brain and models observed behavior directly using various computational models. The top-down approach can deal with the abstract principles that allow agents to solve real-world problems [43], and examples include probabilistic models like Bayesian reasoning.

Bottom-up approach: In the bottom-up (also known as the connectionist) approach, cognitive scientists model cognition based on the idea that the knowledge underlying cognitive activity is

stored in the connections among neurons [44]. In connectionist models, knowledge is acquired by altering the strengths of connections among neuron-like processing units [44]. The simplest way to model the neural network in our brain is to treat each neuron as a function node with different weights [45]. In fact, this computational architecture have also inspired artificial neural networks used in machine learning [45] and control algorithms [46]. Even more recently, deep architectures in unsupervised learning, also known as deep learning, has been heavily applied to represent high-level abstractions such as vision, language, and other AI-level tasks [47, 48]. For example, using unlabeled images to train deep neural networks to recognize high-level concepts such as cats [49]. These deep architectures, composed of multiple levels of non-linear operations, such as neural nets with many hidden layers, can be paralleled to the extremely complicated neural networks formed inside the human brain. The incredible ability of the aforementioned example in learning a categorical concept of the world without supervision draws a strong a parallel with how humans adapt and learn using the neural network in our brains [50].

To further the physical representations of the human brain, functional models of each neural node in the neural network is replaced with biological cellular models with more complex representations. To fully understand and simulate human brain activity, the Human Brain Project attempts to build biologically accurate models of the brain from first principles, models each neuron individually on an IBM chip, and connects into a giant network, also known as IBMs Blue Gene supercomputer [51]. In addition, the electrical signal passing between the neural network is subject to multiple sources of noise [52]. As a precursor to simulating the human brain, [51] has successfully modeled the entire brain of a 2 week old rat.

Despite the ability of the bottom-up approach to model simple processes and inference problems, it is still limited to a small range of applications and it is unclear how this can scale up to decision making scenarios which involve dynamic environments. Therefore, to bypass the limitations of the neural network model while still pertaining to the physiology of the brain, neuroscientists have focused on certain regions in the brain i.e. motor cortex for motor control [53]. With brain-imaging techniques such as the functional magnetic resonance imaging (fMRI) and virtual reality experiments, researchers can now verify certain concepts and how it correlates to a specific part of the brain. For example, mirror neurons, initially discovered in the premotor cortex of monkeys, have recently been shown to respond in both scenarios where in one, a particular action is performed by a monkey and in the other scenario, the same monkey observes another monkey performing the same action [54], which supports the simulation theory [55]. A similar matching system has also been shown to exist in humans. When experts in either classical ballet or capoeira, or non-experts were individually asked to watch videos of classical ballet or capoeira actions, fMRI showed greater bilateral activation in the regions of the brain associated with motor action for subjects who were experts of the movements shown in the video [56]. This suggests that the human brain understands actions by motor simulation. However, the computational component of complex processes like these are hard to model with the bottom-up approach. Instead, the top-down approach can directly model the observed behavior using computational models.

Top-down approach: The top-down (also known as the probabilistic) approach, bypasses the need to consider how the neural connections lead to cognitive concepts, and models behaviors directly from observations [43]. There are many forms of modeling using the top-down approach. The most popular approach that have garnered a lot of attention recently is by using Bayesian methods. The fundamental idea behind Bayesian methods is treating humans as probabilistic decision makers, which lines naturally with how humans behave with uncertainty. Uncertainties can come from the environment, our senses, our actions, or even others actions. Integrating and balancing the uncertain world around us with our own prior knowledge of the world is the core of Bayesian methods. Many of the Bayesian methods have been applied to inductive reasoning which involves making inferences about the world. It involves combining noisy observations with prior knowledge in order to make inferences about the world. For example, probabilistic Bayesian methods like the Rao-Blackwellized particle filter was used to model state estimation and human visual attention during multiple object tracking [57]. Inference problems are a necessary component in human decision making because it provides us with state estimation feedbacks of the systems involved. Statistical decision theory is another top-down approach that is used to model how we integrate our uncertain estimates of the world to arrive at the best calculated decision in dynamic problems. We will discuss this in detail in the next section.

1.3.2 Computational models for decision making

Decision making problem which involve dynamics, or temporal evolutions of states, is an important part in understanding human behaviors for intelligent systems. We will introduce examples and discuss several characteristics related to the decision making of human behaviors which include: optimality of behavior, forward models and mental simulations, and dealing with state uncertainty.

Human sensorimotor skills can be modeled with concepts borrowed from the controls community [58]. Reaction times and corrective movements suggests that forward models, similar to the principles of feedforward control [59], are used in reaching movements to predict sensory feedback. An affluent copy of the motor command is used to predict future position of the hand. Inverse models are used to calculate how much force is required to achieve a certain goal [60]. The ability to perform fast reaching in the presence of sensory feedback delay suggests the use of these forward models [59]. The activation of mirror neurons during both the action and observation of identical tasks suggests a connection to the simulation of forward models [61].

Also included in sensorimotor learning is the importance to include the uncertain and noisy component of human behavior in computational models, since it has been shown both experimentally and computationally that multiple sources of noise contributes to cellular and behavioral trial-to-trial variability [52]. The different sources of noise in the nervous system, from the molecular to the behavioral level, is also reviewed in [52], and noise enters the control framework in many forms included in the sensing and actuation [62]. The Kalman filter is a classical linear systems observer used in many controls applications, and is used to model how humans process sensory feedback corrupted with noise [62]. For example, Bayesian statistics and optimal control have also been used to model reaching movements [63]. Subjects are asked to reach towards a target point with their visual feedback of the hand position occluded. Subjects are provided

with feedback midway but the virtual feedback has a noise component added to it. The noise is sampled from a prior distribution. Results are compliant with Bayes rule and models show how subjects tradeoff between prior belief about the noise distribution and their own belief about the hand position. To summarize, motor behavior of humans are viewed as a problem of maximizing the utility of movement outcome in the face of sensory, motor and task uncertainty. [64] discusses a variety of utility functions that is used by the human in the trajectory planning including cost of time, minimum energy, speed-accuracy tradeoff, discounted reward, naturalistic versus explicitly defined cost function etc.

We conclude that human decision making, in tasks which involve temporal evolution of system states and require trajectory planning, can be viewed as an optimal control problem where the effects of sensory, motor and task uncertainty is considered. This also implies that in order to find the optimal solution or trajectory to the planning problem, humans intrinsically possess the ability to make state predictions during the cognitive processes involved in decision making.

1.4 Conclusions and research direction

Reviewing the discussion in Section 1.1, we have established that there is an increasing interest on human-centered intelligent systems which involve human-machine interactions or machines designed to performed tasks that were traditionally done by the human. Therefore, it is important for intelligent systems to have knowledge of this interaction and understanding of human behaviors and cognitive states in various scenarios and contexts.

The need to better understand human behaviors for the purpose of designing improved human-machine systems motivates the following two contributions addressed in this thesis:

- To use computational modeling frameworks to better understand the mechanisms and factors which affect decision making in different problem contexts, both from the controls design and psychology perspective.
- To introduce a unifying framework for integrating human policies into controller design in order to improve the performance of human-machine intelligent systems.

In Section 1.2, we presented the various engineering applications of intelligent systems that are designed with the purpose to either imitate human behavior, or behave in a manner that can be integrated to work with humans. Examples included humanoid robots for personal assistance and military use, as well as self-driving cars. We argued that, although the data-driven models used in these applications can be efficiently identified from extensive data and capable of providing good imitations of human behavior, the inner structure of the algorithm and how it relates to human cognition is not addressed and therefore it is harder to generalize the identified behaviors to new scenarios.

To better understand the underlying principles behind human cognition, we reviewed in Section 1.3, literature from cognitive psychology and neuroscience and the various parametric methods

that were used to model these fundamental concepts. We presented the top-down and bottom-up approaches from the field and established that even though top-down approaches bypasses the physical implementations of the neural network in our brains, it has the capability to address a wider range of abstract principles that allow agents to solve real-world problems.

Focusing on the top-down approach, we discussed examples of how probabilistic models are used to explain the cognitive principles behind inductive reasoning in the uncertain world. To connect this with dynamic decision making, we presented examples in the context of sensorimotor learning and used hand reaching experiments to identify how the human makes decisions on the trajectory of the hand in the presence of sensory uncertainties. Literature from various hand reaching experiments have concluded: that humans will act either optimally or sub-optimally in order to minimize different cost functions under the presence of state uncertainty; and that humans have intrinsic representations of forward models for improving the performance of motor control.

To draw motivation to our research, we argue that even though the above discussed hand reaching experiments have identified the principles behind continuous decision making with motor control, it still did not address the issue of decision making that involves manipulation or interaction with another object in a dynamic environment. Such problem contexts are especially important for human-machine interactions and has been studied with data-driven approaches of modeling. In addition, the cognitive states of the human in these experiments were inferred from observed hand trajectories and reaction times instead of being measured directly.

To integrate the two extremities, this thesis will use advanced test apparatus and sensor technologies to measure the cognitive states of the human and design experiments which will be able to draw a connection between the results from psychology to the applications in engineering. We will approach this by conducting dynamic experiments in the real-world setting such as driving, and simultaneously design game-based scenarios which parallels the former. We aim to use advanced modeling methods to identify parametrically the psychological principles, such as optimal behavior, forward models and dealing with uncertainties in the system, that was discussed in the hand reaching examples.

In addition to the principles behind decision making in a continuous setting at the lower-level, many intelligent systems also need to be designed to handle higher-level decision making which often involves discrete variables. As mentioned in Section 1.2, lane changing and turn detection has been modeled with data-driven methods. Section 1.3 briefly introduced a descriptive approach using cognitive architectures to simulate attention allocation in a distracted driving setting. However, descriptive modeling languages like this is more difficult to integrate into control systems which adopt computational algorithms. Similarly, we will address this issue and conduct both real-world driving and game-based experiments involving attention allocation, and use parametric methods to parallel the associated psychological principles.

Lastly, to address our second contribution, we will introduce an advanced control framework which is able to seamlessly integrate predictions of human models directly into the control algorithms. With the large variety of computational methods for modeling human behaviors, both data-driven and parametric, the novelty in our model predictive control framework is the capability to handle any form of modeling, as long as a sequence of predictions is provided by the human model.

Chapter 2

Modeling Framework: Direct and Optimization-Based Methods of Policy Functions

To better understand the underlying principles behind human decision making in various contexts, psychologists have traditionally approached this question from a behavioral perspective [65]. Increasingly we are seeing more research efforts applying computational methods to explain specific behavioral trends at different levels of abstraction. For example, the connectionist approach models the neural network in the cerebellum [44], probabilistic models for inductive reasoning [43], or a control theoretic approach to modeling sensorimotor control in reaching [64]. In this chapter, we follow a similar control-based paradigm and present the different computational models that can potentially be used to model the decision making process, especially those involving interactions with a dynamic system.

Borrowing the architecture from the controls community [58], we start by introducing the general framework corresponding to the decision making in a dynamic environment, and discuss how the associated process modules including control and feedback relates to the human. Decision making is posed as a policy function mapping observed input and goals to the decision. We propose that the policy function map can be represented by two methodologies: direct methods and optimization-based methods. Direct methods maps input to output directly, as opposed to the optimization-based methods which involve state predictions and searching for a solution to satisfy certain criteria. Direct methods have the advantage that the associated model parameters are much easier to identify, and therefore is also a suitable candidate for the integration with controller design of semi-autonomous systems.

We present the mathematical formulations of: direct methods including linear feedback, piecewise-affine, probabilistic models and support vector machine; and optimization-based methods including predictive models and Markov decision process. The models introduced here will be used later in Chapter 4 for behavioral modeling and identification of specific applications, and also in Chapter 5 for integration into controller design.

2.1 Framework for modeling decision making in a dynamic environment

The complexity of human behaviors makes finding a unifying framework a difficult task. Different frameworks will be selected depending on the purpose of the modeling problem. For example, cognitive architectures [38] are designed to simulate and integrate all the cognitive processes involved in psychology, however the use of descriptive languages makes it harder to combine with control algorithms. We will start from the control theoretic point of view and discuss the roles of the different modules involved in this framework, integrating principles from psychology and neuroscience and drawing parallels to existing control systems.

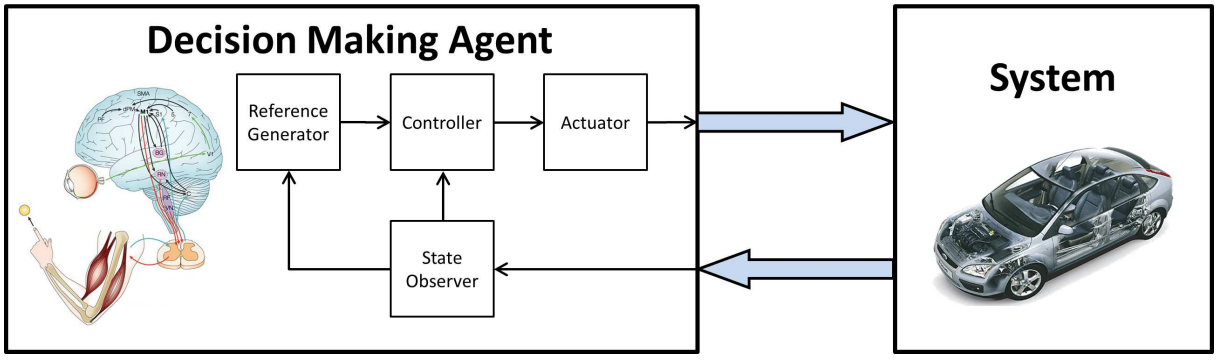


Figure 2.1: Control-theoretic framework for modeling the decision making agent and its interactions with a dynamic system. (Image source: [66])

2.1.1 Components in the decision making framework

Presented in the Figure 2.1 is the framework used to describe the interactions between a controller or decision maker, and the system and the environment. The blocks represent different computational processes, and the arrows represent the flow of information between them. As shown in the diagram, the two major components in the framework are two interacting blocks / processes exchanging information with each other such as state feedback, inputs or decisions.

The first block represents an agent who is the controller and decision maker. An agent can refer to humans, robots or any other intelligent systems such as the control algorithm of autonomous vehicles. The second block can either be another agent, or it can represent the process or system that is controlled by the first agent. Systems can refer to simple physical phenomena such as a ball bouncing on the ground, or more complex processes such as a car been driven on the road. For simplicity, in this chapter we will confine the discussion of the framework down to an agent controlling a system (and not with another agent), and will focus on discussing the various components that exists within the agent.

System / process component: The systems module represents the processes that are being controlled by the controller or the human subject. Often times, this refers to a continuous process which evolves over time, governed by differential equations i.e. $\dot{x} = f(x)$. To simplify the analysis and controller design, linear systems, which has linear matrix representations of the differential equations, are desirable. Therefore, even with nonlinear systems, linearization techniques are often used. When modeling a controlled process, the variables which represents the state and inputs need to be clearly specified. For example, a system which describes the process of a vehicle being driven: can have the position, velocity, yaw angle, and yaw rate as the states; and the steering wheel angle, brake pedal, and throttle position as the inputs; and differential equations which govern state evolutions can be derived from tire force models and kinematic equations [67].

Decision making agent: The human decision making agent can be viewed as an integration of many different computational modules that internally pass information with each other, as shown in Figure 2.1. Feedback from the system will be provided by the *state observer* where the states of interest are estimated, and the *controller* algorithm will compute the set of required control decisions to pass to the *actuator* in order to achieve a set of goals or set-points provided by the *reference generator*. The details of each modules are discussed as follows.

State observer: In order for the controller to accurately track a desired reference, the controller constantly needs information updates on the current states of the system. Once the current states are compared to the desired values, the controller can compute the next input. To obtain these state estimates, the agent needs the means to measure and observe the system states. Various types of sensors can be used to measure the states of the system. For example, in an autonomous vehicle, inertial measurement units are used to measure the vehicle accelerations, and estimate the velocities. In addition, in order to map out the states of the environment, lidar and radar systems are used to detect the states of the surrounding vehicles. Similarly, human agents are born with sensors in order to observe the physical world around us. Sensory feedback from our eyes, ears, tactile sensors on our skins etc [68], are all integrated to obtain the optimal state estimates.

However, the sensors that are used do not always have a direct and accurate representation of the states of interest. Often times, the sensor signals will be corrupted by various level of noise. For example, for human agents, these noise could be introduced from the sensory input or in the neural connections [52]. In order to reconstruct the best estimate of these noisy signals, various filtering and state estimation algorithms can be used. Kalman filtering is a very popular method that is widely used in many control systems [69]. It can also be used to model how human agents perform state estimation in the presence of noise [63]. Other methods such as Bayesian inference [43] are also widely researched to represent human inductive reasoning in the presence of uncertainty.

Reference generator The reference generator refers to the parsing and breaking down of a high-level goal to smaller lower-level subgoals. For example the goal of driving safely from point A to point B will be parsed into smaller goals such as route direction and following the curve of the

highway etc. The references can be generated by a higher-level path planner as in [67], or learned from experienced and retrieved as memory [70].

Controller algorithm The controller module represents the agent or algorithms that is controlling and reacting to evolutions of the system in order to achieve certain goals. There is a set of goals or set-points that the agent wants to achieve, provided by the reference generator. Upon receiving the state feedback of the system, the controller will compute the necessary inputs to apply to the system in order to achieve these goals. The control algorithm will vary accordingly depending on the structure of the system, the goals, and any uncertainties or constraints. For example, for linear systems, it is relatively easy to design a state feedback controller that ensures stability of the system, and convergence to the setpoint. But if there are uncertainties in the state feedback, then one might want to use the linear-quadratic-gaussian formulation to control the system [69]. Similarly, if there are constraints in the state and inputs, then one might use a model predictive control (MPC) approach [71]. We can treat the human subject as the controller when interacting with a dynamic system such as driving a car. There are certain destinations that the human wants to reach, constraints such as not deviating from the lane on the highway, and uncertainties such as the behaviors of other drivers that is taken into consideration.

Actuator: The actuator of the system take input commands from the controller and physically interact with the system. For example, the actuators in an autonomous vehicle would be the mechanical systems of the active steering, braking and throttling. In a driver, it could be the actual neuromuscular system of the arms which interacts with the steering wheels of the car [72]. Often times, there would be discrepancies between the intended control and the actual output of the actuator due to uncertainties in the perceived model of the actuator [64]. Humans account for this uncertainty in the forward model with sensory feedback to determine the necessary corrective actions [63].

2.1.2 Characteristics of the controller model

To model the human agent's thought processes in a dynamic environment, such as decision making, control, environment assessment and state estimation, it can be posed as a black box which maps input to output. The definition of the input and output features will depend on the process that is being modeled. The realm of problems modeled include all areas of human behaviors that have been rigorously studied in human factors, psychology and cognitive sciences. As discussed before, these can be broken down into different levels of abstraction, from low-level muscular control to high-level decision making such as path planning.

Inductive reasoning is one example that has been heavily studied in the field of psychology. It involves combining noisy observations with prior knowledge in order to make inferences about the world. For example, probabilistic Bayesian methods like the Rao-Blackwellized particle filter was used to model state estimation and human visual attention during multiple object tracking [57]. Methods like these can be used to model the state observer component of the framework shown

in Figure 2.1. In this thesis, we would like to focus on the controller component of the decision maker, which maps state estimations of the world to the necessary control commands in order to achieve a desired goal. We will now discuss the input, output and the policy map in detail with examples.

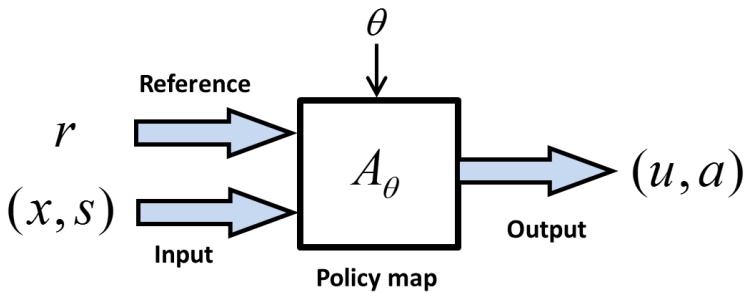


Figure 2.2: Controller component in the decision making framework showing input, output and policy map.

Input vector: As shown in Figure 2.1 and 2.2, the input vector consists of observations and state estimations of the world, as well as goals or references. The reference r can be state trajectories such as X-Y coordinates in a path following problem [67], or a desired set-point. State observations are divided into continuous and discrete variables, $x \in \mathbb{R}^{n_c}$ and $s \in \mathbb{N}^{n_d}$ respectively. For example, in driving situations, continuous observations can include the relative velocity and position between vehicles, and discrete observations can include the signals the other vehicles gives out, like brake lights, blinkers etc.

Output vector: The output vector represents decisions or control commands which, depending on the level of abstraction captured by the model, will be inputs to another process module. Similarly, the output vector can be divided into continuous and discrete variables, $u \in \mathbb{R}^{m_c}$ and $a \in \mathbb{N}^{m_d}$ respectively. Continuous variables are typically used to represent lower-level decisions i.e. neuromuscular commands of the agent [72], hand trajectories in reaching [73], or steering wheel commands to the vehicle [33]. On the other hand, discrete outputs typically refer to higher-level decisions i.e. lane change decisions and turn signals [31], or attention allocation during driving [74].

Policy map: The policy map $A_\theta(\cdot)$, parameterized by θ, maps the input to output. θ is initially unknown and its values are identified and assigned during system identification. Model parameters of the human subject can be time-varying, especially during the learning stage when the subject is still adapting his policy to new scenarios. The structure of the policy map and how it is parameterized by θ will depend on the modeling approach and the nature of the variables involves i.e. discrete or continuous.

To computationally represent the policy map, we will introduce two main modeling approaches: direct method and optimization-based method. Direct methods are associated with reflexive and repetitive processes which involves minimum cognitive processing. Optimization-based methods on the other hand, involve state predictions and internally finding a solution which meets a set of criteria, and therefore require much more cognitive processing.

2.2 Direct methods to modeling human decision making

The first approach to modeling decision making is the direct method where the policy maps are stored in memory, and making a decision corresponds to looking up a learned function map given the observations. Direct methods typically involve minimal computational effort, unlike the optimization-based policies. Following this characteristic, direct methods naturally becomes a suitable candidate for modeling behaviors which appear to be more repetitive, reflexive, and associated with lower-level decision making problems, which will be discussed with application examples later in Chapter 4.

The function maps can also appear as the end product of many policy search algorithms where the optimal behavior is computed off-line and simplified into direct feedback policies to reduce the computational complexity of online implementations, i.e. in linear quadratic regulators [69], or explicit model predictive control [75]. Since we are focusing on modeling the cognitive processes involved with online decision making, policies computed this way are still categorized under the direct method even though the original formulations are optimization-based,

We will present the model formulations for: continuous decisions using the linear feedback model and mode-based models which includes the switched piecewise affine model and the probabilistic clustering model; and discrete decisions using the support vector machine framework. We will also discuss how the parameters in each model can be identified using various state-of-the-art identification algorithms.

2.2.1 Linear feedback map

The most widely used method for modeling continuous decisions is the linear feedback model, and the continuous mapping from input to output $f : x \to u$ is simply an affine mapping $u = Ax + b$. The model parameters for the affine map are $A \in \mathbb{R}^{n_c \times m_c}$ and $b \in \mathbb{R}^{m_c}$. This is the simplest form of the direct methods and is often used in many applications for both its simplicity in computation and identification. For example, driver steering behaviors models can be adapted easily online as the driver is still driving [33]. The variables of the linear matrix A and b can be easily identified with collected data using least squares methods or methods with L1 regularization such as LASSO [76].

Time-varying linear feedback models $u = A(t)x + b(t)$ can be used to capture the adaptive natural of human behaviors. An example of this could be when a driver is learning how to drive a car, or even testing out the new car, the amount of steering wheel angle the driver inputs into the system will change over time as the driver becomes more familiar with the sensitivity of this par-

ticular vehicle. As mentioned previously, if the mapping is linear, recursive least squares methods can be used to find this time-varying parameter online [77]. The idea of a time-varying component in the model parameter is not exclusive to feedback gains but can be applied to any model structure which takes the adaptive natural of the parameter into account. To avoid repetitions, we will define the models as time-invariant, but keep in mind, it is easy to introduce the time-varying adaptation to the framework as well.

2.2.2 Mode-based models

The linear feedback method first introduced in this section focused on mapping continuous input features directly to the output features through a single mapping function. This is a good way to model linear behavior or even linear approximations to nonlinear phenomena in the simplest and quickest way possible. However, given the complexity of human behaviors in many applications, a simple linear map may not be sufficient to capture its observed nonlinear and/or probabilistic nature. Therefore, higher fidelity models are required to explore these aspects. One approach is to use either simple intuition or machine learning to divide the state space and cluster similar behaviors into modes, within which, simpler relationships can be used to fit the observed behaviors.

Generally speaking, for the mode based framework, the policy selection is two-fold. First, the mode of operation is selected, then the output is determined by the policy map which corresponds to that mode. Through the described structure, each mode-based model will need to define the following,

- $\mathcal{M}$ mode of operation
- $\sigma = g(x,s)$ mode selection map
- $u = f_\sigma(x,s)$ policy map based on mode

The policy map $f_\sigma(\cdot)$ can be of any model structure. The simplest form would be to adopt the linear feedback model in Section 2.2.1 as the policy map. The mode selection map can be any function that maps observations to the corresponding mode of operation. We will introduce two mode-based models: piecewise affine function and cluster-based probability distributions. Other examples of mode-based maps include hybrid systems [78].

Piecewise affine models

The first mode based model will be a natural extension to the linear feedback maps. Often times direct feedback methods are sufficient to model a linearized version of a nonlinear relationship if the nonlinearity is not too extreme. However, for behaviors that are highly nonlinear, a simple linear gain will fail. The piecewise affine (PWA) framework can be used to approximate a nonlinear map with smaller segments of linear maps. Different policy maps are simultaneously stored and selected depending on the values of the input features. We will refer to the different policy maps as modes. Values of the discrete input can directly specify which mode to operate, or the continuous input features can be mapped to match the mode conditions.

For the PWA model, the policy map $f_\sigma(\cdot)$ defined in Section 2.2.2 will be the linear feedback models introduced in 2.2.1. The mode selection map can be as simple as having the discrete values indicate the mode, or the continuous variable can be divided into intervals and corresponds to different modes. For example $\sigma = i$ if $x \in [a_i, b_i]$. More complex mode selection maps could involve polytopic computations such as those used in explicit MPC. These are all generic examples of piecewise affine feedback models. Gain-scheduling is another example where the feedback gains are pre-computed, stored and selected based on the values of other variables. We present a special class of PWA models, namely, the *PieceWise AutoRegressive eXogenous* (PWARX) formulation, where the switching mechanism is determined by a polyhedral partition of the regressor domain [79].

Model Formulation: For fixed model orders n_a and n_b, the regressor r_k is defined in terms of the input vectors $u_{k-i} \in \mathbb{R}^p, i = 0, 1, ..., n_b$ and past output vectors $y_{k-i} \in \mathbb{R}^q, i = 1, ..., n_a$,

$$r_k = [y_{k-1}^T \ ... \ y_{k-n_a}^T \ u_k^T \ u_{k-1}^T \ ... \ u_{k-n_b}^T]^T \tag{2.1}$$

The current output y_k is expressed as a piecewise affine function of r_k,

$$y_k = \theta_{\sigma(k)}^T \begin{bmatrix} r_k \\ 1 \end{bmatrix} \tag{2.2}$$

where $\sigma(k) \in \{1, ..., \mathcal{M}\}$, is the discrete state, $\mathcal{M}$ is the number of modes. $\{\theta_i\}_{i=1}^{\mathcal{M}}$ is the matrix of parameters defining the system dynamics in each mode. The domain of the discrete modes are determined by a polyhedral partition of the regressor domain $\mathcal{R} \subset \mathbb{R}^d$, where $d = q \times n_a + p \times (n_b + 1)$. The discrete state $\sigma(k)$ is given by

$$\sigma(k) = i \quad \text{iff} \quad r_k \in \mathcal{R}_i \quad i = 1, ..., \mathcal{M} \tag{2.3}$$

and $\{\mathcal{R}_i\}_{i=1}^{\mathcal{M}}$ is a complete partition of $\mathcal{R}$. Each region $\mathcal{R}_i$ is a convex polyhedron described by

$$\mathcal{R}_i = \{r \in \mathbb{R}^d : H_i \begin{bmatrix} r \\ 1 \end{bmatrix} \leq 0\} \tag{2.4}$$

Model Identification: For identification of the parameters in the PWA model, both the linear parameter $\{\theta_i\}_{i=1}^{\mathcal{M}}$ in each mode and the polyhedral partitioning of the regressor space $\{\mathcal{R}_i\}_{i=1}^{\mathcal{M}}$ need to be determined in order identify how the modes are defined. Various identification algorithms are discussed in [79], including those which also determines the number of modes in the polyhedral partition. In Chapter 4, we used the Hybrid Identification Toolbox [80] to identify the parameters for our model. As part of tuning, the number of modes need to be predefined prior to running the cluster-based identification algorithm.

Probabilistic maps

The previously described modeling framework for the PWA system is an example of mode-based models which integrates linear policy maps into the different modes of the system. Now we will consider the approach where each mode is directly associated with a probability distribution, and able to handle the vast degree of uncertainty that is often observed in human behaviors.

Model Formulation: To define the structure of the probabilistic mode-based framework, consider for each mode $\sigma \in \{1, ..., \mathcal{M}\}$, there is a set $O_\sigma = \{o_i\}_{i \in \{1,...,|\sigma|\}} \subset O$ of observations associated with each mode, where $|\sigma|$ denotes the number of elements in mode σ. Each o_i is a vector composed of observations. The output of the model $\mathcal{U}$ is a function of the empirical distributions that is constructed for each mode separately. The input of the model I is associated with the current information and determines which mode the input currently belongs to. Formally defined as,

$$\mathcal{U} = \Pi(P(X|O, I)) \tag{2.5}$$

To make a prediction using this modeling framework, first the current observation I is used to determine from O_σ which cluster I belonged to. Then once the mode is identified, the associated probability distribution of that cluster is used in $\Pi(\cdot)$ to compute $\mathcal{U}$. For the function $\Pi(\cdot)$, we can use expected values as an example for computing the output.

Model Identification: Identification of this mode-based probabilistic model involves two steps. First the modes are identified from recorded data using clustering algorithms, then an empirical distribution is constructed within each cluster separately. [81] provides a survey of all the clustering algorithms. For the driver application in Chapter 5, we used a k-means clustering algorithm [82] to identify the modes.

2.2.3 Support vector machine

The methodologies introduced thus far has focused on modeling the policy functions for continuous decisions. We will now introduce another direct method of modeling for discrete outputs, namely support vector machines (SVM) [83], popularly used for many classification problems across a diverse range of applications i.e. pattern recognition [84], gene selection for cancer classification [85], and optical character recognition [86] etc. SVM is a form of supervised learning algorithm which requires the labeling of the training data. The most basic form of SVM is used to classify a two-class problem, although many extensions has been proposed for using SVM in multi-class problem as well [87].

Model Formulation: Given a set of data where for each data point i, it is labeled with $y = \{-1, 1\}$ depending on which one of the two-class the point belonged. We are interested in finding a function $y = f(x)$ which maps the feature vector $x \in R^n$ into the output $y = \{-1, 1\}$. SVM uses a hyperplane as the function,

$$f(x) = (w \cdot x) + b \tag{2.6}$$

and interprets positive or negative values of the output to be corresponding to be belonging to that class.

For non-linear cases where the hyperplane fails to classify, the feature vector of the data set x can be preprocessed with a nonlinear map $x \to \Phi(x)$ and now the classification map becomes,

$$f(x) = (w \cdot \Phi(x)) + b \tag{2.7}$$

Model Identification: Given a data set $\mathcal{D}$ consisting of m data points where each point consists of a feature vector x and classification label y, we are interested in finding the values of w and b which minimizes classification error. Consider the loss function $L(y, \hat{y})$, where the value is either 0 or 1 depending on whether $y = \hat{y}$, the hyperplane which classifies the most number of points from $\mathcal{D}$ correctly minimizes the loss function, formally defined as,

$$\min_{w,b} \quad \frac{1}{m} \sum_{i=1}^{m} L(w \cdot x_i + b, y_i) + \|w\|^2 \tag{2.8}$$

Without going too much into the details, Equation (2.8) can be reformulated into a quadratic programming problem with relaxed constraints, and loss function converted to a hinge-loss function. Other tricks, such as the use kernel functions, can handle nonlinearity in the data. [83] details the state-of-the-art methods used in identification of SVM. In addition, the SVM field of machine learning is very well-established and there exists many toolboxes available to use and interfaces with many different programming languages. LibSVM [88] is one toolbox that is widely used.

2.3 Optimization-based methods to modeling human decision making

Optimization-based models are policies which require computations that tries to find the optimal solution. The modeling framework is posed as an optimization problem,

$$\min_{x} \quad J(x) \tag{2.9a}$$

$$\text{subj.to} \quad f(x) = 0 \tag{2.9b}$$

$$g(x) \leq 0 \tag{2.9c}$$

where $J(x)$ in (2.9a) is the cost function, or reward function for maximization problems, $f(x)$ in (2.9b) is the equality constraint, and $g(x)$ in (2.9c) is the inequality constraint. In the context of the human controller, the optimization variable x is the output vector of the policy map.

This type of formulation, termed optimal control, is a systematic way of generating an open-loop set of control sequence that are optimal for a particular cost function. It is widely used in control systems where the control objective is to minimize some cost or measurement criterion, which could be a function of the states, and/or control inputs, for example, error from reference trajectory. The predicted effects of control inputs/decisions on future state evolutions are also

considered, and states are to be kept within certain constraints. The state predictions could either be obtained from a look-up table, from motion primitives [89], or from a model of the system dynamics.

Control examples include control algorithms such as linear-quadratic regulator [69] and model-predictive control [71]. From a human decision making perspective, optimization models seems to naturally fit into the higher-level decision making frameworks. For many decision making problems, we are concerned about optimizing a certain criterion, it could be time, energy, effort, or error [64]. It could be maximizing the amount of reward you would obtain from a certain task as well. We would refer to these problems as more of a planning problem that requires more thinking, planning and therefore, more computational power from the human. So naturally we would like to model these type of behaviors using the predictive model framework to match the way our mind interacts with the environment.

However, this approach of modeling is very difficult to identify from collected data, since it cannot be easily formulated into a machine learning problem like least squares etc. There are many variables within the optimization framework that can be used as model parameters. For example, the optimization objective structure and the respect weights, the parameters in the constraint formulations. On the other hand, the direct mapping of functions in Section 4.1 is much easier to formulate into a standard model identification problem, and there exists many tools and algorithms that processes the data and identify the parameters efficiently.

2.3.1 Model predictive control

A specific case of the optimization model is model predictive control (MPC), or receding horizon control. MPC takes the model of the system dynamics, typically a set of differential equations, to generate future state predictions. From an autonomous control perspective, MPC can be implemented to avoid obstacles [67]. In Chapter 5 we will extend this MPC framework to include predictions of human behaviors. From a human modeling perspective, learning to behave like an optimal controller involves learning the cost function (i.e. reference trajectory), and also learning either the motion primitives or the system dynamics (mental model) of the activity. We will focus on modeling the driver using the latter abstraction. To formally express this framework, we consider the following optimization problem:

$$\min_{U_t} \quad J(\bar{\xi}_t, U_t, \Delta U_t) \tag{2.10a}$$

$$\text{subj. to} \quad \xi_{k+1,t} = f(\xi_{k,t}, u_{k,t}) \quad k = t, ..., t+H_p-1 \tag{2.10b}$$

$$\Delta u_{k+1,t} = u_{k+1} - u_k \quad k = t, ..., t+H_p-2 \tag{2.10c}$$

$$u_{k,t} \in \mathcal{U} \quad k = t, ..., t+H_p-1 \tag{2.10d}$$

$$\Delta u_{k,t} \in \Delta\mathcal{U} \quad k = t+1, ..., t+H_p-1 \tag{2.10e}$$

$$\xi_{t,t} = \xi(t) \tag{2.10f}$$

$$\xi_{N,t} \in \mathcal{X}_f \tag{2.10g}$$

H_p is the prediction horizon. $\bar{\xi}_t = [\xi_{t,t}, \xi_{t+1,t}, ..., \xi_{t+H_p-1,t}]$, $\bar{\xi}_t \in \mathbb{R}^{n \times H_p}$ is the sequence of states predicted at time t, based on the Euler discretized dynamics of the vehicle model (2.10b). $u_{k,t} \in \mathbb{R}^{m_r}$ (m_r is the number of inputs at each time instance) is the k^{th} column of the input sequence matrix $U_t = [u_{t,t}, u_{t+1,t}, ..., u_{t+H_p-1,t}]'$. The cost function (2.10a) is defined as

Modeling the policy map for human decision making as an MPC problem has two implications. First it suggests that humans make decisions based on predicted states in the future for an extended horizon. Second it suggests that humans are able to solve these complex optimization problems to find the optimal solution in real-time. Much similar to an MPC formulation that is applied to vehicle control systems and general solvers [90], the computational complexity can be too cumbersome to perform efficiently in real-time. So often times, we compromise for sub-optimal solutions found after a few search iterations. Alternatively, explicit MPC [75] finds polytopic solutions offline similar to the PWA model in Section 2.2.2. We will focus on the online computation of solutions in the examples later in this thesis.

2.3.2 Markov decision process

Another extension of optimization models is the Markov decision process (MDP), where similar to the MPC, it involves state transitions and minimizing cost (or maximizing reward) that are a function of the states and inputs, but different from the continuous nature of MPC, the MDP framework deals with discrete states and discrete actions/inputs instead. Unlike MPC which finds optimal solutions to the original formulation online, the discrete nature of the MDP model and the finite set of states and actions, allows designers to compute and store value functions of the states offline. The online implementation involves solving just a one-step optimization problem which maximizes the value of the next state. Intuitively, this makes a connection to how valuable or costly it is to be in certain states, and aligns with the skill acquisition and learning aspect of human behaviors, where after becoming familiar with the nature of the task, the subject learn from experience a preference towards being in specific state configurations in order to maximize completion of his goal.

A (finite) MDP is a tuple (S, A, T, R, γ) where,

- $\mathcal{S} \in \mathbb{N}^n$ is a finite set of **states**
- $\mathcal{A} \in \mathbb{N}^m$ is a finite set of **actions**
- $T : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0,1]$ are the state **transition probabilities**
- $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the **reward function**
- $\gamma \in [0,1]$ is the **discount factor**

In solving for the MDP, the goal is to choose the policy $\pi(s)$ which maximizes the cumulative discounted sum of the reward given the state transitions, formally defined as,

$$\pi(s) = \arg\max_a \sum_{t=0}^{\infty} \gamma^t R(s_t, s_{t+1}) \quad \text{subj.to} \quad s_{t+1} = T(s,a) \tag{2.11}$$

Optimal policy: There are many algorithms [91] which solve for the optimal policy $\pi(s)$ for Equation (2.11), i.e. through dynamic programming, value iteration, policy iteration etc. We will present the value iteration approach. In value iteration, the value function $V(s)$ at each state is found by iterating the bellman equation until convergence.

$$V(s) = \max_a R(s,a) + \gamma \sum_{s'} T(s'|s,a)V(s') \tag{2.12}$$

where s' is the next state. We denote the optimal value function as $V^*(s)$. The optimal policy $\pi^*(s)$ is evaluated as the action that achieves the highest reward.

$$\pi^*(s) = \arg\max_a \sum_{s'} T(s'|s,a)[R(s,a) + \gamma V^*(s')] \tag{2.13}$$

Once the optimal policy is evaluated, at any given state s, we can determine the optimal action to take. This optimal action is later compared with the actions of the participants.

Inverse reinforcement learning: The inverse problem, where the reward function which resulted in a given set of observed actions, can be identified through inverse reinforcement learning [92]. The goal of this problem is to find a reward function $R^*(s)$, such that,

$$\mathrm{E}[\sum_{t=0}^{\infty} \gamma^t R^*(s(t))|\pi^*] \geq \mathrm{E}[\sum_{t=0}^{\infty} \gamma^t R^*(s(t))|\pi] \quad \forall \pi \tag{2.14}$$

In other words, the policy chosen by the participant throughout the experiment should be his/her optimal policy, therefore the expected reward of this policy should be greater than all available policies that were not chosen. We use a feature based method and translate the reward as a linear combination of features, $\phi(s)$. By the term *features*, it refers to what we "expect" the participant to use in his/her decision making. Letting $R(s) = w^T \phi(s)$, the expected reward becomes,

$$\mathrm{E}[\sum_{t=0}^{\infty} \gamma^t R^*(s(t))|\pi] = w^T \mathrm{E}[\sum_{t=0}^{\infty} \gamma^t \phi(s(t))|\pi] = w^T \mu(\pi) \tag{2.15}$$

Now solving Equation (2.14) is equivalent to finding,

$$w^* \quad s.t. \quad w^{*T}\mu(\pi^*) \geq w^{*T}\mu(\pi) \quad \forall \pi \tag{2.16}$$

As discussed in [93], in order to find a feasible solution that is not the trivial solution of $w = 0$, the max-margin with slack variables approach is implement. The optimization problem becomes,

$$\min_w \quad \|w\|_2^2 + C\xi \tag{2.17a}$$

$$\text{subj. to} \quad w^{*T}\mu(\pi^*) \geq w^{*T}\mu(\pi) + m(\pi^*,\pi) - \xi \quad \forall \pi \tag{2.17b}$$

where $m(\pi^*,\pi)$ is a structured prediction margin that are larger for policies which are very different from π^*. ξ are a collection of slack variables that relaxes the constraints.

2.4 Conclusions

In this chapter, we drew from the field of controls and machine learning to arrive at the frameworks and methodologies that can potentially be used to model human decision making in a dynamic environment. From a control-theoretic approach, we presented a framework which paralleled the human decision making agent as process module consisting of multiple interconnecting sub-modules. Included in the submodules are: the state observer for estimating the state feedback of the system; the reference generator responsible for planning a set of goals; the controller for determining the set of decisions which achieves the goals; and actuators associated with the physiology of the human.

Focusing on the controller module, we presented two approaches of modeling the policy map of the control strategy. The direct method models stored function maps in memory, and maps input to output directly. We presented examples of the direct method, specifically the linear feedback model, mode-based PWA and probabilistic models, and SVM for modeling discrete decisions. Along with the various models, we also introduced identification methods that could be used to identify the best set of parameters to fit the recorded data.

Even with the ease of parameter identification, direct methods lack the capacity to model the planning and predictive natures of cognitive process which involve high-level planning. Optimization-based methods are introduced to capture this cognitive phenomenon, where state predictions are simulated in order to find a suitable solution to meet a set of criteria, including cost function and constraints. Extensions of the optimization-based methods included MPC which handles continuous decisions and MDP for discrete decisions. Due to the modeling structure of optimization-based methods, there are no clearly defined algorithms to handle the general parameter identification problem, and only for very specific contexts. For example, reinforcement learning can be used to extract the weights of the reward function in the MDP.

We will use the framework and formulations presented here to apply to specific human modeling examples in Chapter 4 where we explore a range of human behavior from game-based applications to real-world driving. In Chapter 5, we will use the methodologies discussed here to present a unifying framework for integrating human models with controller design. In particular, we will use MPC to include predictions of driver behavior in semi-autonomous vehicles.

Chapter 3

Description of Experimental Setups, Test Scenarios and Dynamics of Associated Systems

With the various modeling methods laid out in the previous chapter, we will now describe in detail the experiments that are performed as examples of human behavioral modeling applications. The application context will be derived from the two research motivations emphasized in Chapter 1. In one direction, we will address the interest of modeling human behaviors for the purpose of improving human-machine intelligent systems. Specifically, we will focus on the modeling of driver behavior with the intention of applying the resulting model into an autonomous driving framework. We will approach the other part of the motivation, which is to use the proposed computational modeling frameworks to better understand the mechanism and factors which affect human decision making, from the psychological perspective. Therefore, in addition to the analysis of data collected from the context associated with our everyday lives such as driving, we will also scale down and simplify the problem scenario in order to conduct repeatable and controllable experiments, collect consistent data, and focus on a smaller subset of principles.

In the driving section, we start off with the simplest case of everyday driving, which includes lane keeping, lane changing and curve negotiation. Then we introduce complex driving maneuvers performed in slippery conditions which requires the knowledge that tire-road dynamics have both linear and saturated regions [94]. This allows us to investigate how drivers can handle operations in this nonlinear regime. The last example will study the behavior of drivers in a distracted scenario, i.e. texting while driving, in order to derive an improved semiautonomous system.

In the scaled down scenarios performed in a controlled lab setting, we will study the decision making in multiple game-like experiments. We will use a projectile shooting game to illustrate how subjects handle an environment with nonlinear relations affecting the trajectory of the related states, similar to the complex driving setting which involves nonlinear tire dynamics. We will also attempt to parallel the distracted driving scenario with a dual-task game. Lastly, we will use the example of juggling to examine the subject's ability to simulate real-time forecasts in a dynamic task. The fusion of the analysis from experiments conducted both in the real-world setting and the

lab setting ties together the psychology approach of focusing on the underlying principles from parametric models with real-world application for control design.

This chapter is structured as follows. First we present all the experimental details associated with the driving context, which include: the coordinate system associated with the vehicle dynamics and the surrounding environment; the experimental setup of the vehicle including sensors and actuators; the experimental setting including the test centers and virtual driving; the sensor measurements to monitor human states; and lastly, detailed discussion of the previously introduced driving scenarios. The second part of this chapter presents the experimental setup of the dynamic task of juggling, and the projectile shooting and the dual-task game.

3.1 System dynamics, experimental setup and data collection for driver modeling and control

In this section, all the experimental setups and test scenarios designed for understanding driver behaviors will be discussed. We will first present the system dynamics associated with the vehicle, then detail the on-road vehicle setup that is used to collect driver behavior on real roads and actual road driving. Then we will describe the simulation setup used to record virtual driving in a lab setting, followed by the equipment and methodologies used to estimate the internal and external states of the human subject, and finally, outline the different driving scenarios that is incorporated into the experiments.

3.1.1 Vehicle dynamics and tire model

Figure 3.1 illustrates a vehicle system and shows the conventions of some of the associated variables. These are listed below for ease of reference,

- X, absolute X position in the global coordinate framework
- Y, absolute Y position in the global coordinate framework
- $\dot{x}$, longitudinal velocity in the body coordinate framework
- $\dot{y}$, lateral velocity in the body coordinate framework
- $\ddot{x}$, longitudinal acceleration in the body coordinate framework
- $\ddot{y}$, lateral acceleration in the body coordinate framework
- ψ, yaw angle / heading angle in the global coordinate framework
- $\dot{\psi}$, yaw rate / rotational velocity about the vertical axis
- $\ddot{\psi}$, rate of change in yaw rate / rotational velocity about the vertical axis

- δ, steering angle of front wheels
- $F_{x,i}$, forces applied to vehicle in the longitudinal direction from tire i
- $F_{y,i}$, forces applied to vehicle in the lateral direction from tire i
- $F_{l,i}$, longitudinal tire forces of tire i from the tire model
- $F_{c,i}$, cornering tire forces of tire i from the tire model
- v_i, velocity vector of tire i
- α_i, side slip angle of tire i
- e_y, lateral deviation from center of lane
- e_ψ, heading deviation from curvature of preview point
- d_ψ, distance to preview point
- $x_{r,i}$, relative distance to the i-th vehicle
- $\dot{x}_{r,i} = \dot{x} - \dot{x}_i$, relative speed to the i-th vehicle

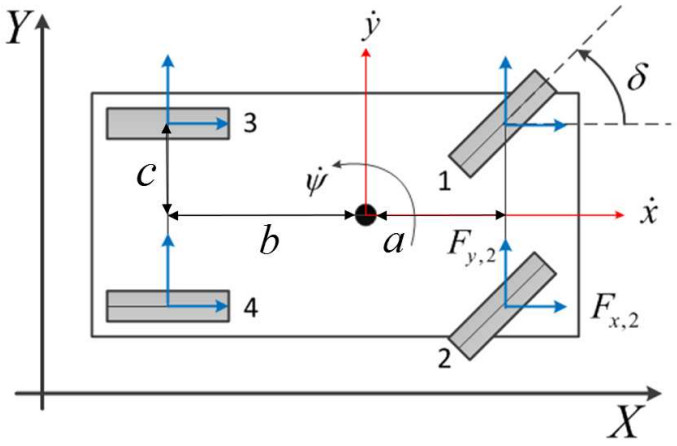


Figure 3.1: Four wheel vehicle model illustrating the vehicle states.

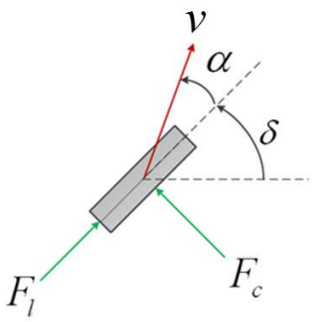


Figure 3.2: Longitudinal and cornering tire forces and side slip angle.

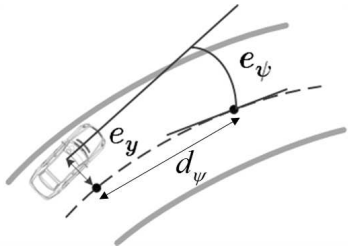


Figure 3.3: Deviation from lane center and curvature.

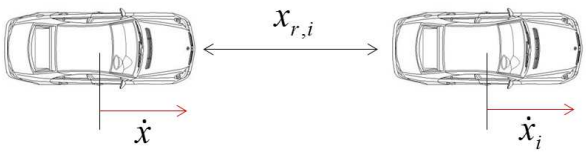


Figure 3.4: Relative velocity and distance with neighboring vehicles.

Vehicle model for controller design: Used in the semi-autonomous controller, as discussed later in Chapter 5, we introduce the system dynamics of the four wheel vehicle model [67], which is a six state nonlinear model $\dot{\xi}(t) = f(\xi(t), u(t))$ capturing the lateral, longitudinal and yaw dynamics. The state vector and input vector is defined as $\xi = [\dot{y}, \dot{x}, \psi, \dot{\psi}, Y, X]'$ and $u = \delta$ respectively.

The equations of motion w.r.t. the center of gravity of the vehicle are:

$$m\ddot{y} = -m\dot{x}\dot{\psi} + F_{y,1} + F_{y,2} + F_{y,3} + F_{y,4} \tag{3.1a}$$
$$m\ddot{x} = m\dot{y}\dot{\psi} + F_{x,1} + F_{x,2} + F_{x,3} + F_{x,4} \tag{3.1b}$$
$$I\ddot{\psi} = a(F_{y,1} + F_{y,2}) - b(F_{y,3} + F_{y,4}) + c(-F_{x,1} + F_{x,2} - F_{x,3} + F_{x,4}) \tag{3.1c}$$
$$\dot{Y} = \dot{x}\sin\psi + \dot{y}\cos\psi \tag{3.1d}$$
$$\dot{X} = \dot{x}\cos\psi - \dot{y}\sin\psi \tag{3.1e}$$

where m is the vehicle's mass, I is the rotational inertial about the yaw axis, a, b and c are distances between the tires and the center of gravity. The tire forces, $F_{x,i}, F_{y,i}$ are modeled by using a nonlinear Pacejka tire model [94].

Nonlinear characteristic of tire model: Using the Pacejka tire model [94], we introduce the nonlinear relationship between the vehicle states and the tire forces. All the parameters required for the Pacejka tire model i.e. road friction coefficient, normal force, tire slip ratio are assumed to be constant, except for α which depends on $\dot{y}, \dot{x}, \dot{\psi}$ and δ [67].

$$\alpha_i = \arctan \frac{v_{c,i}}{v_{l,i}} \tag{3.2}$$

where $v_{c,i}$ and $v_{l,i}$ are the lateral and longitudinal wheel velocities computed from,

$$v_{c,i} = \begin{cases} (\dot{y} + a\dot{\psi})\cos\delta - (\dot{x} - c\dot{\psi})\sin\delta & \text{if } i = 1 \\ (\dot{y} + a\dot{\psi})\cos\delta - (\dot{x} + c\dot{\psi})\sin\delta & \text{if } i = 2 \\ \dot{y} - b\dot{\psi} & \text{if } i = 3,4 \end{cases} \tag{3.3}$$

$$v_{l,i} = \begin{cases} (\dot{y} + a\dot{\psi})\sin\delta + (\dot{x} - c\dot{\psi})\cos\delta & \text{if } i = 1 \\ (\dot{y} + a\dot{\psi})\sin\delta + (\dot{x} + c\dot{\psi})\cos\delta & \text{if } i = 2 \\ \dot{x} - c\dot{\psi} & \text{if } i = 3 \\ \dot{x} + c\dot{\psi} & \text{if } i = 4 \end{cases} \tag{3.4}$$

Figure 3.5 shows the lateral tire force F_c as a function of α under different road friction coefficients. Notice that the tire forces can be divided into a *linear region* where the relationship between α and F_c is approximately linear and a *saturated region* where F_c maintains roughly constant even when α is changing. Also note that with lower friction coefficient, the linear region is narrower, which makes driving on slippery surfaces extremely difficult for a novice.

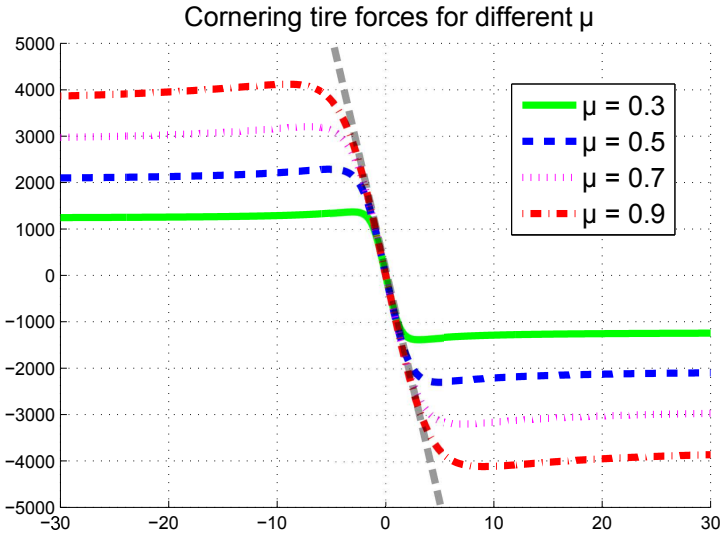


Figure 3.5: Cornering tire forces as a function of side slip angle α for different friction coefficient.

3.1.2 Equipment setup on the test vehicle

In collaboration with Ford Motor and Hyundai Motors, we have used two different manufactures test vehicles on separate occasions to record driver behavior. The vehicles are equipped with the following instruments for data collection and control.

Figure 3.6: Test vehicles provided by Ford Motors and Hyundai Motors.

Oxford Technical Solution sensing system: We used an Oxford Technical Solution (OTS) RT3002 sensing system to measure the position and the orientation of the vehicle in the inertial frame and the vehicle velocities in the vehicle body frame. The OTS RT3002, is housed in a small package that contains a differential global positioning system (GPS) receiver, Inertial Measurement Unit (IMU), and a digital signal processor (DSP). It is equipped with a single antenna to receive GPS information. The IMU includes three accelerometers and three angular rate sensors. The DSP receives both the measurements from the IMU and the GPS, utilizes a Kalman filter for sensor fusion, and calculates the position, orientation and other states of the vehicle such as longitudinal and lateral velocities.

Wheel rotation sensors: The wheel rotation sensors gives measurements of the rotational speeds of the wheels. This combined with the measurements from the IMU gives a state estimate of the longitudinal velocity.

MobilEye camera system: The MobilEye camera system is widely used by many motor vehicle companies in order to provide information for their on-board smart autonomous and driver assistance systems i.e. Tesla's autopilot system [95]. The camera system overlooks the road conditions directly in front of the vehicle, records and processes the images internally with their proprietary algorithm. It monitors and detects proceeding vehicles, pedestrians, and lane information. The results and outputs of the MobilEye camera include lane information i.e. lane curvature, deviation from center of lane e_y, and the relative distance and speed to the proceeding i-th vehicles, $x_{r,i}, \dot{x}_{r,i}$. The MobilEye is a powerful system to use in monitoring the environment, and is easy to install and readily provides processed information.

3.1.3 Test centers

Our driving scenarios on the real vehicle were performed in two different test centers. One for extreme conditions with snow tracks, and the other for normal driving conditions.

Figure 3.7: Hyundai-Kia Motor California Proving Ground [96] (left) and Smithers Winter Test Center (right).

Hyundai-Kia Motor California Proving Ground (CPG): Located in the middle of the Mojave Desert, the CPG test center provides researchers and engineers with various driving tracks to collect driving data and test algorithms. Shown in Figure 3.7, label 3 and 4 indicates the high-speed oval loop which provides the environment for highway driving, and at label 5 is the winding track which is a serious of twist and turns with asphalt surfaces.

Smithers Winter Test Center: In collaboration with Ford Motors, the Smithers test ground is used during winter reason to perform driving under cold and slippery conditions, especially on icy and snow packed surfaces, typically at a friction coefficient of 0.2-0.4.

3.1.4 Virtual driving on the simulator

Driving behavior on a real vehicle is ideally the type of driving setup we want to be collecting information from because it represents the true sensory feedback a driver will experience while driving, however, there are certain driving scenarios and environmental conditions we would like to test which cannot be safely or practically performed on-road, for example, texting while driving, or the sudden appearance of obstacles. In addition, sometimes it is useful to be able to control the exact feedback and environmental conditions the subject will be receiving in order to record repeatable and consistent data. Therefore, specific test scenarios are performed with a virtual setup and simulated driving.

The CarSim software provides a realistic driving world with full dynamics of the car, as well as customizable environments in terms of road layout, road conditions, and the presence of other vehicles. Figure 3.8 provides a screenshot of the type of visual feedback the driver experiences in this simulated driving environment. In addition, we also provide audio feedback to the driver which corresponds to the amount of longitudinal acceleration applied. The driver interfaces with the CarSim simulator through Logitech steering wheel and brake / throttle pedals typically used in gaming.

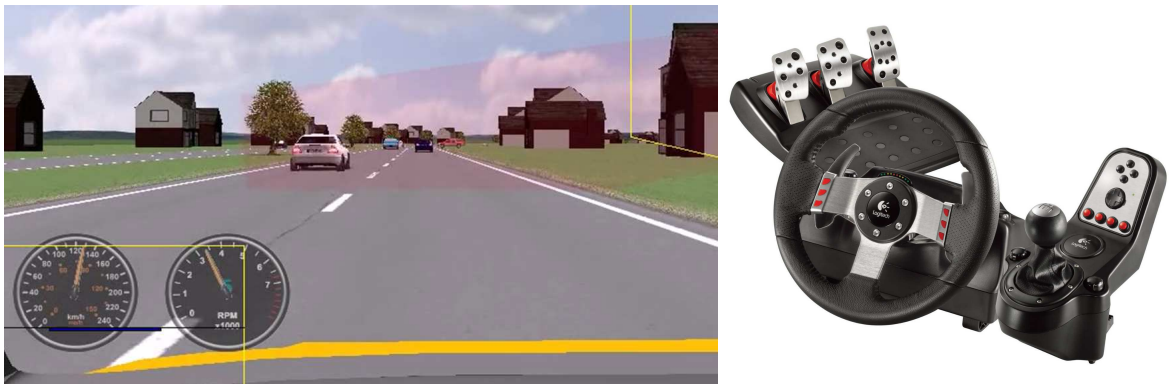


Figure 3.8: Screenshot of CarSim simulator as experience by the subject (left) and control interface using Logitech gaming equipment (right).

3.1.5 Sensors for monitoring human states

The data collection system previously described all focused on obtaining variables which are the sensory feedbacks the human experiences and therefore is only related indirectly to the human subject. Internal states of the subject is often inferred from these measurements. Direct monitoring of both the external body positions and internal mental states provide useful information about the subjects' state of mind.

Eye tracker: Eye tracking provides information about the object in view that the human subject is concentrating on. The point of focus provided by the eye tracker shows the area in space that the subject is currently focusing on. It also provides information about saccade and fixation which are important to detect the switching of attention between different objects.

There are many different eye tracking devices out there, mainly differ in where they are mounted. Desk-mounted eye trackers are good for applications where the task is performed in a limited visual range [97], for example, with experiments which involve a computer screen, the subject is limited to looking at the screen therefore the desk-mounted version is more suitable. On the other hand, head-mounted eye trackers are more invasive and the subject has to physically wear the device, which can be more uncomfortable or may require constant re-calibration when the device shifts. However, the head-mounted eye tracker is usually not limited to a predetermined visual

space since gaze information is calibrated and overlaid on top of image streams from a scene camera. It is suitable for experiments which might involve the subject walking around, or turning of the head, as in our case, in driving scenarios and juggling.

We use the SensoMotoric Instruments (SMI) head-mounted eye tracker, also termed eye tracking glasses, to measure the eye gaze of the subject in driving experiments to see where the driver is looking at and paying attention to. We also use the same eye tracking glasses to record the gaze location of an expert juggler to determine whether forecasts are made. The eye tracking glasses provides data at a sampling rate of 30Hz.

Figure 3.9: SensoMotoric Instruments head-mounted eye tracker [98] (left) and subject wearing eye tracker in an experiment [99] (right).

Skeletal tracking: Skeletal tracking gives us information about the body position, posture, arm, leg and hand positions over time. We can use these information to estimate what the person is doing at certain moments in time. For example, in gaming, skeletal tracking is used to create an interactive and realistic gaming experience where the person uses their whole body to perform certain tasks and therefore would feel a stronger association with real life scenarios [100]. In driving scenarios, we can use skeletal tracking to determine what the driver is doing, for example, texting, hands on the wheel, adjusting onboard controls etc.

There are many ways skeletal tracking can be achieved, i.e. through motion capture sensors [101, 102], or a non-intrusive method with stereo cameras or infrared cameras i.e. Leap Motion. The Microsoft Kinect is one particular examples of using infrared technology to derive 3D point clouds, and has being heavily used over the recent years by both hobbyists and researchers, because of its ease of application, and the vast community of software support developed for the hardware [102]. The Microsoft Kinect used in our experiments has one camera capturing images at 30 Hz, and also an infrared system capturing depth information. The infrared system limits the use of older models to be indoors under controlled lighting. The Skeleton class is used to process raw data and output skeletal tracking in real-time as seen in Figure 3.10.

Figure 3.10: Microsoft Kinect sensor for skeletal tracking (left) and screenshot of real-time skeletal display during tracking (right).

3.1.6 Driving scenarios

With our vehicle apparatus, testing environment and sensor setup described in the previous section, we will now outline the various driving scenarios our test subjects performed in. The driving scenarios include differences in driving conditions, amount of distraction, and goal of the driving task etc. We explain each driving scenario here and detail the modeling process for each scenario later in Chapter 4 and 5. Depending on the scenario, some of the data are collected from driving on actual roads with real traffic conditions and uncontrolled interactions with other vehicles, and some are collected from driving on test grounds or simulators, under controlled traffic conditions, or zero interaction with other vehicle.

Highway driving with lane changing and lane keeping: Highway driving takes up a large portion of what a driver has to experience every day. Fully autonomous driving will take much longer to achieve because of the complication of real traffic conditions in urban environments. However, there are already many highway driving assist features available in commercial vehicles currently out in the market [7], responsible for operating at levels 1 and 2 of vehicle automation as defined by the National Highway Traffic Safety Administration [103]. For example, lane keeping, adaptive cruise control, car following, lane changing, blind spot detection etc.

We asked subjects to perform lane keeping and lane changing, both on an enclosed highway at the CPG and on a real highway with uncontrollable traffic conditions, and record the vehicle states and lane information. With lane keeping, we concentrate on the lower-level control decisions of the steering angle. With lane changing, we concentrate on the higher-level decisions of when the driver will change lanes based on traffic conditions.

Curve negotiation on the winding track: The winding track located at the CPG is used to collect data on how drivers adjust their controls during curve negotiations. This differs from highway driving where it is mostly straight and even the curved sections have large curvatures. On the winding track, we collect additional data using the eye-tracking glasses to determine what visual cues the driver uses to make control decisions, for example, turning the steering wheel and speed adjustments.

Performing extreme maneuvers on slippery roads: Extreme maneuvers refers to driving scenarios where tire dynamics are near the limits of operation. As discussed in the tire model section of 3.1.1 and shown in Figure 3.5, the lateral tire forces are a nonlinear function of the side slip angle and roughly consists of a linear region and a saturated region. We can see from the tire force plots that when the tire slip angles are small, the relationship with tire forces are approximately linear. As the friction coefficient decreases, this linear region shrinks.

Most of the driving performed on everyday roads such as asphalt has high friction coefficients and a typical driver will be able to stay within the linear range. When the driving conditions are harsh, for example on wet or icy roads, or dirt road racing, the friction coefficients are much lower and the linear range is significantly reduced, and therefore the vehicle is much harder to control in slippery conditions

Typical drivers are not trained to handle the vehicle dynamics that are involved with tire saturation. Expert drivers on the other hand, are able to utilize both the linear and saturated regions of tire forces to their advantage, for example, professional dirt road racers will drift around corners to quickly negotiate around a corner without having to reduce too much speed and momentum [104].

For our extreme maneuver scenario, we ask trained drivers to purposefully enter the saturation region and perform a 180 degree turn on icy roads. The data is collected at the Smithers testing center during the winter. Figure 3.11 shows an example of the X-Y trajectory of the vehicle associated with this maneuver.

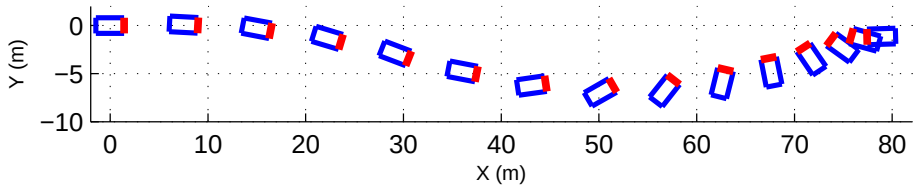


Figure 3.11: X-Y trajectory of 180 degree drift maneuver.

Driving under the influence of distractions: Distracted driving has always been a major concern and factor in accidents. Processing of the environment and reaction times are significantly reduced because of distractions [105]. Anything that the driver does in the vehicle that inhibits him/her from paying full attention on the road can be classified as a source of distraction. For example, fiddling with the on-board controls, talking and/or texting on the phone, even with hands-free devices, it is still a cognitive distraction since the driver is mentally paying attention to the conversation [106]. Texting while driving is especially dangerous because the driver is both not looking at the traffic conditions and also the driver's hands are not physically on the steering wheel.

We tested two distracted driving scenarios, the first one set in CarSim is used to immerse the subject in a busy highway driving scenario with other vehicles nearby and with the sudden appearance of obstacles. The idea is to model, map and predict the future steering and braking behaviors of the driver based on the state of the driver, or the level of *distraction* the driver is

currently under, and integrate this driver model into a semi-autonomous controller. In the second scenario we recorded attention allocation during driving and texting at the CPG.

Texting while driving with sudden obstacles in virtual simulator: The experiments are divided into two sessions: training and controller application. In the training session, we asked the driver to drive for different trials on the CarSim simulator, on some trials, we asked the driver to drive normally with two hands on the wheel, and on some other trials, we asked the driver to respond to a customized messaging application on an Android based smart phone. The phone is preloaded with questions and the driver will have to answer in order to emulate the experience of texting while driving.

To monitor the state of the driver, we have used the Microsoft Kinect to perform skeletal tracking. Skeletal tracking will be able to differentiate when the driver is texting by detecting the skeletal configuration which maps to only having one hand on the wheel while the other hands is busy texting. The downside to this setup is the inability to measure whether the driver is paying attention on the road. We found that while the driver might be holding onto the phone and thus leaving only a hand on the steering wheel, this might be interpreted as a distracted state, however, the driver is still looking at the computer screen and paying attention on the road. A possible improvement to this would be to include possible head tracking, or even eye-tracking to measure the degree of focus.

In the controller application session, we used the driver model to generate sequences of predictions for future states of the vehicle given current observations of the vehicle, environment and driver skeletal configurations, and based on the controller's assessment of safety, apply the necessary amount of intervention in order to prevent lane deviations or collisions with obstacles.

The experimental setup and the driving scenarios on the CarSim is shown in Figure 3.12. On the left, we see the screenshot of the driving simulator and on the right we see the hand configuration when the subject is texting. The left picture also shows the real-time skeletal tracking overlaid on the subject.

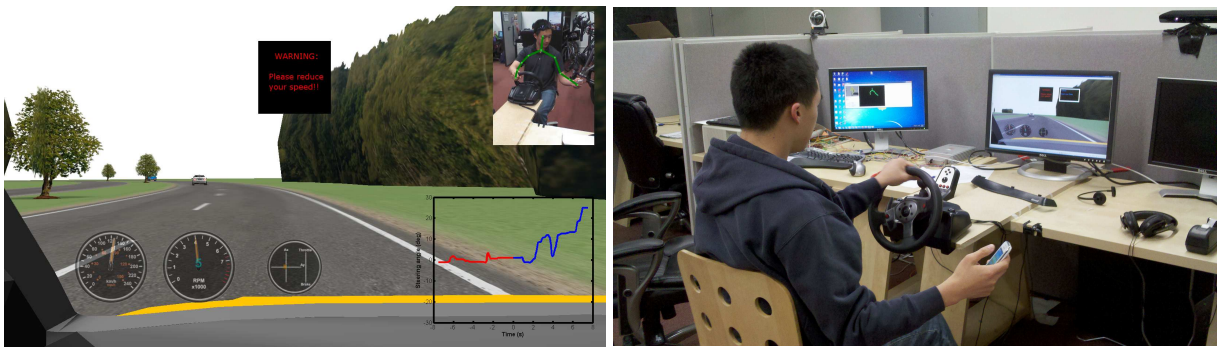


Figure 3.12: View from CarSim driving simulator during texting while driving experiment [19] (left) and experimental setup with control interface [19] (right).

Attention measurement during texting while driving on the highway: The second experiment involves real highway driving in a distracted state. Performed at the test tracks of the Hyundai CPG, we asked the driver to stay within the lane while texting on a smart phone. We have established from the previous experiment that skeletal tracking alone is not enough to fully determine the cognitive state of the driver in distracted scenarios. Therefore, we equipped the driver with the SMI eye-tracking glasses to measure and estimate when the driver is paying attention on the road, and when the driver is paying attention to texting on the phone. With this experimental setup, we are interested in modeling how driver allocate attention between the primary task (driving) and the secondary task (texting), based on the traffic conditions and vehicle states such as speed.

3.2 Simple dynamic task and game-based decision making

The previous section have focused on detailing the driving applications of human modeling. It is important to recall that in trying to understand and model different human decision making problems, two major motivations are: the possible utilization and integration of human models and behavior prediction into intelligent frameworks, and a better understanding and quantification of behavioral concepts. While the previous section focused on driving and the possible application of human models into autonomous systems, some attempts on understanding behavioral concepts were also made. However, the experimental setups and the uncertainty of traffic conditions and random stimuli makes it difficult to vary environmental conditions and repeat certain experiments with consistency. Therefore, to focus on specific principles affecting the decision making and to create an arbitrary environment for the subject, we scaled down the problem and investigated into simple dynamic tasks and virtual game-like experiments which tries to probe at similar psychological concepts associated with driving.

3.2.1 Juggling

The task of juggling is used as an example of a simple dynamic task that still encompasses a layer of complexity. As can be seen from the skill differences between a novice and a professional juggler, juggling is not an intuitive task and requires practice and learning of the hand-eye coordination [107]. Juggling can be viewed as a task which involves both throwing and catching at precise timings, and therefore the juggler needs to have good predictions of ball trajectories in order to sustain a juggling pattern. By examining the gaze trajectories in relation to the ball positions, we hope to determine whether expert juggler make trajectory forecasts of the ball.

We equipped a professional juggler with the eye-tracking glasses, as seen in Figure 3.13, to determine the gaze point of the juggler while ball is in flight. We recorded the gaze trajectories in relation to the ball (provided by the scene camera shown in the right of Figure 3.13 in two different scenarios. In the first scenario, we asked the juggler to perform a three-ball cascade using the same type of balls, and in the second scenario, we asked the juggler to also perform the same pattern, but this time using three objects of different weights, shapes and sizes: a ball pin, a light large ball, and a smaller heavier ball.

Figure 3.13: Expert juggler performing the 3 ball cascade with eye-tracking glasses (left) and screenshot from scene camera with gaze point (right).

3.2.2 Projectile game

The first game-like experiment of interest, which we term the *projectile game*, is similar to the popular computer and mobile-based game angry birds [108], where subjects are asked to aim at a target in a virtual environment where gravity affects the flight trajectory. Figure 3.14 shows screenshots of what the game looks like as presented to the subject. Subjects are initially shown with the instructions about how the game works and how the control inputs work. Then the subject plays through a training set to practice and experience the mechanisms of the game before being presented with the actual trials in which their actions are recorded and analyzed. Projectile games like this are created to mirror the concept of throwing a ball at a target in real life, which can be considered to be an essential element in the juggling task.

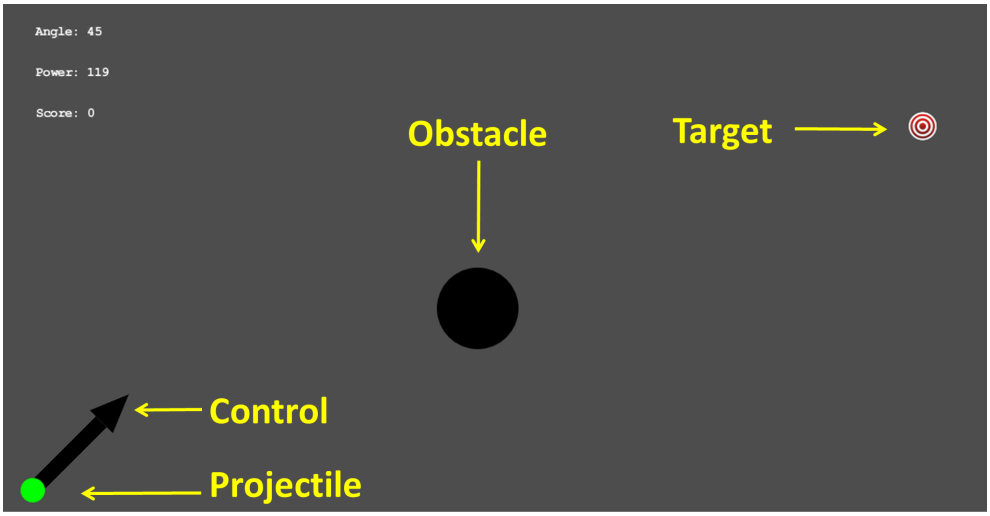


Figure 3.14: Labelled screenshot of projectile game showing the mechanism and configurations of the game.

Game setup and control: The game setup and control interface work in the following manner. Subjects are presented with a circular *target* (shown in red), and a circular *obstacle* (shown in black) of set radiuses. The subject's goal is to adjust the initial conditions of the projectile flight such that the flight trajectory of the particle (shown in green) intersects with the target and avoids intersecting with the obstacle at the same time. On the instructions page, subjects are informed that if they hit the obstacle, they will lose 2 points regardless of whether they also hit the target; will lose 1 point if they miss hitting the target; and will gain 2 points if they successfully hits the target and avoided the obstacle. The accumulated score is displayed in the top-left corner of the screen.

Subjects control the game using the keyboard. The up/down keys adjusts the power level (or initial speed) v of the projectile, and the left/right keys adjusts the initial angle α. Subjects receive feedback on the size of v and α through either the numerical value that is displayed in the top-left corner of the screen, and also visually indicated by the size and orientation of the arrow. Subjects are under no time pressure to decide on the best initial v and α to apply, and once they are satisfied with the current setting, hitting the space bar will release the particle in its flight.

At the beginning of a *trial set*, subjects are presented with a new configuration / position for the target and the obstacle which we refer to as a *scenario* or a *trial*. If the subjects fails at the current scenario, they will repeat the same scenario without any change to the target and obstacle configurations until they have successfully complete the current trial, and then presented with subsequent trials with different scenario settings. Each trial set consists of twenty different scenarios the subject will have to go through, and each trial set will be configured under specific settings i.e. gravitational constant, means of control interface, means of feedback etc.

Associated system model: Figure 3.15 illustrates the variables that are associated with the dynamics and computational analysis of the associated projectile game, which are listed and explained below for clarity and for ease of reference later in Chapter 4:

- $p_{\text{proj}} = (x_{\text{proj}}, y_{\text{proj}})$, x and y coordinate of projectile center point
- $p_{\text{targ}} = (x_{\text{targ}}, y_{\text{targ}})$, x and y coordinate of target center point
- $p_{\text{obst}} = (x_{\text{obst}}, y_{\text{obst}})$, x and y coordinate of obstacle center point
- v, initial velocity of projectile
- α, initial angle of projectile
- g, vertical gravitational acceleration constant
- R_{proj}, radius of projectile
- R_{targ}, radius of target
- R_{obst}, radius of obstacle

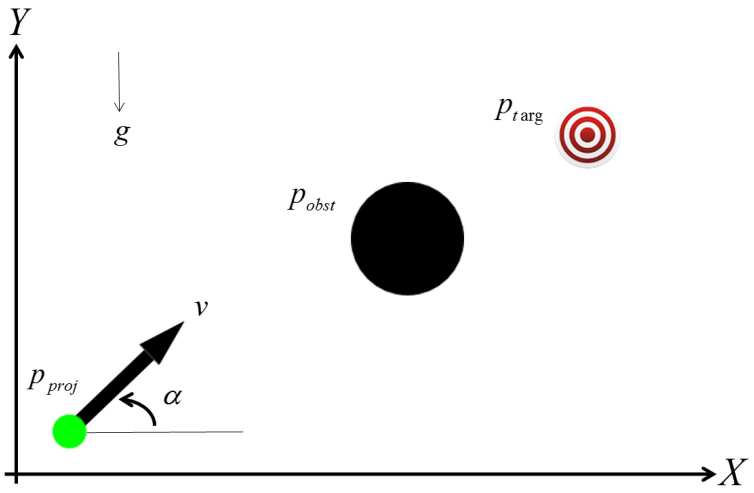


Figure 3.15: Coordinate system of projectile model.

Following simple kinematics with only vertical acceleration, the projectile trajectory can be subscribed over time t as a function of initial conditions.

$$x_{\mathrm{proj}}(t) = v_x t \tag{3.5a}$$
$$y_{\mathrm{proj}}(t) = v_y t + 0.5 g t^2 \tag{3.5b}$$
$$v_x = v\cos\alpha \tag{3.5c}$$
$$v_y = v\sin\alpha \tag{3.5d}$$

3.2.3 Dual-task game

In the *dual task* experiment, we are interested in emulating the experience of texting while driving, where the primary task is to pay attention on the road and keeping a safe distance and speed with respect to the traffic, and the secondary task is responding to text messages, and model the decision making process of selecting which task to pay attention to.

The subject is asked to play a game on the computer shown on two screens, each screen has one window in view. The first window shown in Figure 3.16, will display the primary task (Task 1), and the second window shown in Figure 3.17 displays the secondary task (Task 2). The setup is designed so that when the subject shift his/her attention between windows, his field of view will be constrained to focusing on a single window only.

The primary task consists of obstacles (small blue dots) that fall down from the top at a constant speed. The participant has control over toggling the target (large blue dot at the bottom) between staying in the left or the right column. The participant is told his goal is to avoid obstacles by shifting the position of the target, and if he/she hit an obstacle, he will incur J_1 points in penalty. In the same window as the primary task, there is a display box which issues instructions to the participant. Messages that are displayed include: “Press space to continue”, “Text completed”, or

"Check your text". The participant is told whenever "Check your text" appears, he should consider switching his attention to the secondary task.

In the secondary task, the participant is prompted to read a string of random numbers and enter the same sequence of numbers using the number pad on the keyboard. The secondary task is considered complete if the numbers match. He/she is told that whenever the secondary task is cued by the message, he will incur J_2T in penalty, T being the amount of time taken to complete the secondary task.

Intuitively, being given these instructions, the participant should aim to avoid all obstacles, at the same time complete the secondary task as quickly as he can without hitting the obstacles. Therefore, he needs to have some sense of predictions of the state of the primary task when he is attending the secondary.

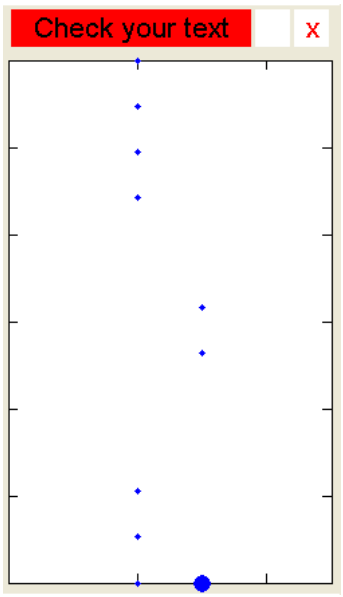


Figure 3.16: Primary task of obstacle avoidance in window 1 of dual-task game.

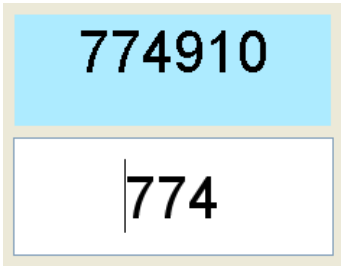


Figure 3.17: Secondary task of texting in window 2 of dual-task game.

3.3 Conclusion

In this chapter we detailed the various experiments performed to support our study of human decision making. The experiments included everyday tasks in the real-world such as driving on the highway and also when under distractions such as texting while driving. Data collected from these experiments will facilitate in identifying a suitable driver model for the purpose of integrating driver predictions into a semi-autonomous systems for guaranteeing future safety of the driver. We also performed experiments which involved scenarios where expertise is essential, such as driving in extreme conditions and three ball juggling. With data collected from these scenarios, we would like to see that experts do intrinsically learn the knowledge of the more complex system and use it to make future forecasts. Through the use of eye-tracking glasses, we can better understand the cognitive states of the human in various contexts. Lastly, to further explore the fundamental principles behind human decision making, we designed scaled down game-based experiments which parallel the contexts in everyday tasks, and attempt to bridge the different modeling abstractions between the data-driven approached for controller applications with the parametric approach for psychology studies.

Chapter 4

Modeling, Identification and Results of Human Behavior Applications

Using the data collected from the experimental applications described in Chapter 3 and the modeling methodologies introduced in Chapter 2, this chapter aims to present the first major contribution of this thesis. We will use the proposed computational modeling frameworks to better understand the mechanism and factors which affect human decision making in different problem contexts, both from the controls design and psychology perspective. With applications that span the two extremes, we will attempt to bridge the disconnection between real-world experiments and simplified scenarios in psychology, and draw parallels between the two.

Instead of an application-based approach, where the ordering of discussions are categorized by the experiment, we feel it is more suitable to center our discussions on the fundamental principles behind the various models and support them with application examples. Following the order in Section 2, we will first address modeling methods which involve directly looking up stored policy functions without the extra computational load associated with finding an optimal solution in real-time. The modeling principles behind linear feedback, PWA and SVM will be used to explain the control strategies in both driving and the projectile game, with the emphasis that these methodologies of lesser computational effort support the reflexive behavioral patterns observed from the subjects. The second part of this chapter will focus on optimization-based models, which are policies that cannot be applied directly but instead need extra computational time to solve for an internal optimization problem. The optimal framework extending to constrained optimization, MPC and MDP will be used to simulate the predictive and cost/reward based strategies seen in complex driving maneuvers and distracted driving, as well as projectile and dual-task gaming applications. Lastly, we will use eye-tracking information obtained from driving and juggling to support the claim that subjects perform state prediction through forward models in the optimization-based models.

4.1 Direct method: function map of stored policies

The first approach to modeling decision making is the direct method where the policy maps are stored, and making a decision corresponds to looking up a learned policy given the observations. To make a connection with the bottom-up approach involving the central nervous system, the learned policies and the process of learning can be paralleled to the adjustment of the weights in the neural network [73]. Direct methods typically involve minimal computational effort, unlike the optimization-based policies which will be discussed in the second part of this chapter. Following this characteristic, direct methods naturally become suitable candidates for modeling behaviors which appear to be more repetitive, reflexive, immediate and associated with lower-level decision making problems, as opposed to higher-level decision making which involves more planning and mental simulations [109]. The lesser computational effort of direct methods can be paralleled to cognitive processes which can free up the processing bottleneck allowing subjects to multi-task [65].

Starting with the simplest method of linear feedback, we draw some parallels between highway lane keeping and adjusting control inputs in a projectile game. However, simple linear feedback might not be enough to capture the nonlinearity and uncertainty associated with the projectile game, and possibly a switched system is required to deal with different subgroups of data. A similar argument is applied to extreme driving maneuvers involving the nonlinearity of tire dynamics, where a switched system is used to capture the steering control. Lastly, the discrete decisions of choosing the mode of control in the projectile game can be used to parallel the discrete decisions made in lane switching applications.

4.1.1 Linear feedback model for highway driving and lane keeping

To start off our discussion of using direct methods to model lower-level decision making problems, we will cite from [110] and outline briefly the driver model that is used to model lane keeping on the highway. The lane keeping component of highway driving can be viewed as a lower-level control problem that requires less planning, i.e. it is a well-practiced and repetitive task with which a typical driver will have no problem performing other tasks on top of. Often times, drivers will have to navigate and perform path planning in heavy traffic, the task of staying inside the lane boundaries becomes an automatic process which does not require a lot of attention.

The driver sees deviations from the center of the lane or upcoming curvatures and will try to steer just the right amount to realign the vehicle properly with the lane. The look ahead point and feedback gain will be a matter of experience, preference, vehicle steering wheel sensitivity, and current road condition such as road surface. Referring back to Figure 3.3 and the vehicle state descriptions in Section 3.1.1, the model is formally defined as,

$$\delta = K_y e_y + K_\psi e_\psi \tag{4.1}$$

where K_y and K_ψ are the linear feedback gains. The continuous input vector into the model map is $x = [e_y, e_\psi]$, which are the deviations from the lane center and the road curvature at a distance from the current position of the vehicle. We call the point where this distance is measured from

the preview point, and this is a parameter that is subject dependent. The continuous output vector from the model is $y = \delta$ is the steering angle.

4.1.2 Linear feedback model for the projectile game

The modeling for the projectile game experiment can be treated as a hybrid of two different models, with one capturing more of the planning and predicting aspects of the decision making process, and the other, similar to the repetitive, low-level philosophy of the previously introduced steering control for lane keeping, directly mapping the feedbacks from the previous trial to the input decisions of the current trial. We will first present the direct mapping approach and its relation to the subject's observed behavior and discuss the optimization-based predictive models later in the chapter.

As presented in Section 4.1, the linear feedback model is a direct map between the feature input vector r and output vector u, and the model structure makes parameter identification simple and data-driven. Therefore, in modeling the projectile experiment with the this approach, during the model identification stage we will include as many variables as possible in the feature vector, even if some features doesn't intuitively correlate to the decision making, or might be slightly repetitive. After including a large feature vector, we apply the LASSO identification technique to minimize the number of features. This will provide us with insight as to which features are important to the subject when making the next decision.

Definition of input / feature vector r

Based on the context of the problem, the candidate features are classified into three categories: *scenario*-based, *input*-based and *feedback*-based. Scenario-based features include p_{targ} and p_{obst}, which are positions of the target and obstacles. Input-based features include v and α, which are the initial conditions of the previous trial. Lastly, feedback-based features are $f(p_{\text{proj}}, p_{\text{targ}}, p_{\text{obst}})$, which are functions of the visual feedback the subject receives from the previous trial consisting of the projectile trajectory in relation to the target and obstacle positions.

The feedback-based features are the only ones which models how the subject adjusts the next input based on the most current feedback it receives. This models the short term nature of mapping current deviations to the control input. On the other hand, the scenario-based features are fixed for the current trial and the variables don't change based on the subject's last input.

Feedback-based features: There are several factors to be considered when determining the function which defines the feedback based features. Firstly, we are interested in finding a relation which captures the error between the trajectory to the target and obstacles. Since p_{proj} is parameterized by time t in Equation (3.5), there are multiple alternatives as to the value t to use in the feature vector. One option is to find the point $t_{\min,*}$, which gives the minimum distance to the respective points of interest p_*, formally defined as,

$$t_{\min,*} = \arg\min_t \|p_{\text{proj}}(t) - p_*\| \tag{4.2}$$

where the star symbol $*$ is a placeholder for either the target or the obstacle. Alternatively, the y position where the trajectory intersects with either the x coordinate of the point of interest can also be used.

$$t_{y,*} = \{t : y_{\text{proj}}(t) = y_*\} \tag{4.3}$$

Lastly, the highest point in the trajectory flight at t_{apex} is also used to extract the x and y position for the feature vector,

$$t_{\text{apex}} = \{t : \frac{dy}{dt} = 0\} \tag{4.4}$$

In post experiment surveys, this point in flight was reported by the subjects to provide valuable information about the nonlinear map between the control input and the flight path. In our own study of juggling and in [111, 112], which involves a similar trajectory mechanic, it was also observed that subjects focused their attention on the apex of the balls to provide information for optimal state estimations.

Definition of output / control vector u

With the input features vectors explained in detail, the output vector of the linear map are

- Δv, change in initial velocity from last trial
- $\Delta\alpha$, change in initial angle from last trial

We have chosen to model the control inputs as the amount of *change* to add to the previous input since this aligns with the way the control interface in the experiment is designed, which is from the incremental adjustments of the keyboard without resetting the values after every trial.

Model formulation and identification

Following the linear feedback framework introduced in Section 2.2.1, and the defined input r and output u vectors, the model can be formally presented as follows.

$$u_k = \theta \begin{bmatrix} r_k \\ 1 \end{bmatrix} \tag{4.5}$$

where the input feature vector r_k is defined as,

$$r_k = \begin{bmatrix} x_{k,\text{targ}} \\ y_{k,\text{targ}} \\ x_{k,\text{obst}} \\ y_{k,\text{obst}} \\ v_{k-1} \\ \alpha_{k-1} \\ \Delta v_{k-1} \\ \Delta\alpha_{k-1} \\ x_{k-1,\text{proj}}(t_{\text{min,targ}}) \\ y_{k-1,\text{proj}}(t_{\text{min,targ}}) \\ x_{k-1,\text{proj}}(t_{\text{min,obst}}) \\ y_{k-1,\text{proj}}(t_{\text{min,obst}}) \\ y_{k-1,\text{proj}}(t_{x,\text{targ}}) \\ x_{k-1,\text{proj}}(t_{\text{apex}}) \\ y_{k-1,\text{proj}}(t_{\text{apex}}) \end{bmatrix} \tag{4.6}$$

and the output control vector u_k is defined as,

$$u_k = \begin{bmatrix} \Delta v_k \\ \Delta\alpha_k \end{bmatrix} \tag{4.7}$$

We use LASSO identification [113] to find the optimal parameter θ which minimizes the error between the recorded output to the predicted output from the model for a given value of λ.

$$\min_{\theta} \|u_k - u_{k,\text{s}}\| + \lambda\|\theta\|_1 \tag{4.8}$$

A larger value of λ will favor the algorithm to find a more sparse θ with more zero entries, and therefore, towards eliminating redundant features.

Identification Results and Analysis

In this section, we present the model identification results of the formulation discussed before. Feature vectors r_k and model outputs u_k are extracted from recorded data of the projectile experiment described in Section 3.2.2. LASSO is used to identify the linear feedback gain θ which minimizes prediction error. The results will be discussed in the following order. First we present the prediction output where the entire data set is used in the identification without parsing. Then we describe how the data set is parsed into different groups, and present the results where identification is performed separately on individual groups. For selected groups, we discuss in detail the specifics related to the LASSO algorithm and how it shows redundancy in certain features. We will continually to make connections between the results from the data and the behavioral patterns. Finally, we conclude the discussion that a simple linear feedback model can only capture specific scenarios and is insufficient to model the entire experiment, which leads to using the optimization-based models later in the section.

Results of all recorded trials: Shown in Figure 4.1 are plots comparing the error between the predicted initial conditions of the model and the actual initial conditions that was applied by the subject. We can see that the model makes some bad predictions and the error between the predicted and the actual for $\Delta\alpha$ is approximately between -0.2 to 0.2 radians (rad), and for Δv between -20 to 20 pixels per second (pix/s). For $\lambda_\alpha = 0.00079$, and a cross-validation partition of 10, the mean squared error (MSE) for the prediction of $\Delta\alpha$ is 0.0043. (For the sake of simplicity, all the LASSO fits were performed with a cross-validation of 10, therefore, we will omit this detail later on when making references to the identification results.) For $\lambda_v = 0.11$, the MSE for the prediction of Δv is 50.6 which is considerably high relative to the admissible range of $v \in [0, 200]$. So even though the initial angle predictions doesn't deviate as much, the velocity predictions deviate by a significant amount and will correspond to a vast difference in the resulting trajectory.

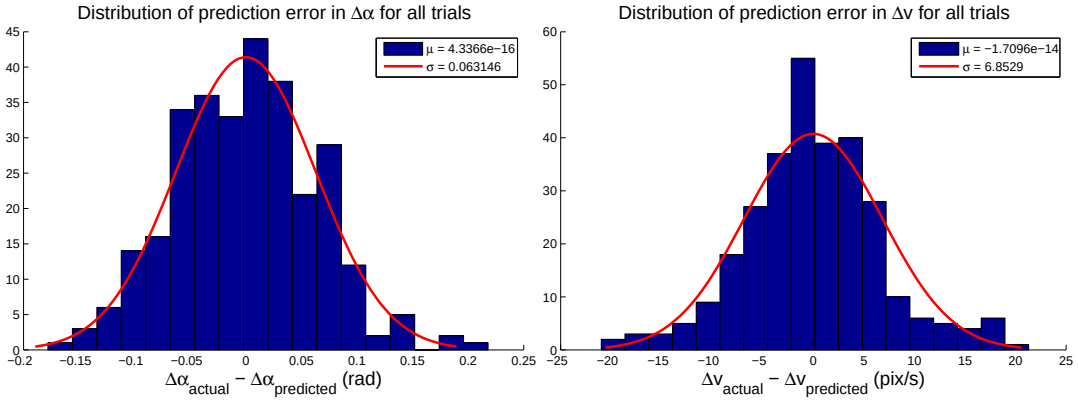


Figure 4.1: Distribution of prediction errors for $\Delta\alpha$ and Δv for all trials.

To illustrate the difference in the trajectory path due to the discrepancies between the predicted and the actual, Figure 4.2 shows two of example screenshots of how the predicted initial conditions map out to projectile trajectories and how it differs with the subject's trajectories. The dashed blue line shows the feedback of the projectile trajectory from the previous trial, which is used to extract features for the regressor. The solid blue line shows the actual projectile trajectory that resulted from the subject's selection of initial conditions. Finally, the solid red line shows the projectile trajectory that resulted from the initial conditions predicted from the identified model. We can see from the left plot of Figure 4.2 the actual input missed the target but the prediction actually hits the target. Similarly, the right plot shows the actual input was successful in hitting the target and the prediction hitting the obstacle and almost missing the target.

So far we have seen that LASSO has failed to identify a good linear model when the entire data set is used. Unlike the lane keeping application on the highway, this shows that a simple linear model is insufficient in capturing the complexity of all the decision making that is involved with the projectile game. To tackle this problem, we will now try to break down the data into specific scenarios and attempt to fit a linear model for each individual scenario.

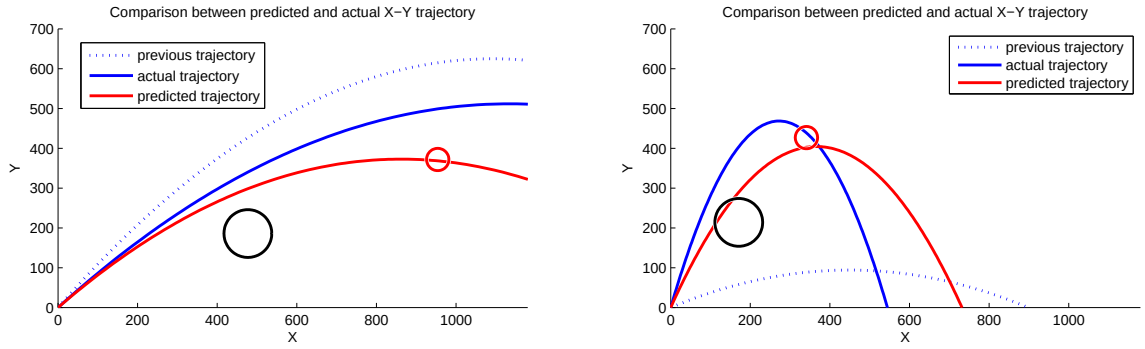


Figure 4.2: Comparison between predicted and actual X-Y trajectory.

Parsing the trials into different characteristic groups: The most obvious method of parsing the data set is to group them into either *old* or *new* scenarios. Old scenarios are defined as scenarios with the same target/obstacle configurations as the previous one, which the subject has to make a second or subsequent tries at due to failure at the previous attempt i.e. missing the target or hitting the obstacle. New scenarios refer to new target/obstacle configurations which the subject is presented with after a successful attempt. Psychologically, with an old scenario, the subject has better memory of the relation between the scenario configuration and the trajectory feedback from the previous trial, and will most likely react differently or adopt a different decision making strategy when presented with a new scenario. In addition, with new scenarios that are positioned significantly different from the previous, the trajectory feedback might not be as valuable or consistent to be used in the feature vector of the linear map.

The second approach of grouping the data set is to separate the initial conditions which resulted in a successful trial. For a successful trial, subjects are more consistent with their decision making strategies, perhaps due to past experiences and has learned the best strategy over time. Unsuccessful trials on the other hand, results from inconsistent strategies or unfamiliar configurations which the subject has not yet learned to solve. If the subject fails at an attempt, then he/she is will likely be updating the model, in this case the feedback gains, until the trial is successful. Intuitively it makes more sense for a subject to be consistently successful, rather than consistently failing. Therefore the feedback gains for successful attempts will be more constant, whereas for failed attempts, it is updated and constantly fluctuating. This is similar to the situation in driving where the road conditions are constantly changing i.e. due to weather conditions. If the driver is used to driving on dry asphalt surfaces and suddenly come across a wet or icy area, the friction coefficient will drop substantially, and therefore the linear gain in (4.1) will be adapted accordingly.

The third criterion for partitioning the data is to use the mode of control input, which can be separated into three groups: $\{\Delta\alpha = 0, \Delta v \neq 0\}$, $\{\Delta\alpha \neq 0, \Delta v = 0\}$, and $\{\Delta\alpha \neq 0, \Delta v \neq 0\}$. We feel certain configuration and feedbacks will result in the subject choosing one strategy over the other, and the uncertainty and mind set associated with whether to manipulate one or two control inputs

Group	Size	Old Scenario	New Scenario	Success Trial	Failed Trial	$\Delta\alpha \neq 0$ $\Delta v = 0$	$\Delta\alpha = 0$ $\Delta v \neq 0$	$\Delta\alpha \neq 0$ $\Delta v \neq 0$
1	100%	✓	✓	✓	✓	✓	✓	✓
2	47%	✓		✓	✓	✓	✓	✓
3	53%		✓	✓	✓	✓	✓	✓
4	53%	✓	✓		✓	✓	✓	✓
5	47%	✓	✓	✓		✓	✓	✓
6	3%	✓	✓	✓	✓	✓		
7	22%	✓	✓	✓	✓		✓	
8	75%	✓	✓	✓	✓			✓
9	36%	✓		✓		✓	✓	✓
10	22%	✓		✓	✓		✓	
11	22%	✓		✓	✓			✓

Table 4.1: Characteristic of partitioned data from the projectile game.

will be different.

Upon separating the data set into different groups, the LASSO formulation is again performed on each of the individual groupings. Table 4.1 shows a description of how the groups are partitioned, labeling each group from Groups 1-11, in addition the respective percentage of the data set in each of the groups is included. Figure 4.3 shows the normalized MSE (NMSE) for the predictions of all the sub-groups listed in Table 4.1. NMSE takes the largest MSE value for $\Delta\alpha$ and Δv in Groups 1-11 as the normalizing constant. Note that Group 1 from Table 4.1 is the original data set without parsing used for the analysis in the previous section.

As discussed previously, Groups 3 and 5, which are the new trials and the failed trials have the greatest NMSE. New trials have larger uncertainty associated and in failed trials, the linear gains θ are being adapted in real-time. We can see from the NMSE of Groups 4 and 7 that the subject is more consistent in adjusting Δv than $\Delta\alpha$. The NMSE, significantly lower than for Group 1 (original data set), indicates that the subject has learned and converged to a more constant θ_v associated with the velocity input more than the angle. In fact, the extremely low percentage of $\Delta\alpha$ only trials compared to the higher percentage of Δv only trials in Table 4.1 shows that subjects are more comfortable and confident in using Δv to hit the target.

Relating results of LASSO to behavioral features: We will now analyze the results of LASSO, in particular the search and interpretation for the sparsity of θ in relation to the feature vector. We feel it will be too redundant to repeat the same analysis for all 11 partitioned data set groups, as it is not our goal to interpret the projectile game in detail but to abstract the fundamental idea of using the LASSO technique to facilitate in making connections to human behavior in general. Therefore, to keep the analysis simple, we will use the results of Group 2 and 3, which involves either old or new scenarios, as an example.

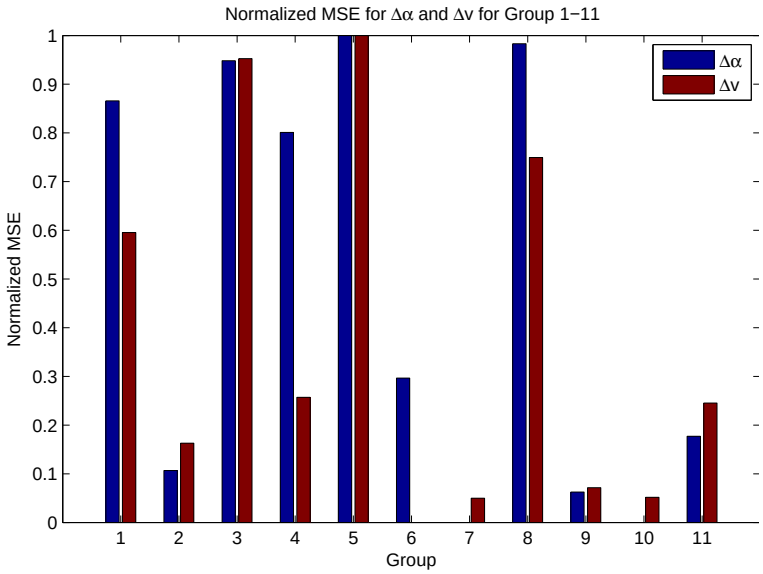


Figure 4.3: Normalized MSE for $\Delta\alpha$ and Δv for Group 1-11.

Shown in Figure 4.4 and 4.5 is a comparison between the MSE for different λ values. Larger λ results in larger MSE since more weight is placed on increasing the number of zero entries in θ and thus making the parameter less flexible. In addition, Figure 4.6 and 4.7 shows the respective values of θ for different λ. Comparing between which parameter values affect the control inputs, we can see that for Group 2, which is the group with old scenarios, the dominant features which affect both $\Delta\alpha$ and Δv are the 9th, 10th, 12th, 13th and 15th entries of θ. Referring to (4.6), these correspond to $x_{k-1,\text{proj}}(t_{\text{min,targ}})$, $y_{k-1,\text{proj}}(t_{\text{min,targ}})$, $y_{k-1,\text{proj}}(t_{\text{min,obst}})$, $y_{k-1,\text{proj}}(t_{x,\text{targ}})$, and $y_{k-1,\text{proj}}(t_{\text{apex}})$ respectively. Interestingly, these are all associated with the feedback-based features. In comparison, for Group 3, which is the group with new scenarios, the parameter values are now dominated by features 1-5. These correspond to the coordinates of the target and the obstacle and the input v_{k-1}, which means now the scenario-based features dominate the decision making.

Concluding remarks

We have shown that the repetitive nature associated with the projectile game in certain trials can be considered to be of lower-level control with minimal computational effort, and therefore the direct feedback mapping method can be used to model the decision making policy of the subjects, in the same framework as the steering control on the highway. Partitioning of data sets into different groups according to several characteristics illustrates the adaptive nature of the linear gains, as well as convergence to a consistent gain for the velocity input option. In addition, the nature of the trials, whether it is a new or old trial, demonstrates the limitations of using a single feedback gain to model the entire task. Old trials provide subjects with better error estimates from the feedback trajectory of the previous trial, and therefore leads to a more consistent linear feedback model. New trials, on the other hand, have error estimates that are more uncertain due to memory loss,

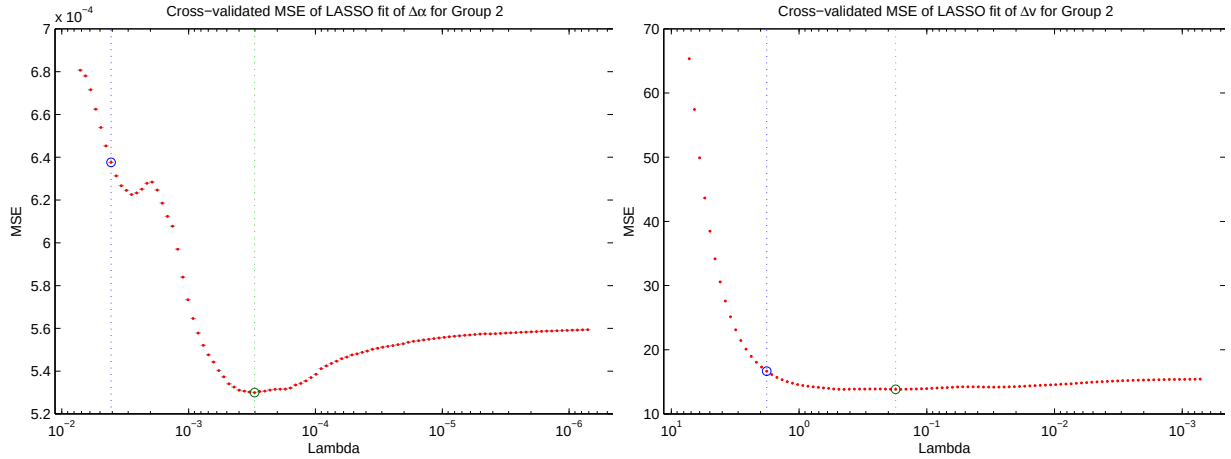


Figure 4.4: Cross-validated MSE of LASSO fit for Group 2.

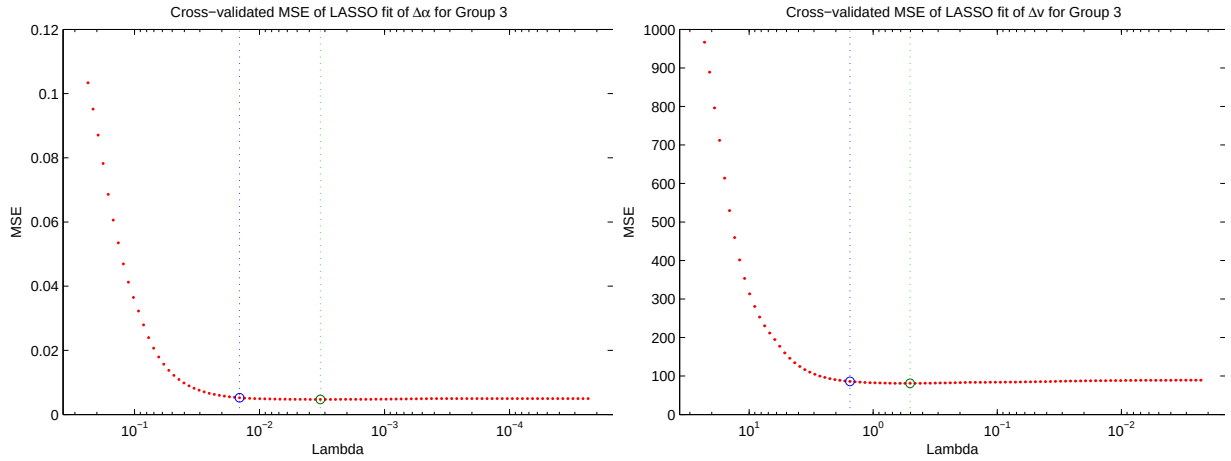


Figure 4.5: Cross-validated MSE of LASSO fit for Group 3.

and therefore, relied more on the configuration of the scenario. This leads to the idea that a more complex model is required to capture the associated nonlinearity and uncertainty.

One approach is to use the optimization-based method to deal with the planning of trajectories in the uncertainty environment. We will discuss this later in the chapter. Locally weighted regression as applied to robotic juggling in [114], can also be used to find locally linear models. Similarly, PWA models introduced in Section 2.2.2, which is also a switched system with segmented linear maps can be used to model the associated nonlinear behavior. We will give an example using this PWA approach in the next section, discussing extreme driving maneuver which involves the nonlinearity of the tire dynamics.

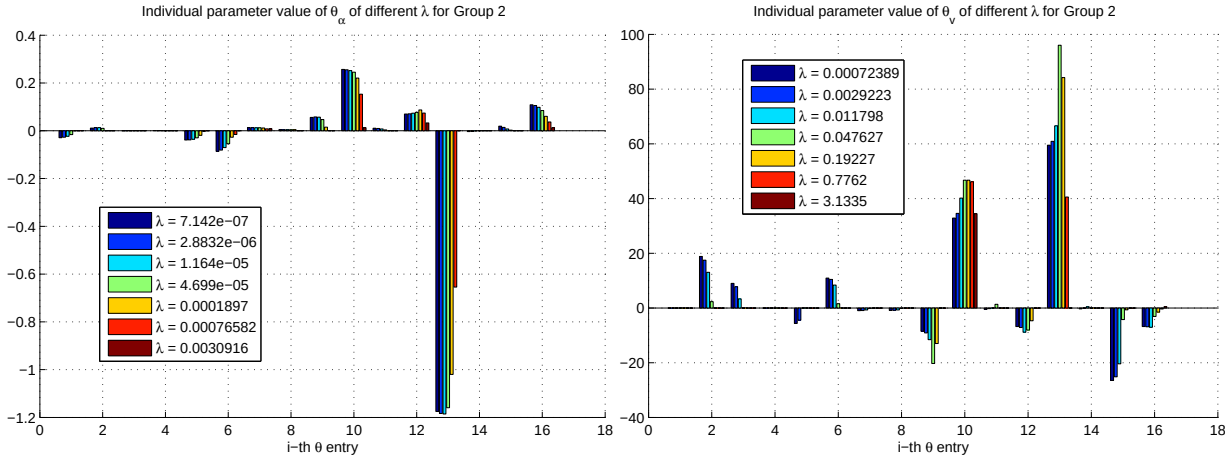


Figure 4.6: Individual parameter value of θ_α of different λ for Group 2.

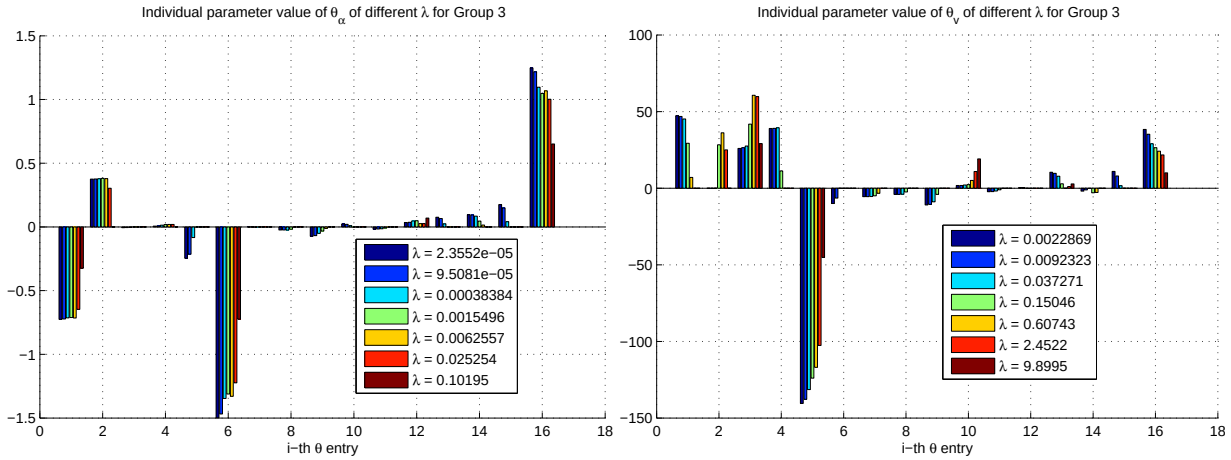


Figure 4.7: Individual parameter value of θ_α of different λ for Group 3.

4.1.3 Piecewise affine model for extreme driving maneuvers

Piecewise affine (PWA) models is another example of the direct method. Similar to the linear feedback method which simulates the direct mapping between input and output with small computational effort. The PWA model is a particular example of switched systems which extends the complexity of the policy map to include multiple sets of gain values depending on the context of the problem. For example, with the partitioning of data sets in Section 4.1.2, PWA can be used to approximate the nonlinear map with smaller segments of linear maps. As mentioned previously, gain scheduling is the simplest form of PWA models where the gain values are different for different segmentations of the state variable. In situations such as lane keeping, drivers can be modeled as a simple linear controller with a state feedback law [33]. As for the extreme maneuver introduced in Section 3.1.6, which encompasses the full nonlinearity of tire dynamics, modeling the human driver as a simple linear controller will not suffice. In this section, we will model steering

behavior in extreme conditions using PWA models.

Switched systems involves switching between different modes of operation. Different modes might have different system models and/or control laws. This intuitively lines up with human decision making. Under different conditions, humans have corresponding control strategies. Following common intuitions, we propose to decompose the drifting maneuver into two control strategies:

1. The maneuver starts off with the vehicle traveling in a straight line. In order to *drift*, the controller is responsible for "destabilizing" the vehicle by saturating the tires, and placing the vehicle at a high lateral velocity and a large yaw rate.

2. Once the vehicle is destabilized, the controller needs to stabilize the vehicle again, more importantly, the vehicle has to end in the desired final state, i.e. heading angle of 180 degrees.

The conjecture is that if the human driver uses a switched control strategy in practice, we should be able to observe a similar behavior by modeling drifting by a human driver as a two-mode switched system. Therefore, instead of having a highly nonlinear and complicated mental model of the world, complex models can be broken down into different modes with simpler dynamics within each mode. Following the PWA modeling framework described in Section 2.2.2, we will now define the associated variables for the steering behavior in extreme conditions that are used to identify the parameters of the PWA model.

Model formulation and identification

To identify the model of the human driver, input and output information are recorded during each trial and the parameter θ_i is identified for each mode. In this paper, we implemented the cluster-based procedure [80]. For comparison of this algorithm with other hybrid identification techniques, readers can refer to [79].

For the drifting scenario, we are interested in making predictions of the driver's steering based on state information. The hybrid model's output vector is the steering angle,

$$y_k = [\delta_k] \tag{4.9}$$

and the input vector is defined from the vehicle states,

$$u_k = [\psi_k, \dot{\psi}_k, Y_k, \alpha_{f_k}, \alpha_{r_k}]^T \tag{4.10}$$

Identification Results and Analysis

Using the PWARX formulation, we identify the parameters θ_i, predict the steering profile $\hat{y}_k = \theta^T_{\sigma(k)}[\hat{y}^T_{k-1}, u^T_k, 1]^T$, compare $\hat{y}_k$ with the actual driver steering y_k and analyze the effectiveness of introducing additional modes in the hybrid system. Three sets of models differing in the number of modes ($s = 1,2,3$) are identified. The maximum prediction error, $e = \|y_k - \hat{y}_k\|_\infty$, for each model is denoted respectively as e_1, e_2 and e_3 and presented in Figure 4.8.

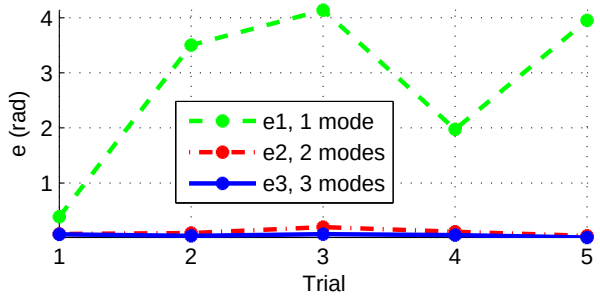


Figure 4.8: Maximum prediction error for 1, 2 and 3 mode PWARX models.

Determining the necessary number of modes: The one-mode system relates to a single linear controller. For the one-mode hybrid system, the prediction error is comparatively higher. This confirms our previously discussed conjecture that in order to drift, the driver has to understand the nonlinearity involved, and therefore a simple linear control strategy will not suffice. Notice that e_1 in trial 1 is smaller than subsequent trials. State trajectories for trial 1 shows that the driver failed to achieve the maneuver. This indicates that in trial 1, the driver is likely to be still using a single-mode strategy and only learns to switch between modes through subsequent trials.

Comparing e_2 and e_3, it can be seen that, only a very small improvement is obtained when an additional mode is added. This confirms our conjecture that it is sufficient to decompose the drifting maneuver into two modes of control destabilizing to achieve high slip angles, and stabilizing to achieve final desired state. We will focus on the two-mode model from here forth.

Observed input of vehicle states: Figure 4.9 shows the trajectories of the states which were used as features to the model input in (4.10). The blue and green lines shows the two modes that are identified. We can see that when the side slip angles α are high, the mode switching occurs, and once switching happens, the vehicle is stabilized with a much smaller value of α.

Comparison between actual and predicted steering output: Figure 4.10 compares the actual steering angle with the predicted steering angle from the two-mode hybrid model. Notice that the state evolution began in mode 1, switched to mode 2 at $t = 4.8s$, and stayed in mode 2 without any further switching. This confirms our hypothesis of the decomposition of the control strategy in the drifting maneuver.

θ_1, the parameter of mode 1 of a two-mode model, obtained from model fitting of trial 5 data is validated on two other trials. Keeping θ constant means we are not allowing the predicted driver to switch his strategy. Figure 4.11 shows the comparison between the recorded steering profiles and that predicted from θ_1. We can see that the first half of the steering profile makes very good predictions but the second half, where the driver should be switching modes, fits poorly. The results of our model validation conforms to the hypothesis of a two-mode switching control strategy.

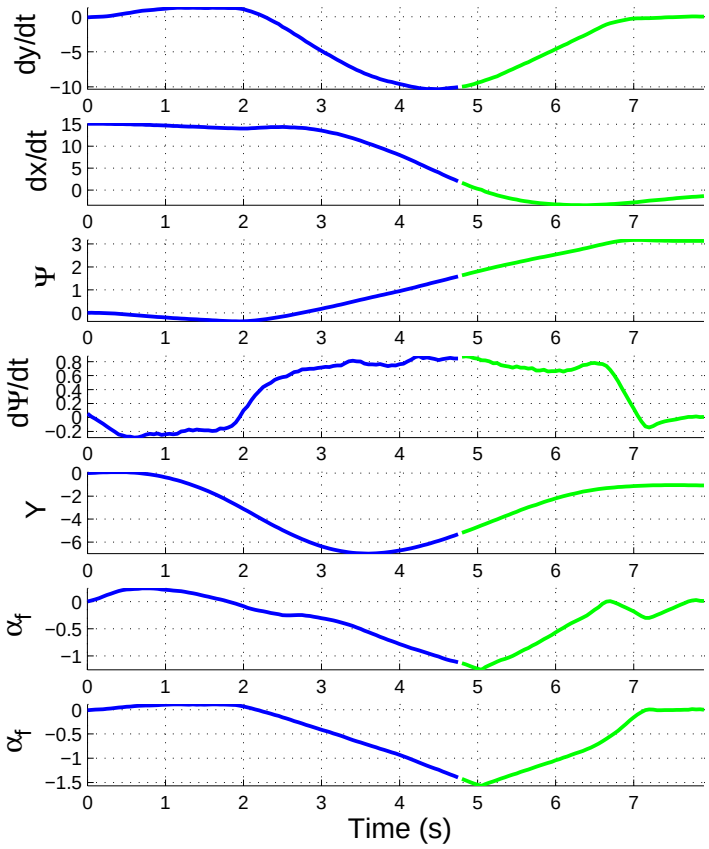


Figure 4.9: Vehicle states during extreme drift maneuver.

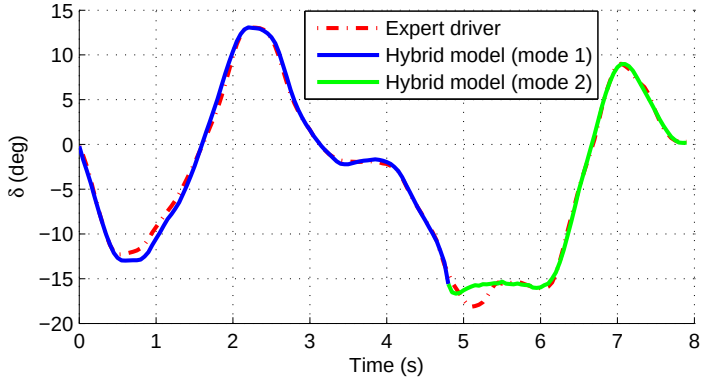


Figure 4.10: Predicted vs actual expert driver steering angle for drift maneuver.

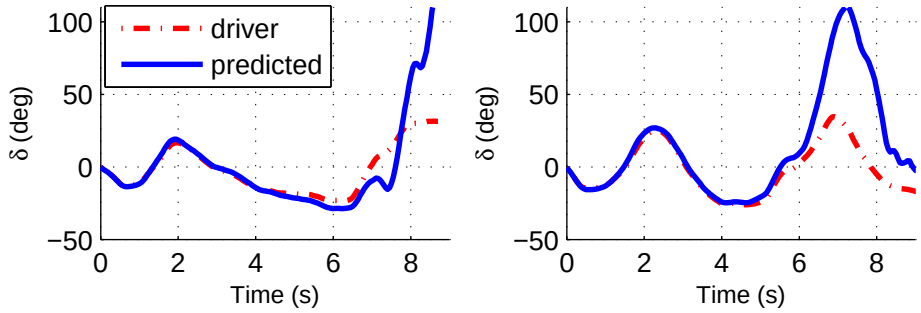


Figure 4.11: Model validation: θ_1 from trial 5 validated on data of trial 2 and 3.

Concluding remarks

In this section, we have shown that the extreme driving maneuver involving the saturation region of the tire can be accounted for if we use a two-mode switched system. We showed that the one-mode, which is essentially a simple linear feedback like the model for highway driving is not sufficient to capture the nonlinearity involved with the tire dynamics, and therefore a switched system is necessary. A similar strategy can potentially be used to capture the different modes for the projectile game. Our intuition of how a driver handles extreme driving with destabilization to the unstable regions, followed by the stabilization of the vehicle states aligns with the switching conditions we saw in the vehicle trajectories.

One could argue that the dynamics and the type of extreme maneuver discussed here is too complex for the driver to be acting reflexively and not planning based, such that direct methods are overly simple to be used for modeling the decision making a driver uses in extreme situations. Alternatively, an optimization-based approach can be used to capture the planning and predictions the driver might be simulating. Before we investigate this further later in the chapter, we would like to first discuss how direct methods can also be used to model discrete decisions, such as the decision to switch lanes on the highway.

4.1.4 SVM model for the discrete lane change decision

As mentioned in Section 2.2.3, support vector machines (SVM) are well-established methodologies that are used in many machine learning applications involving two-class or even multi-class classifications. Recall in the projectile game, when the subject is making the decision on how to manipulate the control input, he/she will first have to decide on which control input to manipulate, and the associated linear strategies that follow this decision will be different. The decision to adjust only the angle, only the velocity, or adjust both simultaneously can be viewed as a three-class decision making problem.

Similarly, for the lane-changing on the highway scenario, if we label the training data at any instance as either *lane keeping* (LK) or *lane changing* (LC) mode, then this will fit as a two-class or two-mode classification problem. Intuitively, when a driver is driving on the highway, he/she will keep the vehicle at a comfortable speed and will try to keep the vehicle within lane boundaries,

in other words, executing the lane keeping mode. Under certain conditions, the driver will turn on the turn signal and decide to switch lanes. For example, it could be destination related where the driver is either following highway signs or trying to get off/on the highway, or it could be state / environment related where the front vehicle is traveling too slow. In this section, we are interested in using highway driving data and SVM methods to model and predict the driver's intent to switch lanes.

Model formulation and identification

It is important to first establish how the training data is labelled. Since we have no clear methods to actually measure and define the driver's *intent* to switch lanes, this needs to be defined explicitly and indirectly from measurable states of the vehicle. One method is to use the turn signal as an indication of when the driver wishes to switch lanes. For example, as soon as the turn signal is switched on, then a window of width w_1 is taken about this instant for the states of the vehicle and the environment, and these data are labeled as corresponding to the lane changing mode, and all the data when the turn signal was off is labeled as the lane keeping mode. However, one potential problem with this approach to labeling could come from different people's habit of driving and when they use the turn signal, or maybe some people don't use it at all. In addition, putting on the turn signal doesn't necessarily indicate that the driver feels it is *safe* to switch lanes. These are two very different mind sets. The first is the intent to switch lanes and correlates more with the states and the traffic flow of the proceeding vehicles. The second is the safety and comfort level of switching lanes and correlates more with the states of the trailing and neighboring vehicles. To model when the driver feels safe to switch lanes, then we need to take a window of width w_2 about the instant when the driver starts to intentionally steer out of the lane, and label these data sets as lane changing mode.

To formally define the variables used in the training data, each data point consists of a label $y \in \{\text{LK}, \text{LC}\}$ and a feature vector $x \in R^n$ consisting of $\dot{x}, \dot{x}_r, x_r$ which, as defined in Section 3.1.1, are the speed of the ego vehicle, the relative speed and distance between the ego and the proceeding vehicle respectively. Note that the observations available from the recorded data are only associated with the ego vehicle and the states related with the immediate proceeding vehicle. Unfortunately the driving experiment was recorded on a test vehicle which only have front facing radars and side or rear radars were not available. Therefore, although extremely important in the context of lane switching on highways, information about trailing vehicles, or vehicles in neighboring lanes were not recorded. Nevertheless, we will attempt to use the available data to model the lane keeping decision making using SVM.

Identification Results and Analysis

We attempt to fit a discrete decision making model of highway lane switching, using the LibSVM package [88]. The results of one trajectory sample of predictions made with the fitted model is shown in Figure 4.12. The second labeling method is used where a lane change decision is made based on the values of the lateral deviations e_y, and the driver's discrete decision to change lanes

is determined as a 1 second window before e_y passes through a threshold value. We can see from Figure 4.12 that even though the model can predict the lane change maneuver 1 second earlier, there are also many false positives (around 120 and 180 seconds) where the model predicts a lane change, but the driver did not execute.

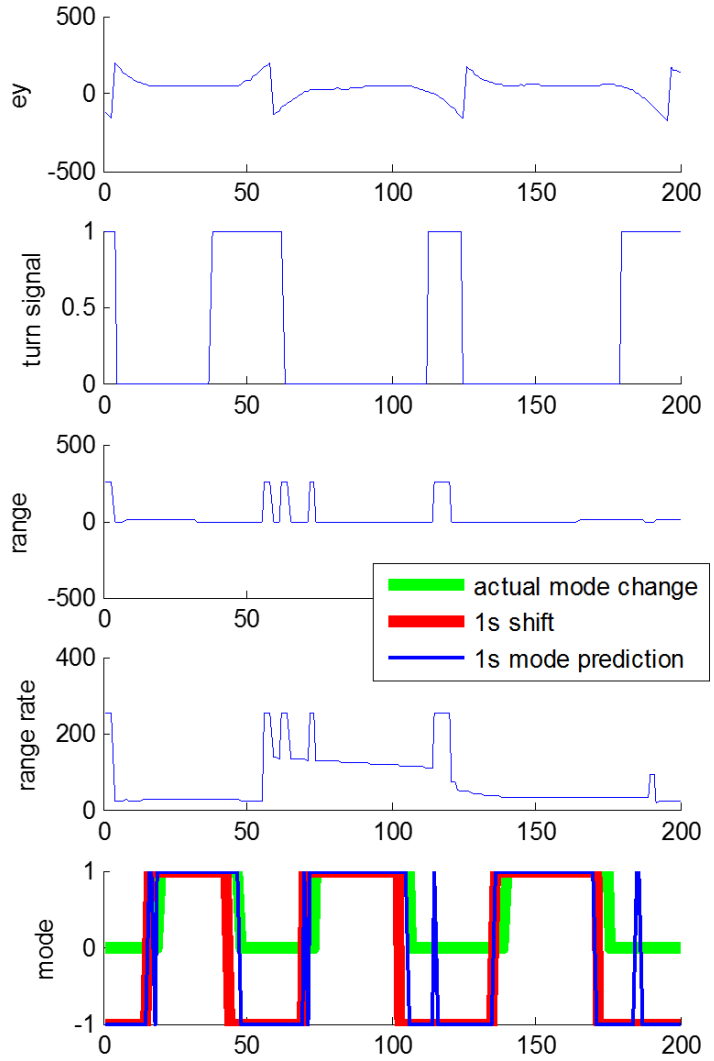


Figure 4.12: Measured vehicle states and comparison between predicted and actual lane switching.

Concluding remarks

We conclude that our formulation of the SVM performs poorly in modeling and predicting of the driver's intent to switch lanes. This could be attributed to the labeling of the data, and also to the

feature vectors available. As discussed earlier, intent can be measured on two abstractions, the first is the intent to change lanes due to traffic conditions conflicting with desired speeds. This is an internal decision made by the driver and better driver sensors such as the use of eye-tracking glasses may be used to better extract the output of this decision making. The second level is more closely associated with directly measurable conditions of traffic flow, such as the cars in neighboring lanes. This second level of decision making follows the first level, where first the driver makes a conscious decision to switch lanes, and then the driver assesses the traffic conditions and the risks associated with changing lanes. Intuitively, this requires more planning and perhaps mental simulations of lane change trajectories in order to avoid hitting other cars.

4.1.5 Limitations of direct methods

So far with our examples of driving and simple gaming, we have attempted to use the direct method to model the policy function of human decision making. Even though direct methods have a more intuitive connection to the structure of weights in the neural network, it is limited in its capability to model more lower-level, repetitive, reactive and less computationally intensive behaviors, such as steering control in lane keeping, direct error feedbacks to old scenarios in the projectile game, or fast steering reactions in extreme maneuvers.

We have seen in some of the presented results that a simple direct method may not be sufficient in capturing the complexity of certain problem scenarios. For example, when new scenarios are presented in the projectile game, it is less intuitive to the subject how much adjustments to make to each of the inputs in order to successfully hit the target. We observe from subject's playing time that the reaction to old trials is much less than those associated with new trials. It appears that subjects could potentially be making predictions and mental simulations of the expected trajectory before committing to the set of control inputs, similar to the study in [109]. Similarly, in the extreme maneuver situation, it is also possible that the driver is trying to follow a learned reference trajectory of state variables rather than using direct feedback. In addition, in the problem with lane changing, drivers seem to also be assessing the risks of traffic conditions on top of the intent to switch lanes i.e. predicting whether the trajectory path of lane changing will collide with any neighboring vehicles.

Seen in both the simple projectile game and real-world driving, we conclude that the mere use of the direct method, although useful in many controls applications due to its ease of identification, is not sufficient to capture the full extent of modeling the psychological principles of human behaviors. More likely a fusion between multiple approaches, such as those which include the optimization-based approach, or the Bayesian induction approach [43], will be able to better model and explain bigger and high-level problems. In the upcoming section of this chapter, we will discuss the use of optimization-based methods.

4.2 Optimization-based policy functions

Contrary to the direct methods discussed in the previous section, optimization-based policy functions are computationally intensive and requires the search for the solution to an optimal problem in real-time. Not to be confused with the policy maps which are pre-computed off-line from optimal frameworks and stored online for function lookup, for example, the state feedback law of linear quadratic regulators [69], or the piecewise affine control law of explicit model predictive control [75], the optimization-based methods referred to in this chapter is adopted explicitly for its nature of solving the optimal problem online, whereas off-line computation of optimal policies will result in function maps which are more in line with the previously discussed direct methods. The higher computational load of the optimization-based policies is one of the main features of this approach, similar to the controlled cognitive processing which require more cognitive attention [65].

The optimization-based method to modeling human behavior poses the decision policy as an optimization problem, with a decision criterion over some function and constraints. First and foremost, modeling human behaviors in this manner is paralleling the notion that subjects intrinsically possess these criterion functions or variables which they are optimizing over. This modeling framework involves the proper definition of a utility function, also known as the cost or reward function. The structure and variables of this cost function is heavily discussed in [64], and candidates such as cost of time, speed-accuracy tradeoff, discounted reward, naturalistic vs. explicitly defined cost function etc., are presented.

The different types of criteria, seen in a wide range of everyday activities, will depend on the context of the activity and based on different variables. For example, in a driving task, the criteria might be to optimize fuel consumption and time while satisfying safety constraints, or in a reaching experiment where the trajectory of the subject's hand can have infinite solutions, and the subject might choose to reach using a combination of shortest distance, smoothest trajectory and also least amount of effort [64]. Another example takes the optimal framework and implements the inverse problem on a robot, where the goal is to convey intent to a subject by optimizing over criterions you expect the subject to be using [115].

The second characteristic of using optimization-based methods to model human behavior is the assumption that humans have an intrinsic representation of internal models, or mapping functions that is part of the optimal problem. For dynamic tasks, these internal models might correspond to differential equations, or state trajectory solutions. Similar to the disconnection between the connectionist and the probabilistic approach to modeling inductive reasoning [44], and even with evidence of the mirror neurons which simulates behavior [56], it is still unclear how such internal model and the optimization of cost can be represented in the central nervous system (CNS). However, like [43], a modeling approach which abstracts away the physical implementations at the neural level yields greater flexibility for exploring the different representations for human behaviors.

We will revisit the application examples discussed in the previous section and draw connections between the principles of optimization-based methods and the observed behavior. First we will pose the projectile game as an optimal problem, and extend this optimal framework to MPC and use it to model extreme maneuvers in driving. We will introduce the dual-task game, model with

the MDP framework and parallel this to distracted driving. Lastly, the observed results of juggling will also be used as an example to illustrate the idea of forecasting for the optimization-based framework.

Recall in Section 2.3 we briefly discussed the issue of identifying parameters for optimization-based frameworks, where unlike the vast number of identification tools and algorithms available for the direct method, identification for the optimization-based method is not so straightforward. Therefore, instead of finding actual parameter values which fits the data, we will simulate the result of the associated model structure and use it to draw comparisons with the observed data.

4.2.1 Non-convex optimization for the projectile game

As mentioned in the concluding remarks of Section 4.1.2, the decision making associated with the projectile game can only be partially modeled by a simple linear gain i.e. for a subset of scenarios. The amount of uncertainty involved in the mapping between the control inputs to state trajectories and the arguable usefulness of the feedback-based features, leads us to investigate the suitability of using the optimization-based framework to model the rest of the problem. We will first present the model formulation associated with the projectile problem, then present and compare the simulated results with the observed behavior.

Model formulation

Recall Section 3.2.2 and Figure 3.15 for the state variables and parameters involved. Building on top of the general optimal control framework posed in Section 2.3, we need to formally define the decision variables u, the cost function J, and the equality and inequality constraints f and g.

Decision variables and the associated uncertainty: To discuss about what consists as the *decision variable* of the human subject, we also need to consider the uncertainty that arises between the actual input intended by the user to what is actually applied to the system. Similar to the case when a person throws a ball, the muscular system and how it interacts with the object introduces uncertainty [116]. Since we are dealing with human subjects and trying to model the internal decision making process happening in the CNS, we would like to pose the decision variable as the *intended* input into the system. The *actual* input applied will have a slight discrepancy to the intended.

In the projectile game example, the inputs are controlled by the subject through the game interface (keyboard). The values set by the subject are the initial angle α_0 and initial velocity v_0 of the projectile. The subject has the opportunity to adjust the values before *releasing* the projectile, and receives feedback both in the forms of numerical values on the screen and the size and angle of arrows in proportion to the input value. Through the subject's perception of the visual and numerical feedbacks, the actual α and v applied to the projectile will be different to the subject's perceived values. The difference will be attributed to factors such as noise in the visual system and memory decay [64]. To define this formally, the actual input values are defined as a function of the

value intended by the user,

$$\alpha_0 = f_\alpha(\alpha_u) \tag{4.11a}$$
$$v_0 = f_v(v_u) \tag{4.11b}$$

where α_u and v_u are the values intended by the subject and are the decision variables in the model,

$$u = \begin{bmatrix} \alpha_u \\ v_u \end{bmatrix} \tag{4.12}$$

As mentioned before, this input discrepancy map can be defined in many ways factoring in different elements that would cause the discrepancy. For the projectile example, we will define the input discrepancy map to be,

$$\alpha_0 = \alpha_u + \varepsilon_\alpha \tag{4.13a}$$
$$v_0 = v_u + \varepsilon_v \tag{4.13b}$$

where ε_α and ε_v are random variables used to capture the uncertainty in the discrepancy between the subject's perceived values and the actual input applied to the system.

Different probability densities can be used to represent the uncertainty, and we can use different densities to capture effects of different factors. For example, the distribution can be specified to be dependent on the previous input value u^-. In this case the function map in (4.11a) can be defined as a joint distribution between the intended input with the previous input u^-, tying together the dependence of the uncertainty on the feedback the subject receives from the previous trial.

Consider the effect of memory decay of the subject's estimation of how the visual and numerical feedbacks are related to the actual input applied to the system. If the intended input is close to the previous input then the subject already has received feedback on how this maps out, and the relative uncertainty will shrink. On the other hand, if the intended input is drastically different from the previous, then the subject is more likely to have forgotten the mapping function and therefore the uncertainty is larger.

Equality constraints: The equality constraints in the projectile example corresponds to the mapping related to the kinematics of the problem. The projectile scenario simply involves gravity in the vertical direction, i.e. $\ddot{y} = g$. The states of the particle are positions and velocities in the horizontal and the vertical direction x, y, $\dot{x}$ and $\dot{y}$. The solutions to the differential equations governing this dynamic system are posed as the equality constraints,

$$x(t) = \dot{x}_0 t \tag{4.14a}$$
$$y(t) = \dot{y}_0 t + 0.5gt^2 \tag{4.14b}$$
$$\dot{x}_0 = v_0 \cos\alpha_0 \tag{4.14c}$$
$$\dot{y}_0 = v_0 \sin\alpha_0 \tag{4.14d}$$

recall that α_0 and v_0 both depend on the decision variable shown in (4.12). Respectively, the subject can determine how particle states map out by controlling the initial velocity and angle. The only parameter value in the constraint is g, which denotes the gravity, formally the acceleration in the vertical direction. This value, although predefined and determined by the experiment as part of the environment parameter is unknown to the subject, and over time, subject will learn and have an estimation of this value.

Inequality constraints: The inequality constraint involves both state and input constraints. Input constraints involves limitations set on the decision variables, which in our projectile example is set between $\underline{u}$ and $\bar{u}$. The state constraints correspond to the relative distance between the projectile trajectory and the target / obstacle positions. Since the target / obstacles are circular with radius R, avoiding the obstacle is equivalent to having a minimum distance between the particle and the obstacle. These constraints are formally defined as,

$$\exists\, t > 0,\ \|p(t) - p_{\text{targ}}\| < R_{\text{targ}} + R_{\text{proj}} \tag{4.15a}$$
$$\forall\, t > 0,\ \|p(t) - p_{\text{obst}}\| > R_{\text{obst}} + R_{\text{proj}} \tag{4.15b}$$
$$y \leq Y_{\text{screen}} \tag{4.15c}$$
$$u \in \mathcal{U},\ \mathcal{U} = \{u : \underline{u} \leq u \leq \bar{u}\} \tag{4.15d}$$

where $p(t) = (x(t), y(t))$, $p_{\text{targ}} = (x_{\text{targ}}, y_{\text{targ}})$, and $p_{\text{obst}} = (x_{\text{obst}}, y_{\text{obst}})$ are the x and y coordinates of the projectile, target and obstacle respectively. Constraint (4.15a) corresponds to the requirement for the projectile to hit the obstacle at any point in time; Constraint (4.15b) corresponds to the requirement for the projectile to avoid the obstacle for the entire trajectory; Constraint (4.15c) corresponds to the upper bound for the y coordinate of the projectile due to screen limitations; and Constraint (4.15d) corresponds to the upper and lower bounds on the control input, which in our experiment is set at $\alpha \in [0, \pi]$ and $v \in [0, 200]$. Notice Constraint (4.15b) makes the optimization problem non-convex.

Cost function: The cost function for the projectile example will depend on the distance to the target, and since uncertainty is added, the function will be the mean squared error $e(t^*) = p(t^*) - p_{\text{targ}}$ of the particle position to the target reference,

$$J_1 = E[e(t^*)^T e(t^*)] \tag{4.16}$$

and since the particle positions x and y is parameterized by time t in Equation (4.14), we include t^* in the optimization variable. The presence of uncertainty in v and α will affect the state trajectory and J_1. We introduce two additional constraints that could possibly be used by the subject in order to directly minimize the effects of uncertainty on state evolutions.

$$J_2 = \Delta u^T Q \Delta u \tag{4.17}$$
$$J_3 = t^* \tag{4.18}$$

where Δu indicates the change in the control input between trials and Q is the weight matrix between $\Delta\alpha$ and Δv. J_3 will try to find the minimum time required to approach closely to the target. With the model formalized, we will now present the result and analysis, which will include both analytical and numerical solutions.

Analytical and numerical results and analysis

In this section, we will describe several phenomena and behavioral patterns we observed throughout the subjects' experimental data, and attempt to capture the fundamental principles behind specific decisions. Model structures and parameters of the optimal framework described before will be used to reason subjects' responses, either analytically through the model, or through numerical simulations and solving of the optimization problem.

Effects of uncertainty on the decision making in trials without obstacles: Starting off with the simplest scenarios, the first set of trials presented to the subjects when they start the experiment are the ones without obstacles. Subject are asked to hit the target without any obstacles constraining the flight trajectory. We observe that even though initial control inputs might vary greatly among subjects, after a few trials the subjects learn the factors involved, and they all eventually converge to the same control strategy. The strategy involves increasing the initial velocity to the maximum level for all trials, and only manipulate the initial angle to match different target positions. This can be seen in Figure 4.13, which shows the various control strategies over several subsequent trials.

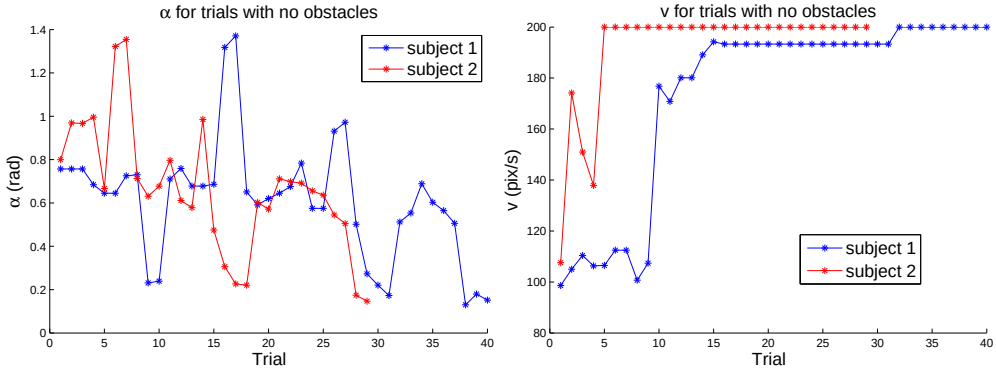


Figure 4.13: Control inputs for scenarios without obstacles.

This decision making strategy can be captured by the amount of uncertainty that is included in each of the control inputs, and of the environment. Recall Equation (4.13a), the intended input of the subject and the actual input applied to the system is biased and corrupted by noise. For the initial angle α, the subject receives direct visual feedback through the arrow indicator, therefore the added noise only arises from the visual-sensory system itself, which is very minimal compared to the amount of uncertainty included in the control input for initial velocity v. Because the indicator for v, provided both numerically and visually, still has to rely on the mapping through the subject's memory from previous trials, the amount of uncertainties involved is much greater than for α.

In addition, there is uncertainty associated with the gravitational constant g, which also relies on memory and learning from the subject.

To connect this line of cognitive reasoning with the model framework, consider the extreme case where the uncertainty in α is much smaller than in v, such that α is posed to be deterministic and only v to be the random variable. Reformulating Equations (4.14) and eliminating t, the new trajectory map for x and y can be presented as,

$$y = \tan(\alpha)x + \frac{gx^2}{2v^2\cos^2(\alpha)} \tag{4.19}$$

Notice v is eliminated in the first term, and appears in the denominator of the second term. g also only appears in the second term. Therefore, as $v \to \infty$, the second term goes to zero, and we are left with $y = \tan(\alpha)x$, which is deterministic since we have reduced α to be deterministic. Therefore, by increasing the size of v, we reduce the effect of the uncertainty in v and g on the projectile trajectory, which aligns with the control strategy adopted by the subjects.

This supports the conjecture that, from the subject's perspective, the uncertainty associated with v is much greater than the uncertainty associated with α, and that the subject's control strategy aligns with the principle of minimizing the resulting uncertainty of the predicted state trajectories. Since we have established that v has a much larger uncertainty relative to α, from here forth we will simplify the analysis by assuming α to be deterministic.

Analyzing decision making with one obstacle: After the subjects have finished the trials with no obstacles and have become familiar with the dynamics and mechanics of the game, we present subsequent trials where a single obstacle is present. Analysis of the recorded data from the subject's game plays and from post-game feedbacks, shows that there are several noteworthy observations regarding the control strategies of the subjects. To summarize, the patterns show control preferences towards the following factors:

- minimum time / trajectory path length
- minimum change in control
- preference towards zero change in $\Delta\alpha$

The above listed conjectures each correspond to a specific formulation of the cost and constraint functions of the optimal problem. We will discuss these behavioral patterns and compare with various baseline solutions of the optimal framework presented in the model formulations. The baseline solutions is computed using YALMIP [117] and the general purpose nonlinear solver IPOPT [118, 119]. The technical details of how the baseline solutions are computed is included in Appendix A.1.

Control strategy of minimum time: We observe that the subjects' control strategy closely relates to an optimal control problem which minimizes the time to hit target, i.e. the cost function includes term (4.18). This can possibly be attributed to the uncertainties associated with the control inputs and the environment. Since we are dealing with a minimum time formulation, we solve the optimization problem (2.9) using with two cost criterions: $J = t_{\text{targ}}$ and $J = -t_{\text{targ}}$ with $y(t) < 672$ due to screen constraints, which provides our analysis with baseline solutions of lower and upper bounds, i.e. minimum time to hit target and maximum time to hit target.

The trajectory shown in Figure 4.14 compares baseline solutions with the subject's response. We can see in this example the subject's response closely follows the lower bound of the baseline solution with minimum time to hit target. In fact, for certain scenarios and obstacle-target configurations, subjects have generally shown preference towards choosing the set of initial conditions from the feasible set hat results in shorter flight trajectories.

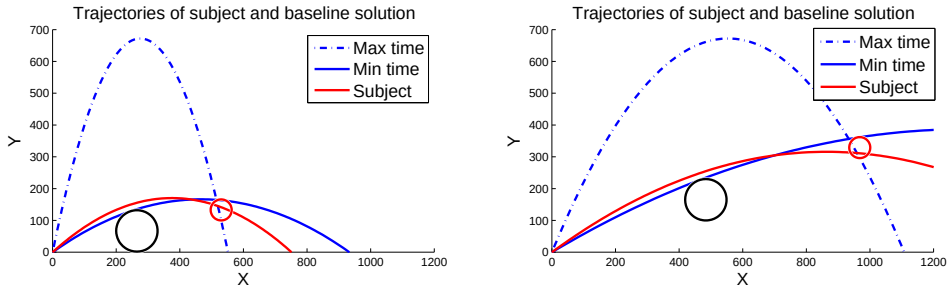


Figure 4.14: Comparison of trajectories between subject input and the maximum and minimum time baseline solutions.

We can directly relate this biased preference towards minimum time to the control input α. Consider system dynamics defined by (4.22)-(4.14d), larger α will result in longer flight trajectory in order to reach the target. Subjects in general will choose a smaller α in order to minimize the length of the flight trajectory. Figure 4.15 shows the distribution of subject input relative to the maximum and minimum time baseline solutions. The values in the shown distribution are the ratio $\rho_{t_{\text{min}}}$ of the difference between subject input and minimum time baseline solution over the difference between the maximum and minimum time baseline solutions,

$$\rho_{t_{\text{min}}} = \frac{\alpha_{\text{subj}} - \alpha_{t_{\text{min}}}}{\alpha_{t_{\text{max}}} - \alpha_{t_{\text{min}}}} \tag{4.20}$$

$\rho_{t_{\text{min}}} = 0$ means the subject's control strategy is equivalent to the minimum time baseline solution, and similarly $\rho_{t_{\text{min}}} = 1$ would equal the maximum time baseline solution. The distribution in Figure 4.15 shows the mean value of $\rho_{t_{\text{min}}} = 0.26$ and a standard deviation of 0.1, illustrating the preference that subjects are biased towards the minimum time solution.

This particular control strategy can be explained by looking at the uncertainty involved in the system dynamics. As explained in the no obstacle scenarios, we can assume that there is no or

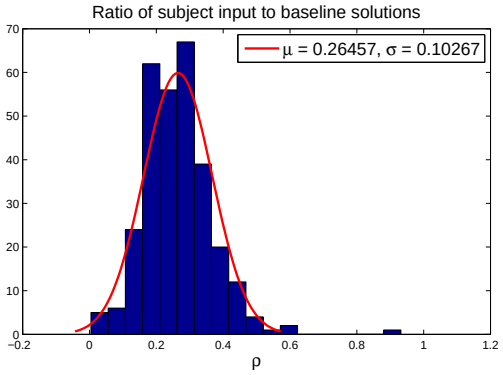


Figure 4.15: Distribution of subject input relative to the $t_{\max}$ and $t_{\min}$ baseline solutions.

very minimal uncertainty involved in the initial α when compared to v. There is also a considerable amount of uncertainty associated with the gravitational constant g as this is also a learned parameter from feedback of previous trials subject to memory decay.

To simulate the effect that uncertainty plays in the flight trajectories, Figure 4.16 compares three different sets of flight trajectories computed from the worst case bounds on the uncertainty in $g \in [\bar{g} - \delta g, \bar{g} + \delta g]$ and $v \in [\bar{v} - \delta v, \bar{v} + \delta v]$, where given a specific value of α and the target positions $(x_{\text{targ}}, y_{\text{targ}})$, the nominal value of $\bar{v}$ satisfies (4.22)-(4.14d) reformulated as,

$$v^2 = \frac{\bar{g} x_{\text{targ}}^2}{2\cos^2\alpha(y_{\text{targ}} - x_{\text{targ}}\tan\alpha)} \tag{4.21}$$

The results in Figure 4.16 is obtained with $\bar{g} = -10, \delta g = -1, \delta v = 5$ and $\alpha = \{\frac{\pi}{8}, \frac{\pi}{4}, \frac{3\pi}{8}\}$. We can see that with a larger value of α, the resulting state trajectory will reach the target at a longer t, and the predicted bounds at target is bigger than for $\alpha = \pi/4$, making it is less likely to hit the target. The limiting factor will be the position of obstacle which obstructs the path.

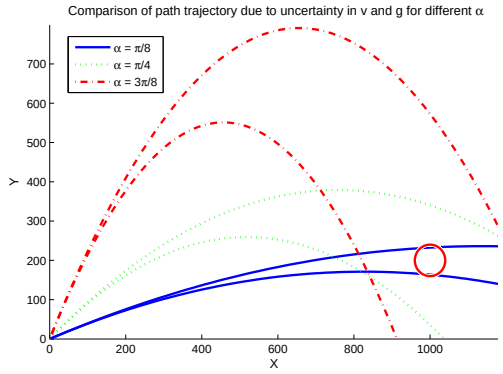


Figure 4.16: Comparison of path trajectory due to uncertainty in v and g for different α.

Control strategy of minimum change in input: The next set of observations and behavioral trends we see from the subjects are the preferences towards minimum change in control inputs. This is equivalent to including (4.17) in the cost function. Similar to the analysis for the previously discussed minimum time formulation, we solve optimization problem (2.9) to find the baseline solutions for: minimum time to hit target (cost term (4.18)); minimum change in α (cost term (4.17) with zero weight on Δv); and minimum change in v (cost term (4.17) with zero weight on $\Delta\alpha$).

Shown in Figure 4.17, are comparisons between the subject's change in the input control with the baseline solutions for both $\Delta\alpha$ and Δv. Three different baseline solutions are computed using the optimal framework in (2.9) , and different cost functions minimizing exclusively either change in α, change in v, or time to hit target t_{targ}.

Firstly, looking at the distribution of the baseline solutions shown in blue, we can see the solution for the minimum time to hit target formulation has a bias towards a decrease in α and an increase in v; $\Delta\alpha$ for the minimum $\Delta\alpha$ formulation is also mostly zero, which means no change or very little change in α is still feasible in many cases; and Δv for the minimum Δv, although also centered around zero, has a slightly larger distribution.

Now we focus and compare the subject's input to the baseline. Like the formulation for minimum $\Delta\alpha$ and minimum Δv, the subject also has a distribution where the mean is centered on zero change in α and v, showing that subject prefer to minimize the change in input. Looking at the plot for 'Δv from subject vs. Δv from min$\Delta\alpha$', we can see the distribution is almost identical, whereas in the plot for '$\Delta\alpha$ from subject vs. $\Delta\alpha$ from minΔv', the distribution is vastly different. This shows that, although subjects generally prefer to minimize the amount of change to apply to the control inputs, there is a stronger bias towards putting more weight on minimizing $\Delta\alpha$ over Δv.

Scenario dependent hybrid control strategy: Similar to our analysis of the projectile game using direct methods in Section 4.1.2, we follow the conjecture that the control strategy will follow a hybrid structure which is scenario dependent. Focusing specifically on the partitioning of scenarios based on whether it is new or old (repeating failed trials), we plot the partitioned data against the same baseline solutions discussed in the previous section. Figure 4.18 shows the results of the new scenarios and Figure 4.19 shows the results of the old scenarios.

Several observations and comparisons can be made from Figure 4.18 and 4.19. Firstly, looking at the subject's distribution in the new scenarios, even though Figure 4.18(a) and 4.18(d) shows that the baseline solution has successfully obtained close to zero values of $\Delta\alpha$ and Δv to be feasible, the subject's distribution, although centered around zero, has a wider distribution. In addition, Figure 4.18(c) and 4.18(b) shows the distribution between the subject and the baseline solution to be closely matched.

In comparison, looking at the result in old scenarios, distributions shown in Figure 4.19(a) and 4.19(b) shows that the subject's control strategy follows closely with the baseline solutions that are computed from the minimum $\Delta\alpha$ formulation. On the contrary, Figure 4.19(c) and 4.19(d) shows that the control strategy does not align with the minimum Δv formulation quite as well. In addition, there is also a slightly better fit with the baseline solution of the minimum time to hit

target formulation in the new scenarios as seen in Figure 4.18(e) and 4.18(f).

From the above observations, we can conclude the following. Firstly, the subject uses a different set of decision making strategies when encountering different scenarios. Specifically, in the old scenarios, it appears that the subject prefers to adjust the velocity input over the angle, putting more weight on minimizing $\Delta\alpha$ than on minimizing Δv or t_{targ}. Conversely, in the new scenarios, there appears to be more of an equal balance of weighting on all the terms, minimizing both $\Delta\alpha$ and Δv as well as t_{targ}. $\Delta\alpha$ is more likely to be dominating when the trajectory needs to avoid the obstacle and the only feasible solution is to alter the flight trajectory through increasing $\Delta\alpha$.

We can possibly attribute this behavioral pattern to the idea that human subjects are better at deducing linear relations, and are more likely to assign linear relationships to functions [120]. This explains why subjects are more likely to manipulate v over α, since v appear linearly in (4.14c). Recall from (4.14) the x and y coordinates of the trajectory can be defined in terms of α and v,

$$x = v\cos(\alpha)t \tag{4.22}$$

$$y = v\sin(\alpha)t + 0.5at^2 \tag{4.23}$$

Differentiating with respect to v and α,

$$\frac{dx}{dv} = \cos(\alpha)t, \quad \frac{dy}{dv} = \sin(\alpha)t \tag{4.24}$$

$$\frac{dx}{d\alpha} = -v\sin(\alpha)t, \quad \frac{dy}{d\alpha} = v\cos(\alpha)t \tag{4.25}$$

As shown, Δv affects both Δx and Δy linearly, whereas $\Delta\alpha$ is nonlinear. Linearization can be performed on this nonlinear relationship about the current α, however, is restricted to small regions for accuracy. Especially since the sine and cosine functions are complementary to each other, it is harder for subjects to learn about this nonlinear relationship, as evident from the preference towards the larger distributions in Δv over $\Delta\alpha$ in Figure 4.19.

Connecting the role of feedback to control strategies: From a psychology standpoint, the feedback the subject receives from previous trials helps the subject in solving the inverse problem where the parameters of the input mapping (4.22) are identified. However, the feedback from previous trial are stored in short term or working memory which is subject to decay and memory detail loss [65], which means over time, it is most likely that the feedback trajectory received from the most current trial replaces other trajectories from previous trials. This combined with the tendency for human subjects to interpolate relationships linearly [120], leads to the input map to have distributions where estimations around the most recent input will have the least amount of uncertainty and estimations that are vastly different from the current input will have a great amount of uncertainty, which has been shown from Figure 4.19 that the uncertainty associated with deviating from the current control inputs far exceeds the accumulation of uncertainties associated with a longer flight trajectory.

In other words, we can pose this relationship between the current and the previous input as having a joint distribution. In turn we can model the effect of a given previous input on the estimation

of the current and compute the conditional densities of the current input given the previous. To formally present this, define the input vector (game control) selected by the subject for the current scenario (trial i) as $u_s^i = (\alpha_s^i, v_s^i)$, the input vector that the subject estimates will be applied to the system for trial i as $u_a^i = (\alpha_a^i, v_a^i)$, and the input vector (game control) selected by the subject for previous $i-j$ trials as $u_s^{i-j} = (\alpha_s^{i-j}, v_s^{i-j})$, then the conditional probability of the applied input can be posed as $p(u_a^i|\theta^i)$ where θ^i represents the parameters that specifies the distribution, i.e. the mean and standard deviation for a Gaussian distribution, and is conditional on the current and previous game controls as well as the trajectory feedback O^{i-j} the subject receives from applying $u_s^{i,k}$. The marginal distribution for u_a^i given previous control and feedback observations pairs,

$$p(u_a^i|u_s^i, u_s^{i-1}, ..., u_s^{i-N}, O^{i-1}, ..., O^{i-N}) = p(u_a^i|\theta^i)p(\theta^i|u_s^i, u_s^{i-1}, ..., u_s^{i-N}, O^{i-1}, ..., O^{i-N}) \quad (4.26)$$

To simplify the equation, assume the memory from the most current feedback dominates the estimation of θ^i reducing to $j = 1$. In this case, $p(\theta^i|u_s^i, u_s^{i-1}, O^{i-1})$. Observations helps the subjects solve the inverse problem where the initial conditions are identified for the associated u_s^{i-j}. To find a connection between Equation (4.22) and why subjects prefer to minimize the change in control inputs, consider two different feasible game controls the subject can choose to apply $u_{s,1}^i$ and $u_{s,2}^i$, where $\Delta u_1 = u_{s,1}^i - u_s^{i-1}$ and $\Delta u_2 = u_{s,2}^i - u_s^{i-1}$, with the condition that $\|\Delta u_1\| < \|\Delta u_2\|$. Suppose u_a^i $\mathcal{N}(\mu, \sigma)$, $\theta^i = (\mu, \sigma)$ specifies a Gaussian distribution with the size of σ positively correlated with the size of $\|\Delta u\|$. Then u_2^i, with the larger change in control will result in a wider distribution. Substituting into Equation (4.22), this will give a higher probability of failure, and therefore subjects are more likely to choose a smaller control difference to minimize the uncertainty.

Concluding remarks

In this section, we have presented the decision making associated with the projectile game as an optimal control problem. Similar to the discussion about how reward functions can be formulated in [64], we investigated the structure of the cost function in our experiment. We have found that the subject adopts a hybrid strategy that is dependent on different scenarios and the uncertainties associated with the control inputs and system dynamics. In new scenarios, subjects will optimize over a combination of minimum time to hit target and minimum deviations from current control inputs, whereas in old scenarios, subjects adopted a strategy which focused on minimizing deviations from the current initial angle. This can be attributed to how subjects view the system as locally linear functions.

Typically, motor studies deals with reaching experiments where the subject has an infinite number of hand trajectories and optimizes for minimum control effort, smoothness etc. [64, 73]. In our experiments, we extend this concept by introducing nonlinearity into the trajectory mapping, as imposed by the simulated gravity in the projectile game, and also including avoidance regions posed by the obstacles. By introducing nonlinearity and obstacle constraints into the problem formulation, we can draw parallels to the task of driving, where the driver is also dealing with constraints and nonlinear regions of the tires. We use a similar optimal framework in the next section to model driving under extreme conditions.

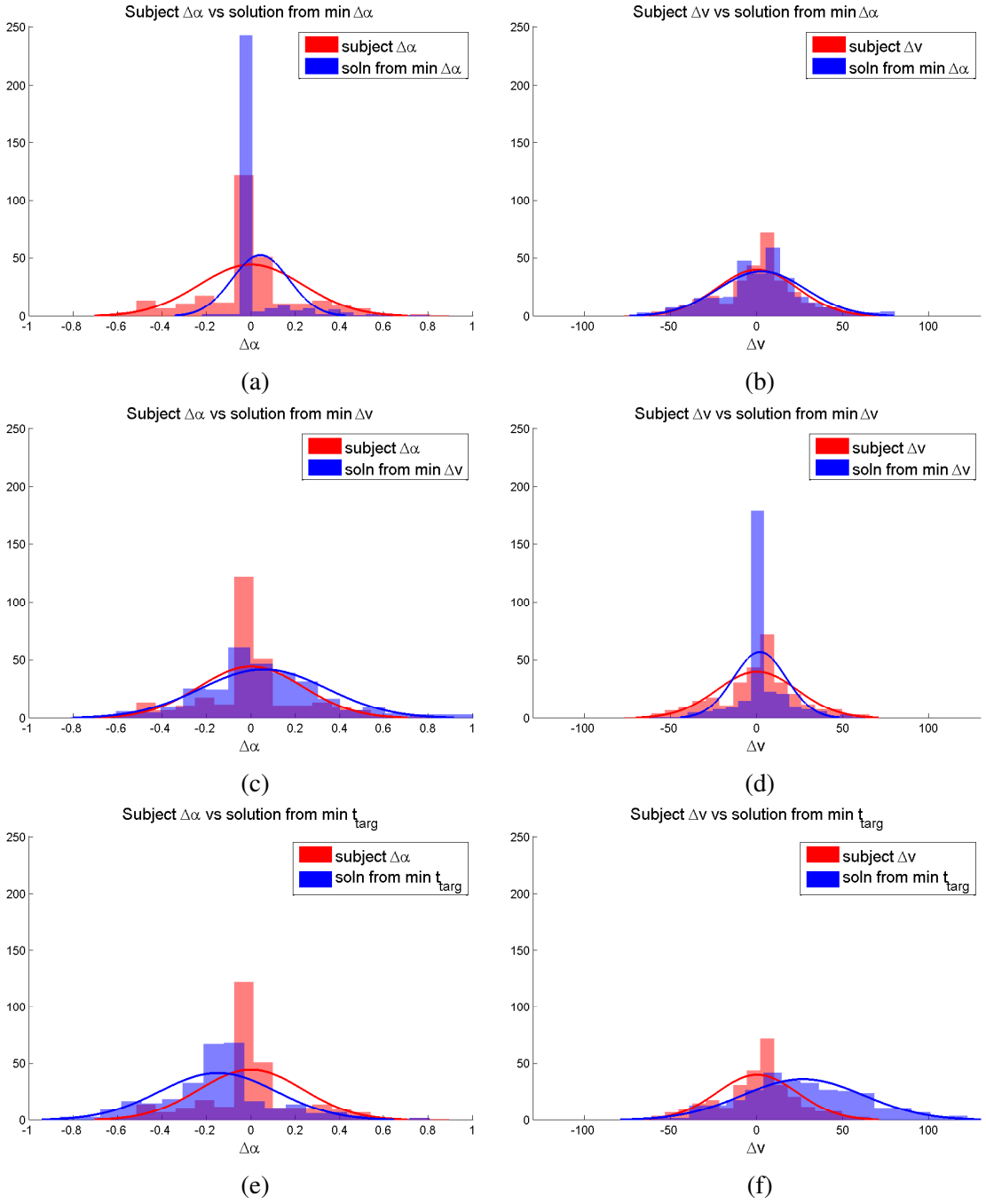


Figure 4.17: Comparison of subject $\Delta\alpha$ and Δv with baseline solutions for all trials.

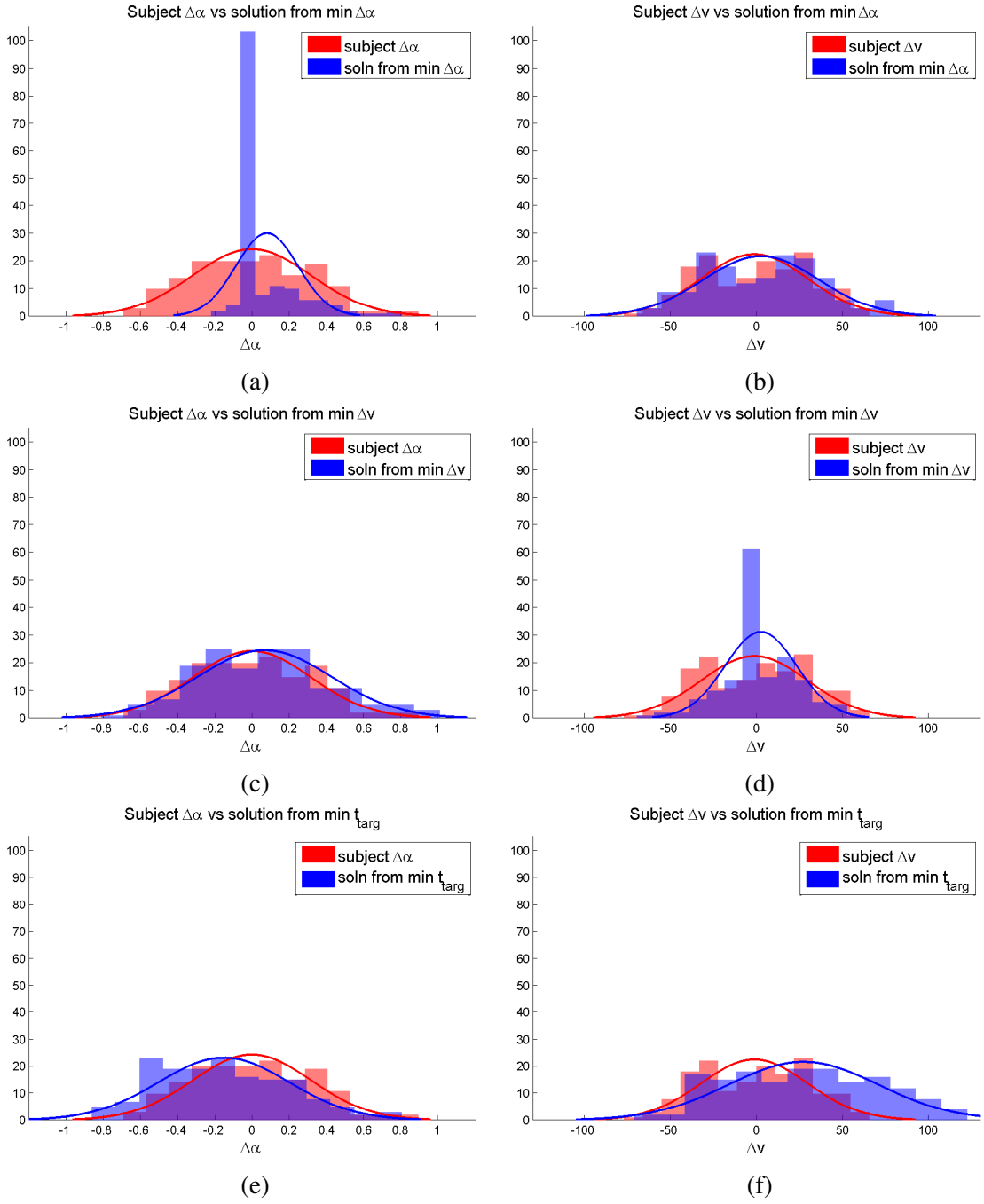


Figure 4.18: Comparison of subject Δα and Δ*v* with baseline solutions with only new scenarios.

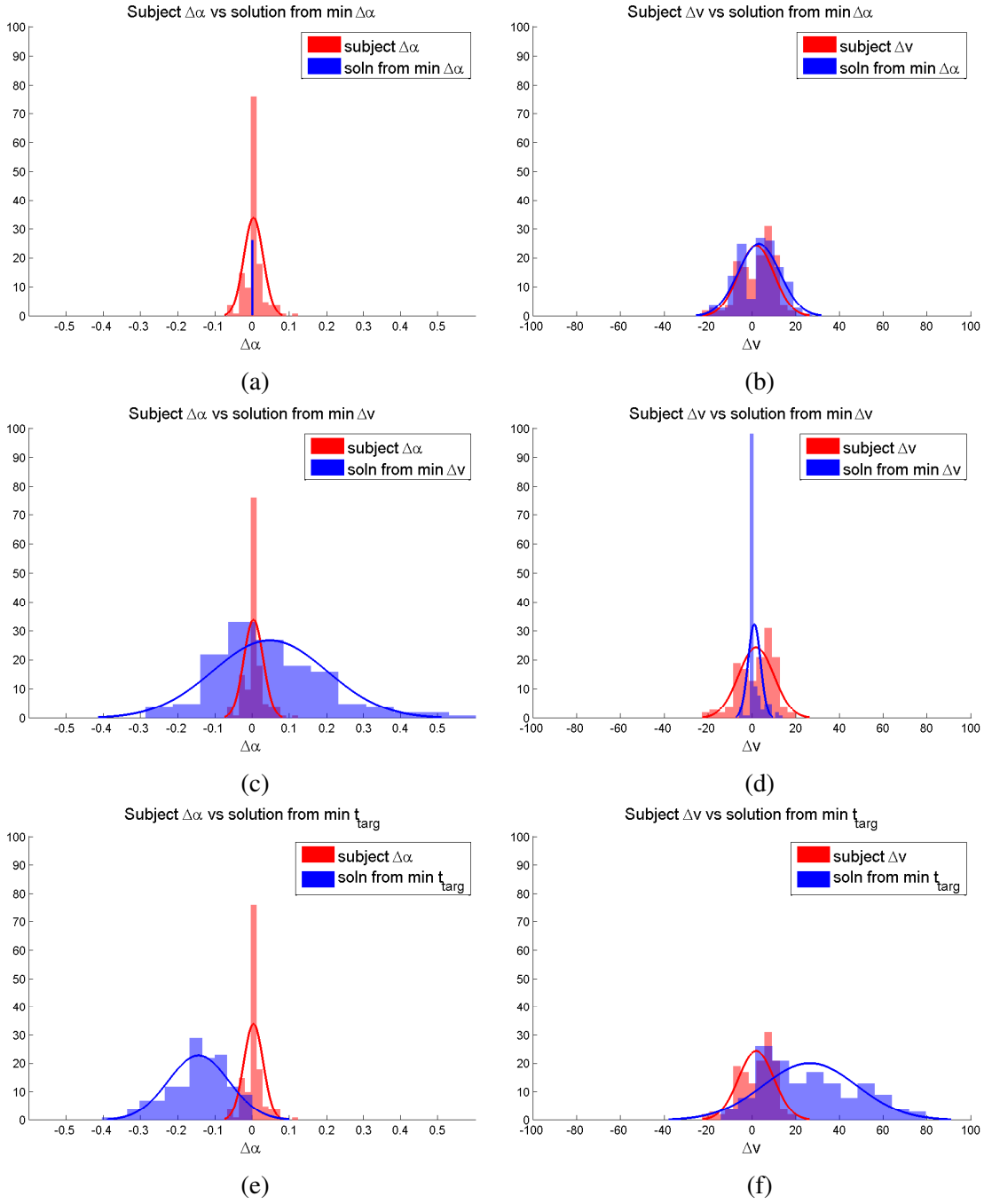


Figure 4.19: Comparison of subject $\Delta\alpha$ and Δv with baseline solutions with only old scenarios.

4.2.2 MPC model for extreme driving on slipper roads

Using the same optimal framework as the projectile game, we will now extend the game-based application to real-world examples, specifically in the context of driving. For the steering control introduced in Section 4.1.1 and in [121, 122], direct methods which maps observed feedback to steering input through a linear gain is a suitable candidate for modeling simple highway driving on typical road conditions. Applications like this, where the task is familiar to the person, align more with the principles behind direct methods where the decision making is more on the lower-level and responses are more reflexive rather than predictive.

Lane keeping on the highway deals with small side slip angles and therefore primarily involves only the linear region of tire dynamics familiar to the driver. However, as reported in [123], many car accidents involve slippery surfaces and nonlinearity of the tires, which typical drivers have little experience with. Driver training on skidding as part of the license requirement [124] is proposed to increase their knowledge on how to respond to the unexpected change in vehicle dynamics involving the full nonlinearity of the tires.

In this section, we will model the behavior of an experienced driver who is familiar with the nonlinear dynamics involved in driving on slippery surfaces, and knows how to utilize the full nonlinearity of the tire to first induce tire saturation and slipping, and then stabilizing the vehicle from the drift. With the idea that the experienced driver has knowledge of and an internal representation of the vehicle dynamics and the nonlinearity in the tire, the MPC framework introduced in Section 2.3.1 can be used to model decision making based on mental simulations and state predictions.

Previously in Section 4.1.3, we modeled the same maneuver using direct methods and the PWA framework postulating that after many trials, the subject learns the linear gains for two different modes of operations, saturated and unsaturated. In this section we postulate that the subject learns a reference trajectory over time of all vehicle states for the 180 degree turning maneuver, and using a model predictive framework and knowledge of the vehicle dynamics to choose the control variables which will follow the learned reference trajectory.

Model formulation and associated variables

Following the MPC framework described in 2.3.1, we will model the extreme maneuver on ice using the predictive approach. First the cost function $J(x)$ in (2.10) is defined for the driver application as,

$$J_N(\bar{\xi}_t, U_t, \Delta U_t) = \sum_{k=t}^{t+H_p-1} \|\xi_{k,t} - \xi_{ref_{k,t}}\|_Q^2 + \|u_{k,t}\|_R^2 + \|\Delta u_{k,t}\|_S^2 \tag{4.27}$$

i.e. by minimizing a weighted combination of (i) deviations in the reference trajectory, (ii) the control input $u_{k,t}$ and (iii) the rate of change in $u_{k,t}$. The subscript of the norms denotes $\|x\|_P^2 = \|Px\|^2, P = Q,R,S$.

The cost function in (4.27) involves the tracking of a reference trajectory by minimizing state deviations from the reference states, which requires a mental representation of the vehicle model in order to make state predictions over the upcoming horizon. This leads to the following questions. Firstly, what *reference trajectory* is the driver trying to track? As proposed in Section 2.1.1, the

reference generator will either be a higher level optimization-based path planner similar to [67], or simply looks up stored reference trajectory from memory.

Secondly, how is the driver making *predictions of vehicle states*? In this section, we propose the driver is predicting state evolutions from the nonlinear model described in Section 3.1.1. But the driver could simply be retrieving stored pieces of trajectories from a look up table [89], or using a simpler dynamical model such as the point mass model [67] with a linear tire. This could possibly be the difference between expert and novice driving. The expert driver is able to make better predictions using a nonlinear model, whereas the novice driver is only using the linear model, and would fail to act appropriately in slippery conditions.

Thirdly, what is the *prediction horizon* of the driver? A long prediction horizon considers long term goals, but are computationally expensive. On the other hand, short prediction horizons require less computation, therefore can be updated more often. Shorter horizons are however, short-sighted, and often requires a good reference. For example, using this predictive framework to model the different levels of the driver hierarchical scheme mentioned in [125]. For a driver who wishes to minimize total traveling time, at the strategic level, the driver makes predictions of traveling time based on distance and traffic information, and makes a decision on which roads to take. This is one long horizon prediction and is seldom updated unless something unexpected comes up (e.g. unexpected traffic). At the maneuvering level, the human has to make decisions on maneuvers such as running a yellow light or passing a slow vehicle. The prediction horizon in this level is a few seconds, and gets updated often. Then at the control level, the driver constantly makes decisions on the steering/braking/throttling controls i.e. to maintain speed limit, lane keeping etc.

For the extreme drift maneuver described in Section 3.1.6, we take the state trajectory of a successful maneuver which the driver has learned as the reference trajectory, and we model the driver at the control level by treating the driver as a MPC controller.

Results and model simulations

Figure 4.20 shows the steering input and the simulated state trajectories of a 180 degree drift turn when the MPC formulation in (2.10) is solved using the nonlinear solver NPSOL [90]. The sampling time is 50 ms and $H_p = 160$. The trajectory obtained from an expert driver is used as the reference trajectory for the model simulation (red dash-dotted line).

Figure 4.20 indicates that an optimal control framework is suitable in replicating driver behavior. The solid line in Figure 4.20 shows the simulated state and input trajectories when the nonlinear Pacejka tire model [94] and (3.1) is used to make state predictions in (2.10b) and in the simulation. We can see that the simulated state trajectories tracks the reference trajectories reasonably well, and the input trajectory from the solver has a similar profile with the driver steering. Note that in the MPC formulation, the difference between solver steering and driver steering is not penalized in the cost function, which is why the steering profiles do not exactly match. Yet even though there are large quantitative differences, the qualitative behavior (the general shape/pattern of the steering profile), is quite similar. The quantitative differences are due to model mismatches between the test vehicle and the model used in the solver and the uncertainty in the actual road conditions. A more accurate model would reduce this mismatch.

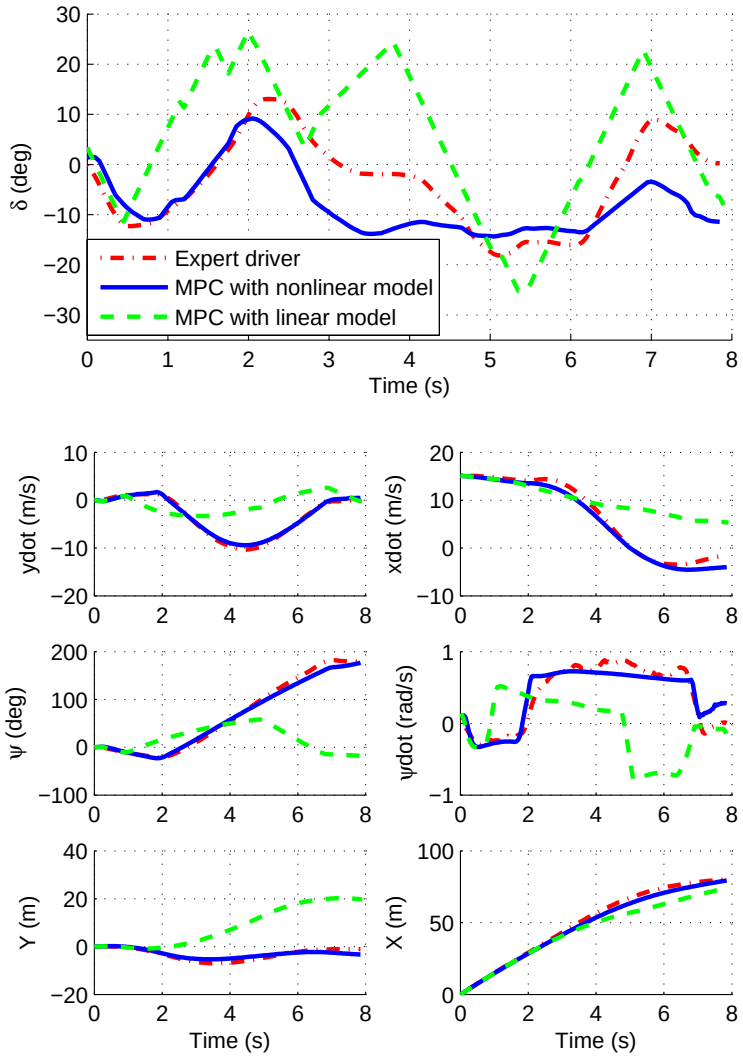


Figure 4.20: Comparing expert driver and MPC result of drift maneuver using both nonlinear and linear approximation of tire model: input trajectory δ and state trajectories $\xi = [\dot{y}, \dot{x}, \psi, \dot{\psi}, Y, X]$.

Figure 4.20 also shows that for complex maneuvers which involve the saturation regions of the tires, a high fidelity model is needed in the predictive framework. Most importantly, the nonlinearity in the tire forces needs to be recognized by the driver/solver in order for the drift maneuver to be performed. If the tire forces are treated to be only linear in the tire side slip angle, then the driver/solver will not be able to track the reference trajectory that replicates this drifting maneuver. This is shown by the state and input trajectories represented by the dashed line in Figure 4.20. The input trajectory is solved when the tire force in (3.1) is only obtained from a linear tire model (the dashed linear approximation in Figure 3.5. The state trajectories are simulated by passing this input profile through the original nonlinear model.

Concluding remarks

We have shown that the optimization-based method using the MPC framework can be used to model and simulate driver behavior in more complex driving scenarios involving slippery conditions and nonlinear dynamics involving the tire. Similar to the projectile game experiment, the subject/driver requires knowledge of the system dynamics in order to make mental simulations of state evolutions in order to find the most optimal set of control inputs to apply to the system. Subjects are required to hit targets while avoiding obstacles in the projectile game and therefore there are certain coordinates in the state space which the subject should and should not enter. Similarly, in the extreme driving situation, drivers are given guidelines, either learned from demonstration or from memory, on what the vehicle states should be over the next horizon and tries to track this reference trajectory over the next horizon.

Up to this point, we have introduced the use of optimization-based methods in the modeling of control strategies that only involved continuous decision variables. Next, we will attempt to model discrete decisions with the same optimization-based approach, specifically by using the MDP framework to model decision making in a dual-task game and a paralleling situation in the real-world with distracted driving.

4.2.3 MDP model for attention allocation in the dual-task game

Recall in the projectile game, we introduced the notion that subjects might be using a different set of control strategies for different scenarios and partitioned the data set accordingly to demonstrate this idea. One of the partitioning characteristic was how the subject selected the control inputs and whether it involved adjusting only the angle, only the velocity or both simultaneously. This precursor to the finer control of continuous inputs, such as the actual values of angle and velocity, can be an example of the decision making process involving discrete variables.

In Section 4.1.4, we introduced how the direct method of SVM mapping can be used to model the discrete intent to change lanes on the highway based on the traffic conditions, and concluded that for this particular application, there might be more planning and predictions involved in the assessment of safety and therefore an optimization-based method might be more suitable to capture the predictive nature of the decision making process.

In this section, we will use MDP, an optimization-based framework introduced in Section 2.3.2 to model the dual-task problem presented in Section 3.2.3. As previously described, the dual-task game requires the subject to perform a primary task which involves avoiding obstacles, at the same time completing a secondary task which involves typing in strings of numbers. The MDP framework provides an intuitive way to model this decision making process. The states and actions are naturally discrete and the subject receives reward for completion of the secondary task and receives penalty for hitting obstacles, so there is a tradeoff between the two tasks. In the results discussion we will use the MDP framework to simulate the ideal behaviors for the game and the inverse reinforcement learning algorithm to extract the cost function that is used by the subject and analyze how it differs from what is prescribed by the experimenter.

Model formulation

First we define the associated variables for the dual-task problem, where the states and actions in the MDP tuple (S, A, T, R, γ) is defined as,

- $S = S_1 \times S_2 \times S_3 \times S_4$ where
  - $S_1 = \{0, 1, ..., N_1\}$: distance of the obstacle on the left
  - $S_2 = \{0, 1, ..., N_2\}$: distance of the obstacle on the right
  - $S_3 = \{0, 1\}$: position of the target (left or right)
  - $S_4 = \{-1, 0, ..., N_4\}$ is the state of task 2.
    - (-1) means task 2 is completed
    - (0) means task 2 is cued but unattended
    - (≥ 1) is the total amount of time spent attending task 2
- $A = \{1, 2, 3\}$ is a set of three actions, shifting the position of the target (left or right), or typing on the number pad (indicating attention on task 2)

The game is setup so that the obstacles would fall down at constant speeds and did not switch positions once it appeared. The probability of new obstacles appearing is taken into account in the transition probabilities between S_1 and S_2. The actions of the participants carries no uncertainties, therefore S_3 is deterministic. Task 2 is cued at random intervals, and once the participant starts typing on the number pad, this is taken as attention shifting to task 2, therefore S_4 transitions from 0 to 1 deterministically when $A = 3$ is applied. The amount of time it takes to complete task two is up to the skills and reaction times of the participant, and this is uncertain. Therefore the transition between $S_4 \geq 1$ is probabilistic and the probabilities are determined from the trials.

The reward/cost function is posed as described to match the instructions given to the participants. For example, colliding with an obstacle is equivalent to incurring a cost of $R(s_1 = 0, s_3 = 0) = J_1$ or $R(s_2 = 0, s_3 = 1) = J_1$, and not completing task 2 is equivalent to $R(s_4 \geq 0) = J_2$.

The subject's goal is to choose the control input, whether to pay attention on the primary task of avoiding the obstacle by shifting the position of the target left or right, or to switch attention to task 2 in order to minimize the expected discounted cost given the state transition probabilities learned over time.

Simulation Results and Analysis

To show the results of the MDP model, value iteration from (2.12) is used to find the value function of the discrete states, which is used in finding the optimal action given the current observed state as formulated by (2.13). States presented to the subject when playing the game is used as inputs to the computed policy to simulate the optimal action. The sequence of actions played by the participant is compared to the actions predicted by the optimal policy given identical states. The cost function found through inverse reinforcement learning is also compared to the cost function verbally given to the participants.

Analyzing the computed value function: The value function $V(s)$ obtained through value iteration is shown in Figure 4.21 and Figure 4.22. On the x and y axis of the plot are s_1 and s_2, (distances to the obstacles). Only $s_3 = 0$ (left) is considered since $s_3 = 1$ (right) would yield similar plots with the coordinates flipped. Figure 4.21 plots $V(s)$ for $s_4 = 0$ (task 2 cued but unattended). $V(s_4 = 0)$ is always larger than $V(s_4 = -1)$ since there is no penalty if task 2 is not cued, and $\|V(s_4 = -1) - V(s_4 = 0)\|_\infty \leq 4.55$. The diagonal pattern in the plot makes intuitive sense, since this is indicating that if the distances to the closest obstacles is the same on both sides than the target has to hit one of them eventually. The high value where $s_1 = 0$ is the cost of obstacle collision.

Figure 4.22 plots $V(s_4 = 2)$. We can see the value function is high for $s_1 = 5$, and becomes increasingly higher as s_1 gets smaller. This is showing that since the task 2 completion can require up to 5 time steps, during which no attention is paid on the avoiding task, there is a chance that the target runs into the obstacle if the participant decides to shift attention to task 2 at $s_1 = 5$. As will be explained later, the participant's policies is actually even more conservative than this.

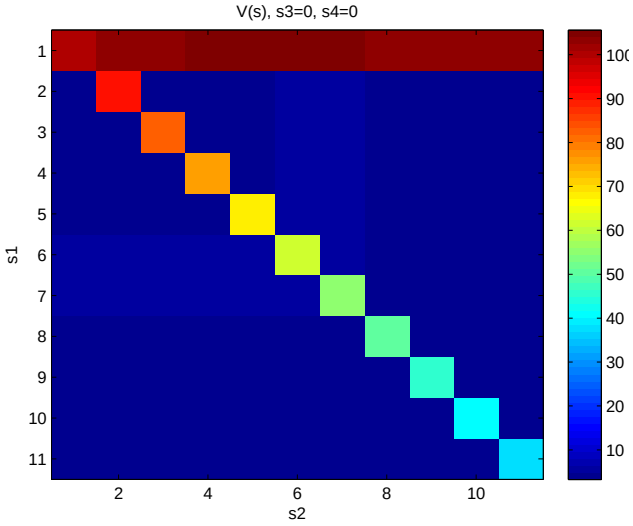


Figure 4.21: Value function of the dual-task game for s_1 and s_2 with $s_3 = 0$ and $s_4 = 0$.

Comparing the simulated optimal action with subject's action: Shown in Figure 4.23, the plots show a snippet from 100 consecutive samples taken from a single game play by the subject, counting down from the top most plot, each subplot displays the following features:

- s_1, distance to obstacle on the left
- s_2, distance to obstacle on the right
- s_3, current position of the target

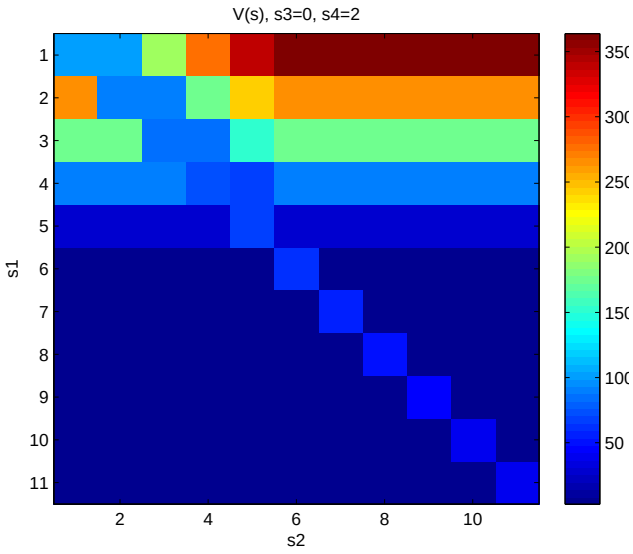


Figure 4.22: Value function of the dual-task game for s_1 and s_2 with $s_3 = 0$ and $s_4 = 2$.

- s_4, completion status of task 2
- onset of task 2 message cue
- input applied by the user based on the key pressed
- optimal input chosen from the optimal policy $\pi^*(s)$ from (2.13) of the original MDP formulation
- optimal input chosen from $\pi^*(s)$ of the MDP with modified reward function
- optimal input from $\pi^*(s)$ of the MDP with modified reward function and modified transition probabilities (discussed later)

As there weren't any additional sensors monitoring the state of the participant, such as a camera, the only way of determining which task the subject was paying attention to was through the type of key he/she pressed. Pressing the left/right keys are considered as full attention on task 1 and pressing on the number pad, enter or backspace keys are considered as paying attention to task 2. The drawback associated with this can be seen at $t = \{5, 53, 65, 91\}$, at which the optimal input acted faster than the participant's recorded input. There is a delay for the subject due to his/her reaction time to the onset of task 2 cue, as well as the time needed to first read the string of number before typing on the keypad. These delays in reaction times are not measured and therefore not modeled in the MDP. Eye-tracking equipment could potentially be used to accurately determine when exactly the subject is shifting his/her attention, as well as the delays associated with this attention switch and keyboard input.

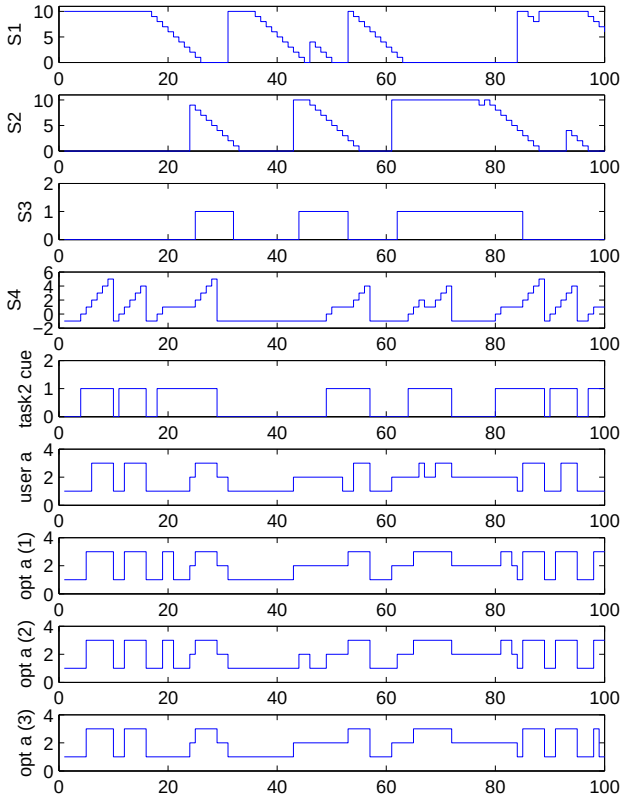


Figure 4.23: 100 sample (t) snippet of a game play.

The time it takes for the subject to complete task 2, which consists of sequences of six numbers, are computed (under the previous condition). The transition probability associated with $S_4 = 1-5$ time steps, $\{P(S_4 = i)\}_{i=1}^{5} = \{0, 0.097, 0.484, 0.387, 0.032\}$. This shows that most of the time the subject is able to complete the task under five time steps.

Despite the statistics of task completion times, the subject's policy still seems a lot more conservative. We can see at $t = \{19, 50, 81\}$, the distance to the obstacles are $\{7, 5, 7\}$. Given that the obstacles are moving at one unit per time step, and given the subject's history of task completion under five seconds, there is enough time for the subject to finish task 2 without hitting the obstacles. However, the subject's actions differ from the optimal policy. We can see at $t = \{25, 54, 85\}$, the subject first waits for a big enough clearance (≥ 8) before switching to task 2. If this is due to a conservative estimation of the task completion time, then the transition probabilities of the MDP would be different.

Also notice that the times when the subject decides to switch lanes. Notice the switching that occurs at $t = \{43, 61\}$, and compare with the optimal input in (i). The optimal policy decides to switch later at $t = \{44, 62\}$, one time step later when the obstacle is right in front of the target. This shows that the subject's policy takes into account the distance to the obstacles and not just based on the occurrence of a collision.

The bottom two subplots shows the optimal policy from a modified MDP. An additional reward feature is added to include the distance to the obstacles. We can see in subplot (g) that the optimal policy of the modified reward function matches the switching times of the subject.

The transition probabilities between s_4 is shifted to account for the conservativeness of task completion estimates. The probability associated with $S_4 = 1 - 8$ time steps is now, $\{P(S_4 = i)\}_{i=1}^{8} = \{0, 0, 0, 0, 0.097, 0.484, 0.387, 0.032\}$. In other words, extra three time steps is added on top of the subject's actual performance. Comparing the optimal policy from the new MDP (modified reward and transitional probabilities) with the original, we can see that there is no longer false alarm predictions, and matches the user's policies better.

One interesting question is what would happen if the goal of the game is phrased in a rewarding context. For example, if the participant was told that the blue dots were points instead of obstacles. Prospect theory states that we tend to be more risk-aversive when dealing with big losses [126], which means subject might no longer act as conservatively as the original problem.

Comparison with user reward function Using the inverse reinforcement learning algorithm described in Section 2.3.2, the reward function of the user is extracted. The features $\phi(s)$ used include:

- $\phi_1 =$ distance to the obstacle in current lane
- $\phi_2 =$ distance to the obstacle in the other lane
- $\phi_3 =$ occurrence of a collision
- $\phi_4 =$ completion of task 2
- $\phi_5 =$ amount of time spent in task 2

Solving problem (2.17), we obtain the features weighting vector $w = [0.123, -0.123, -0.508, 13.11, -0.333]$. $w(1)$ and $w(2)$ shows that the participant adopts the policy of trying to maximize the distance to the obstacle in the current lane and tries to minimize the distance to the obstacle in the other lane. Comparing $w(3)$ and $w(4)$, the completion of task 2 carries a higher weighting than the occurrence of a collision because the participant relies on the distance information to avoid obstacles.

Using the obtained w, condition (2.16) is checked to see which states/policies are not satisfied. Out of 500 samples, 2.4% of the states taken from the data did not satisfy the constraint, which was expected since the participant's actions are uncertain. In addition, there is a learning curve of the participant, during which his/her policy may change over time.

Splitting the data from game play into two parts, the features weighting vector becomes, $w = [0.222, -0.222, -0.111, 12.245, -0.229]$ and $w = [0.099, -0.099, -0.604, 12.245, -0.229]$ respectively. We can see that more weight is put on the obstacle distance in the beginning of the game and the weight shifts more towards the occurrence of collision in the latter part.

The participant was told that he/she would only incur penalty when there is a collision. There is no penalty involved with being *as close to the obstacles as you can without hitting it*, therefore the feature weighting vector of the actual cost is zero for $w(1)$ and $w(2)$.

Concluding remarks

We have used an optimization-based approach to model the discrete decision making of shifting attention between the primary and the secondary task in the dual-task game. Using the same framework as [37] which used MDP to simulate the policies associated with highway driving and lane switching. We extended this to study how subjects deal with distractions framed in the context of a dual-task problem. By comparing the predicted actions of the model with the subject's actions, we were able to better understand the underlying structure of the cost functions involved in this context.

We observed that the original cost associated with hitting the obstacles is not enough to capture the subject's policy. Instead, subjects used a more conservative policy where a distance buffer is added to the obstacles, or the distance to obstacles actually appears in the cost function, like the approach used in the autonomous car in [67]. This could potentially be extended to modeling how drivers respond to distractions on the highway, such as texting while driving.

Through inverse reinforcement learning, we were able to find the weights of the feature vector in the cost function. Partitioning the recorded data into the *training* and *trained* groups, we were able to observe the subject adapting the weights as he/she became more familiar with the problem. We will now scale this game-based application to real-world driving and study the context of distracted driving using the same modeling framework.

4.2.4 MDP model for attention allocation in distracted driving

Scaling up the dual-task game from the virtual world to the real-world, we can draw parallels between the previously described experiment and modeling framework with distracted highway driving. During highway driving, the primary task of the driver is to pay attention on the road, keep the vehicle to the center of the lane and avoid collisions with any incoming vehicles, which is similar to the obstacle avoidance task in the game. The secondary task of highway driving is to pay attention to the phone and text while driving. When the driver shifts his attention to the secondary task, he no longer receives visual feedback about the current states of the vehicle, and therefore has to rely on past experiences to make judgments about safety.

Model Formulation: Using the same MDP framework that is used for the dual-task game, the variables in the MDP tuple for highway texting is defined as follows,

- $S = S_1 \times S_2 \times S_3$ where
  - $S_1 = \{0, 1, ..., N_1\}$: discretized distance to the front vehicle
  - $S_2 = \{0, 1, ..., N_2\}$: discretized deviation from lane center
  - $S_3 = \{0, 1, ..., N_3\}$: discretized longitudinal velocity of vehicle
  - $S_4 = \{-1, 0, ..., N_3\}$ is the state of task 2.
    - (-1) means task 2 is completed
    - (0) means task 2 is cued but unattended
    - (≥ 1) is the total amount of time spent attending task 2
- $A = \{1, 2\}$ is a set of two actions, shifting the attention between driving and texting

Interpreting the associated variables and functions: Similar to the dual-task game, the reward function would capture a tradeoff between completion of the texting task and the penalty associated with deviation from the center of the lane or collision with the front vehicle. The transition function which changes the states would depend on whether the driver is paying attention on the road or on the phone. If the driver is paying attention on the road, then he is constantly correcting the vehicle through visual feedback. Therefore, it is less likely that the vehicle will be deviation from the lane or approach the front vehicle at an unsafe distance. On the contrary, if the driver is spending time on the secondary task, then he will not be able to correct the vehicle in real-time, and therefore the vehicle will have a higher uncertainty associated with future states in the lateral and longitudinal direction which affects S_1 and S_2. In addition, both S_1 and S_2 will depend on the velocity S_3 of the vehicle. If the vehicle is traveling at higher speeds, then S_1 will change at a higher rate, and S_2 will change with a higher probability.

Structure of policy map: The policy map of the driver, which determines whether he/she pays attention on the road or the phone, will be a value function learned from experience, similar to the value function in the dual-task game, and will be dependent on the vehicle's velocity S_3. Higher values of S_3 will be associated with lower values in the value function since the probability of incurring penalties from lane deviations and frontal collisions will be higher, therefore the driver will more likely to revert attention back to the road after a certain time has passed.

Collection and interpretation of driving data: We asked the driver to drive on the highway inside the CPG test ground with no other vehicles, i.e. S_1 can be discounted, but the driver is still required to stay within lane boundaries. As described in Section 3.1.6, in addition to the sensor measurements on the states of the vehicle, eye-tracking glasses is equipped on the driver to accurately determine when the driver switches attention between the road and the phone, as well as the amount of time spent in each. Figure 4.24 shows two screenshots to illustrate the difference.

Parsing the eye-tracking data and combining with the data from vehicle sensors, Figure 4.25 illustrates the relationship between the time spent on the phone with the speeds of the vehicle. We

Figure 4.24: Measuring attention allocation in the distracted driving scenario through gaze fixations: (left) attention on the road and (right) attention on the phone.

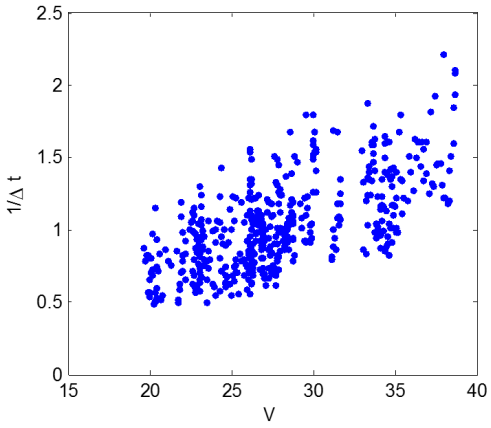


Figure 4.25: Inverse of texting duration vs velocity of vehicle.

can see that roughly when the speeds increase, the amount of time decreases, which aligns with the proposed framework that the state transition matrix is a function of the vehicle velocity.

Concluding remarks: Extending from the same framework used in the dual-task game, we presented how the game-based application can be used to parallel real-world driving. The control strategy of the driver is posed as the attention allocation between the driving task and texting. The proposed MDP framework with vehicle velocity as one of the states will determine the driver's control strategy. We equipped the driver with eye-tracking glasses to measure the duration of the time spent on the phone. The results showed that as the vehicle velocity increases, the duration time decreases, which aligns with the policy from the MDP framework where the transition probabilities and reward functions depend heavily on the vehicle velocity.

4.2.5 Mental representations of system dynamics and making forecasts

Up to this point we have presented two approaches which can be used to model the decision making in various applications involving the human. Direct methods are suitable at modeling lower-level decisions that directly maps input to output, whereas optimization-based methods involves making predictions of states and finding the best control decisions in order to minimize some cost function and satisfy constraints. One of the major factors differentiating the two approaches is the underlying assumption that in optimization-based methods, the decision maker has some internal representations of the systems involved and that state predictions are performed prior to execution of the final decisions.

Similar to the argument connecting probabilistic to connectionist methods [43], although it is unclear how mental representations and predictions are physically implemented in the cerebellum, the higher abstraction in the modeling hierarchy allows greater flexibility for exploring a broader range of human behaviors, from game-based applications typically associated with psychology experiments to real-world applications for human-machine interactions.

In this section, we will discuss the potential evidences which support the claim that human decision makers do possess internal representations of the real-world and that state predictions are simulated. We attempt to validate this by collecting eye-tracking data from subjects in various experiments which involve both driving and juggling, and therefore further strengthening the parametric approach of modeling a subset of human behaviors with the optimization-based approach.

Evidence from literature

Mental models or representations of the world and their uses in the decision maker's control strategies has been discussed in many reaching experiments which involve sensorimotor control of the arm [63, 73]. In [60], this mental representation is termed the forward model and postulates that multiple pairs of inverse (controller) and forward (predictor) models are responsible for sensorimotor learning and control. Evidence from fast reaching movements suggest that forward models are responsible for the speed of these movements [59], and that the amount of delay in the sensory feedback is too slow to perform fast movements based on the feedback model alone.

In addition, the conjecture that humans perform mental simulations in the cerebellum has also been investigated in both the neuroscience and psychology literature. The existence of mirror neurons in the premotor cortex show that subjects will simulate a motor skill familiar to the subject while observing another subject performing the same skill [56]. [61] attempts to make neural connections between the mirror neurons and the forward models. Psychology experiments aims to support the theory of mental simulations from a top-down approach. The model presented in [109] predicts that people would use more samples of mental simulations when it is harder to make an accurate prediction due to higher simulation uncertainty. In [127], experiments showed that people continuously update their physical simulations and predictions in light of new information. Experiments in [128], where subjects were asked to make predictions about the expected range of a ball, suggests that people form a probabilistic distribution over the predictions.

In light of the above discussed studies, we will attempt to further support the theory of mental

predictions through the use of eye-tracking information collected from subjects performing everyday activities which involve dynamic decisions. We will look at two scenarios in particular, one in the context of driving, and the other in the dynamic task of juggling.

Using eye-tracking information to support forward models in driving

The review of eye-tracking in everyday tasks showed that eye movements are used to locate information needed by the motor system and eyes typically seek out information prior to decision making [129]. In the context of modeling the steering control of drivers, [121] proposed a two-level model where in one level is the forward model and executes steering commands in open loop, and the other level will provide corrective steering based on feedback. We equipped drivers with eye-tracking glasses and recorded their gaze-movement when driving on the winding track at CPG discussed in Section 3.1.6, which involves many turns and curves unlike a typical highway.

Figure 4.26: Eye-tracking showing far point (left) and near point (right) gaze points in driving.

Figure 4.26 shows two different screenshots taken from the eye-tracking video that was recorded during the driving experiment on the winding track. We can see that the left picture shows the driver looking at a far point down the road around a curve. This supports that the driver is obtaining visual information about the road i.e. road curvature in order to feed the forward model with parameters. The picture on the right shows a jump from the far point to the near point, illustrating that after obtaining the necessary information for the forward model, the driver has now switched back to using the visual feedback to facilitate in corrective steering ensuring the vehicle remains within lane boundaries.

Eye-tracking of objects in juggling

Juggling is one example of an activity involving hand-eye coordination and requires a lot of practice to master the skill. We will use the context of juggling to relate how familiarity with the activity will affect eye-tracking, and in turn show that expert jugglers are better at making forecasts than novice juggler.

The idea that people who are experts in a particular skill will be better at making predictions is explored in [130], where the gaze location of expert and novice billiard players were tracked while watching videos of billiard shots. Eye-tracking information revealed that expert players tend to saccade between critical points of contact, whereas novice players tend to track the ball trajectories, which showed that expert players had better predictions about the ball's expected path. Similarly, [112] compares the visuomotor strategies between novice and expert jugglers, where novices look at the balls around their zeniths and experts tend to fixate their gaze at a central location within the juggling pattern.

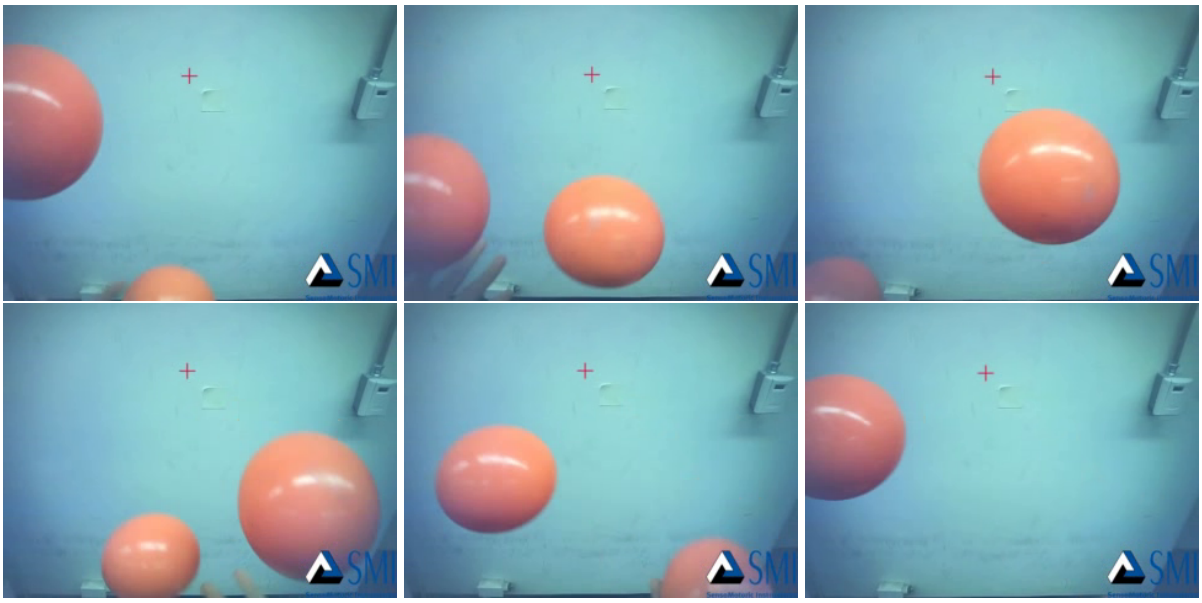


Figure 4.27: Eye-tracking showing non-tracking (using peripheral vision) of objects during juggling with same objects.

We also equipped an expert juggler with eye-tracking glasses and recorded his eye gaze movement under different juggling scenarios. Figure 4.27 and 4.28 shows sequential screenshots (from left to right) taken from two different juggling tasks. In Figure 4.27, we asked the juggler to perform a three-ball cascade using the same type of balls. Shown in the picture, the fixation is fixed in the central location, similar to [112]. In Figure 4.28, we asked the juggler to also perform the three-ball cascade pattern, but this time using three objects of different weights, shapes and sizes: a ball pin, a light large ball, and a smaller heavier ball. As shown in the picture, the visuomotor strategy no longer followed the one for juggling with identical objects. Instead, the juggler's gaze shifts from the expected apex of the objects' trajectories. We also observed that the gaze shifts to the expected apex at a time interval prior to the object's arrival.

These observations and the results in [112] shows that the juggler are making predictions and forecast about the trajectories of the objects. In the scenario where the expert juggler is juggling with identical and familiar objects, he is so familiar with the task that accurate predictions can be

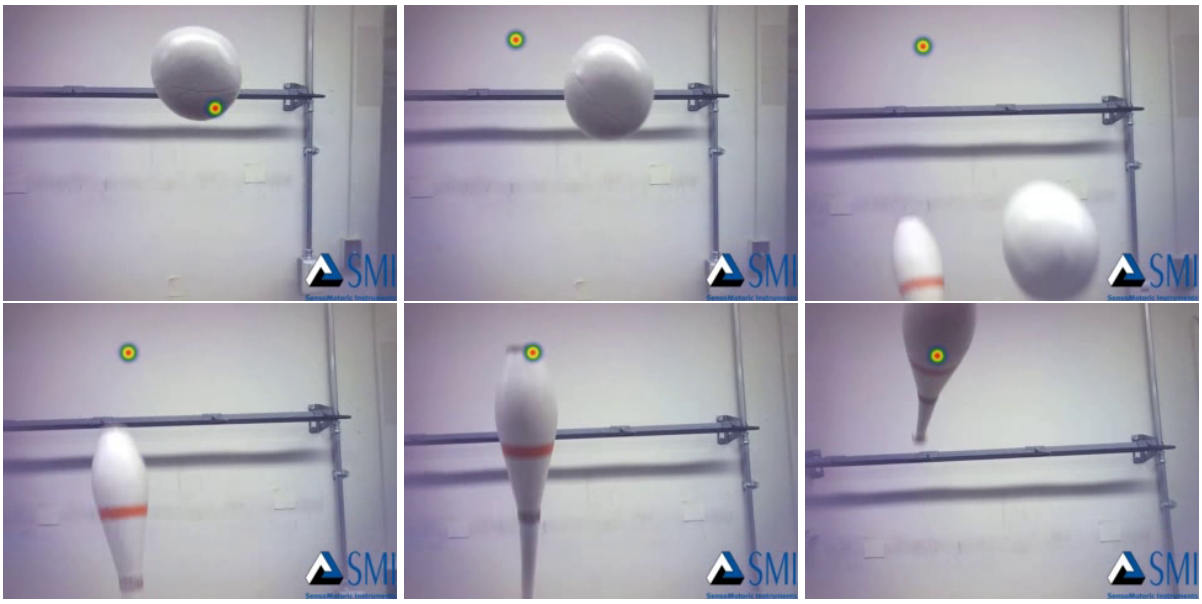


Figure 4.28: Eye-tracking showing prediction and tracking of objects with point-of-focus during juggling with different objects.

made with just the peripheral visions of the juggler. On the other hand, for novice jugglers, or in the scenario where the expert juggler has to deal with three different objects, the predictions are not as accurate, and therefore shifting the gaze to the expected apex of the trajectory is required in order to obtain feedback for corrective action.

This control strategy is similar to the forward/feedback modeling pairs discussed in both driving [121] and sensorimotor control in reaching experiments [60], where the juggler makes forward model forecasts about the expected trajectory of the objects, shifts gaze to the expected location to receive accurate sensory feedback, and then use this feedback information to update state estimates of the objects, and finally adjusts the locations of the hand in order to successfully catch the ball.

Concluding remarks

Mirror neurons in the prefrontal cortex, results from sensorimotor control, and experiments for probabilistic mental simulations all suggest that humans, in addition to feedback control, also possess forward models and makes predictions to facilitate in the decision making of many everyday tasks. Eye-tracking data from driving on the winding track and juggling also supports this modeling regime. This provides us with the core elements of the optimization-based methods in decision making.

4.3 Conclusions

In this chapter we have introduced two approaches to modeling human decision making: direct methods and optimization-based methods. The first approach of direct methods maps observed input to the control decisions directly without extra computations involved with solving for an optimal solution. This methodology is suitable for mapping lower-level decisions which are repetitive and requires no planning. We used examples from driving on the highway, extreme driving on slippery roads, as well as game-based experiments involving kinematic dynamics. We conclude that direct methods are only useful for a subset of the task scenarios that is discussed, and fails to capture the human decision making involved in the entire task.

For example, in the highway driving example, direct methods are good at modeling the lane keeping component of the task, but fails to model the lane switching portions which is postulated to require more planning due to safety assessments. Similarly, in the projectile game, we have seen a pattern where partitioned groups of data showed very different results from parameter identification. For example, MSE values in the new scenarios are much higher than in the old scenarios. In addition, the identified weights on the feature vectors showed new scenarios relied more heavily on scenario-based features, whereas in old scenarios it was dominated by feedback-based features.

We conclude that in the same task, people will switch between different methods of decision making depending on the context. Therefore a hybrid approach integrating different modeling methodologies needs to be adopted to encompass the broad range of behaviors. For example, in highway driving, we can propose a two-level approach similar to [67] where at the higher-level an optimization-based method is used for path planning to avoid collisions with other cars, and at the lower-level tracks the generated reference using direct methods. A similar hybrid framework can also be applied to the projectile game, where subjects switch between the optimization-based method and the direct method.

Since the computational complexity of directly methods is significantly less than the complexity of mental simulations, subject are more likely to adopt the direct approach when the errors are small enough such that the linear maps provide a good approximation to the nonlinear problem. On the other hand, if the subject is presented with a new trial or scenario which is significantly different from the previous, then the error values will not be a good approximation. In this case, the subject will make more careful calculation and estimations based on past knowledge of the input mechanism and the shape of the expected flight trajectories, optimizing over all possible flight paths given their own learned knowledge of the experiment.

In the second part of this chapter, we discussed the implications of using optimization-based methods to model decision making, where subjects are postulated to perform mental simulations and state predictions in order to satisfy constraints and minimize the cost function. Results from the projectile example showed that in order to minimize the amount of uncertainty in the state predictions, subjects are minimizing over the length of the trajectory, as well as the change in control inputs. Similar to the above discussed hybrid approach, we saw the weights on the cost function vary as the context of the problem changes.

Extending the game-based experiments to real-world driving, we modelled the continuous steering input involved in extreme driving maneuvers using MPC and postulated that drivers are

trying to follow a learned state reference trajectory corresponding to the maneuver, and that drivers need to have knowledge of the nonlinearity in the tires in order to achieve this.

In addition to the continuous decisions, we also used the same optimization-based modeling framework to model discrete decision making in a dual-task game, where the subject was required to make decisions on attention allocation between the primary task and the secondary task. We observe, as seen in the shift in the weights in the cost function, that as the subjects become more familiar with the transition probabilities associated with the game, they become more comfortable in switching to the secondary task more often and at higher risks.

Similarly, the same modeling framework is applied to distracted driving on the highway, where the subject's attention allocation between driving and texting is directly measured using eye-tracking glasses. Following the MDP framework, the value function showed dependence on the vehicle's longitudinal velocity and complies with the measured attention durations on the texting task, which shows a decrease in duration as velocity increases.

We drew from literature and data from eye-tracking in driving and juggling that subjects possess forward models and perform predictions in their decision making, a necessary component in the optimization-based methods.

Lastly, with better understanding of how humans make decisions under different circumstances, we can design better intelligent systems that integrates human behaviors into the algorithms. We will discuss how this can be achieved in the next chapter.

Chapter 5

Human Model Integration with Control Applications

As mentioned in Chapter 1 section, an increasing number of intelligent systems are now designed to not only work autonomously on a task, but to operate for applications which either involve servicing or working in cooperation with human subjects, i.e. collaborative robotics [5] or autonomous cars. Therefore it is important to have an understanding and knowledge base of the human's expected behaviors in specific scenarios that the intelligent system caters to. Incorporating predicted behaviors of the human subject into the control algorithm of these systems will lead to more efficient collaborations, as well as greater level of comfort and acceptance by the human in many autonomous systems. Following the previously discussed motivation of relating human behavior with computational models, in this chapter we will present the second major contribution of this thesis, which is to introduce a unifying framework for integrating human decision making policies into controller design and address the interest of modeling human behaviors for the purpose of improving human-machine intelligent systems.

In this chapter, we will first propose the mathematical formulation for the control framework which can incorporate the expected human behavior. We approach this by extending the MPC framework introduced in Section 2.3.1 with additional constraints which correspond to the predictions of human behavior, including additional terms in the cost function in order to minimize controller intervention, and discussing the differences between open-loop and closed-loop formulations. In the interest of generating these predictions, we revisit the two different means of modeling for human decision making: the parametric approach and the data-driven approach.

Parametric approaches are used for their capability to directly relate parameters to observations from the behavioral perspective. However, for some formulations such as the optimization-based methods, the parameters are much more difficult to identify when compared to the data-driven approach. For the data-driven approach, there exist many state-of-the-art identification algorithms and even online recursive algorithms that make models adaptive in real-time [77].

To illustrate the aforementioned system structure with real-world applications, we will take examples from driver modeling for autonomous vehicles. Using the direct methods of driver models discussed in Section 4.1, we present the results for the extended MPC framework with closed-loop

driver predictions. The last part of this chapter will discuss the inclusion of the cluster-based, data-driven driver model in a distracted scenario. We emphasis that the extended MPC framework is a natural candidate in the interest of improving human-machine interactions due to its optimal and predictive nature, allowing the minimization of control intervention while still satisfying safety constraints.

5.1 Control Framework for Semi-autonomous systems

Integrating the human component into the design of human-robot intelligent systems, or semi-autonomous systems, involves many factors ranging from the physical design of the system, to the software implementations of control algorithms. A review of these factors discussed in [131] mostly focused on the higher-level concepts which involved inferring the intent of the human from explicit and implicit communications. Action planning and joint actions from the inferred intent are also discussed in the survey, but is limited to the higher-level decision making as well.

As mentioned previously, cognitive architectures [38] like ACT-R are very good candidates for integrating the different factors of human cognition derived from psychology and neuroscience into one unifying framework. The unified structure provides a good basis for simulating human behaviors across a wide range of spectrum from simple arithmetic to everyday driving [39]. However, unlike computational models, this more descriptive approach to modeling makes integration with typical control algorithms difficult.

In this section, we will present the extended MPC as a unifying framework that can incorporate computational human models, such as those discussed in Chapter 4, directly into the control algorithm of the semi-autonomous system. We want to emphasis that in this framework, human models are integrated as part of the predictive algorithm in the MPC, and therefore can handle both the higher-level and lower-level representations of control.

Model formulation

We are interested in generating a sequence of predicted control decisions from the human model given observations, and utilize a control framework which can make use of this predicted information. Readily used as a framework for control systems such as obstacle avoidance in autonomous driving [67], the way MPCs are posed with state predictions make them a natural candidate for such purpose. Recall Equation (2.10b) of the MPC formulation from Section 2.3.1 describes a model of the future state evolutions as equality constraints. We can add an additional equality constraints to handle the predicted human input u_{pred} and its relations to input variables u_{mpc} posed as the optimization vector to the MPC.

Reformulating formulation (2.10) to include the human model,

$$\begin{aligned}
\min_{U_t} \quad & J_N(\bar{\xi}_t, U_t, \Delta U_t) + J_N(u_{k,\text{mpc}}) && \text{(5.1a)}\\
\text{subj. to} \quad & \xi_{k+1,t} = f_{\text{sys}}(\xi_{k,t}, u_{k,\text{sys}}) \quad k = t,...,t+H_p-1 && \text{(5.1b)}\\
& u_{k,t,\text{sys}} = f_{\text{u}}(u_{k,t,\text{mpc}}, u_{k,t,\text{pred}}) \quad k = t,...,t+H_p-1 && \text{(5.1c)}\\
& u_{k,t,\text{pred}} = f_{\text{pred}}(\xi_{t,t}) \quad k = t,...,t+H_p-1 && \text{(5.1d)}\\
& \Delta u_{k+1,t} = u_{k+1} - u_k \quad k = t,...,t+H_p-2 && \text{(5.1e)}\\
& u_{k,t} \in \mathcal{U} \quad k = t,...,t+H_p-1 && \text{(5.1f)}\\
& \Delta u_{k,t} \in \Delta\mathcal{U} \quad k = t+1,...,t+H_p-1 && \text{(5.1g)}\\
& \xi_{t,t} = \xi(t) && \text{(5.1h)}\\
& \xi_{N,t} \in \mathcal{X}_f && \text{(5.1i)}
\end{aligned}$$

The reformulated MPC has two additional constraints (5.1c) and (5.1d). Note in the original formulation (2.10), the variable u is both the optimization variable of the MPC and the direct input for differential Equation (5.1b). In the new formulation (5.1), the input $u_{t,\text{sys}}$ for the system differential Equation (5.1b), is now defined as a function of the optimization variable $u_{t,\text{mpc}}$, and the predicted behavior of the human subject $u_{t,\text{pred}}$, as shown in Constraint (5.1d). Simple affine maps such as,

$$u_{t,\text{sys}} = u_{t,\text{mpc}} + u_{t,\text{pred}} \tag{5.2}$$

combined with the additional term in the cost functions (5.1a) which minimizes $u_{t,\text{mpc}}$ will result in a solution which tries to minimize the controller intervention and apply $u_{t,\text{sys}}$ as close to $u_{t,\text{pred}}$ as possible.

The second constraint (5.1d), relates to the manner which the human behavior predictions are performed. There are two ways this can be performed in the MPC framework: open-loop or closed-loop predictions.

Open-loop predictions of human behavior

Open-loop predictions in the MPC framework refers to how the human behaviors are predicted based on the current observed states of the system. A sequence of predicted control inputs for the next prediction horizon is generated prior to running the MPC solver. Therefore, while the solver searches for the optimal solution, the predicted sequence remains constant and unaffected by perturbations of states within the MPC. This approach is less computationally intensive and therefore is more suitable for more complex models of human behavior.

Closed-loop predictions of human behavior

Closed-loop prediction, on the other hand, involves predictions which are not only based on the currently observed state feedback but also future state evolutions within the MPC framework. Instead of generating the sequence beforehand, a model, or a state feedback law, of how the predicted

behavior will be generated is included in the MPC framework directly. Through this approach, future states within the MPC framework will affect the value of $u_{t,\text{pred}}$. Evidently, the introduction of this equality constraint will increase the computational complexity of the MPC problem more than the open-loop approach, but the effects of state evolutions in the MPC can be coupled with the human model internally.

We will now discuss in detail applications which uses the MPC framework to include behavior predictions. In particular, we will investigate how the driver models can be applied for intelligent control in semi-autonomous vehicles, both from the closed-loop and the open-loop approaches.

5.2 Integrating driver models into semi-autonomous driving applications

Applications discussed in this section will involve the same autonomous driving framework and cover examples of both closed-loop and open-loop predictions of the driver behavior. The MPC framework, as well as numerous other autonomous driving algorithms from the DARPA grand challenge [8, 9, 11], the Google car [132], and assistance modules from the automotive industry [7], can be implemented by itself without driver prediction. Even for obstacle avoidance, a nonlinear solver can be used to find a solution to an optimal control problem where the obstacle is represented as either a cost term or a constraint [67]. But having a control algorithm which learns and operates to the preferences of specific drivers [34, 35] will improve driver's comfort and acceptance for the system.

To differentiate from purely autonomous vehicles which ignores control actions from the driver, our semi-autonomous framework will focus on letting the driver behavior dominate the control algorithm, and only apply the required intervention when the driver is predicted to act in an unsafe manner, i.e. collision with other cars. In this way, the control inputs and state trajectories of the car will not only feel more naturalistic to the driver himself, but also to the other drivers in neighboring cars. In fact, it was reported that the accidents that were involved in the Google car were primarily associated with the unnaturalistic driving performed by the autonomous controller [133] i.e. driving too slow frustrated other vehicles with human driver. Therefore, it is necessary during the transition from human operated to machine operated cars to include semi-autonomous algorithms.

In addition, many of the solutions do not need to utilize the saturated regions of the tires. For maneuvers which involve the saturated regions, such as drifting or last minute obstacle avoidance on slippery surfaces, the solutions would live closer to the unstable equilibrium points of the system dynamics [134]. It would be much harder for the nonlinear solver to find such a solution. Therefore, as an example, we will incorporate the switched driver model introduced in Section 4.1.3 into the MPC with closed-loop predictions of driver behavior under extreme driving conditions that requires the utilization of the saturated regions of the tires, and show how the driver model can be facilitate in guiding the solver into finding the solution to achieve the maneuver.

In the second example we will implement a semi-autonomous framework to guarantee safety

of the vehicle. We postulate that when the driver is attentive to normal traffic condition, he will be driving in a safe manner, and only when the driver is distracted that the future trajectories will be unsafe. We monitor the state of the driver through skeletal tracking to determine the degree of distraction, combine this with states of the car to arrive at a data-driven approach to model driver behavior. The predicted driver behavior will be applied in open-loop with the same MPC framework in (5.1).

5.2.1 Closed-loop driver predictions in extreme driving maneuvers

Simple lane keeping driver models of a state feedback gain has been implemented in close-loop in a similar framework in [110] and even an uncertain version has been integrated with stochastic MPC [20]. We would like to extend this and see if we can include the switched driver models in extreme driving. To include driver models in the task of extreme driving maneuvers, we will first revise the driver model that is to be used in the semi-autonomous framework and detail the mathematical equivalence to the constraints in (5.1). Then we will present simulated results of the MPC with the driver model included.

Switched driver model for extreme driving

Using the switched driver model for driving under extreme conditions, we will integrate it into the MPC framework (5.1). Recall in Figure 4.10, we see that the switch happens only once in the middle of the maneuver, therefore to decrease the computational complexity, we will simplify the switching condition from state-dependent to a time-dependent formulation. In this first formulation, the optimization variables are the control variable $u_{k,t}$ and are only constrained by the polytopic bounds on $u_{k,t}$ and $\Delta u_{k,t}$. Here we present the time-dependent version of the two-mode switched linear model,

$$u_{k,t} = \theta^T_{\sigma(k,t)} \begin{bmatrix} r_{k,t} \\ 1 \end{bmatrix} \tag{5.3a}$$

$$\sigma(k,t) = \begin{cases} 1 & \text{if } k < T_{sw} \\ 2 & \text{if } k \geq T_{sw} \end{cases} \tag{5.3b}$$

This revised version of the switched driver model will be integrated into MPC formulation (5.1) in two different ways. The first method will use (5.3) to redefine the functional input between the optimization variable and the input applied to the system, guiding the MPC to follow a switched control strategy. The second method will take (5.3) and the parameters identified from the HIT algorithm in Section 4.1.3 to make driver control input predictions in close loop with the MPC. We present the simulation results as follows.

Guiding the controller with the switched strategy

In this section, we are interested in using the two-mode switching model structure discussed in Section 4.1.3 to guide the MPC to act more like the switched strategy of the driver. We reduce the

flexibility in the optimization variable by introducing extra constraints which will force the MPC to find solutions near the saturation regions of the tires.

Equation (5.3) is used to replace Constraint (5.1c), and reformulated with the proper nomenclatures as,

$$u_{k,t,sys} = \theta^T_{\sigma(k,t)} \begin{bmatrix} r_{k,t} \\ 1 \end{bmatrix} \tag{5.4a}$$

$$\sigma(k,t) = \begin{cases} 1 & \text{if } k < T_{sw} \\ 2 & \text{if } k \geq T_{sw} \end{cases} \tag{5.4b}$$

where the feedback gains, $u_{k,t,mpc} = (\theta_2, \theta_2)$, are the reformulated optimization variables to the MPC.

Similar to the results presented in Section 4.2.2, the new MPC formulation is solved using NPSOL with a sampling time of 50 ms and $H_p = 160$ steps. The reference trajectory comes from recorded expert driving data. Note T_{sw} could be posed as an optimization parameter, but to avoid having to solve a mixed-integer program, we fixed T_{sw} and tried a few values. For the following results, we have fixed $T_{sw} = 83$.

Figure 5.1 shows the state and steering trajectories predicted by the optimal control problem. The state trajectories verify that even with the additional state feedback constraint (5.4), the solver is still able to find a feasible solution to track the reference trajectories. In addition, the general shape of the input trajectory also conforms to the steering pattern of initializing drift and counter-steering.

Predicting driver's switched strategy in closed-loop with MPC

Contrary to the above presented framework which imposes model structure to the controller input to the system to follow the two-mode PWA approach, an alternative formulation for the semi-autonomous control system, is to minimize controller intervention. Therefore, we pose a similar MPC problem to (2.10) and reformulate Constraint (5.1c) to minimize controller intervention similar to (5.2), where $u_{t,\text{sys}} = u_{t,\text{mpc}} + u_{t,\text{pred}}$ is the control applied to the system dynamics, and $u_{t,\text{mpc}}$ is the optimization variable to the MPC which measures how much the controller intervenes. By adding this *intervention term* in the cost function like (5.1a), the nonlinear solver will find a solution which minimizes controller intervention while still tracking the desired trajectory and satisfying state and input constraints.

Figure 5.2 shows the controller intervention $\tilde{u}$ for three different MPC formulations. The solid line represents the formulation where $\tilde{u}$ is not penalized in the cost function. The dashed and dashed-dotted lines represents the formulation where $\tilde{u}$ is penalized. The dash-dotted line represents the formulation where predicted driver steering is taken from the open loop sequence, which in this case is just the recorded driver inputs from trial 5. In general, this open loop prediction could be obtained from any higher level module which computes a sequence of driver predictions. The dashed line represents the formulation where driver steering is predicted in closed-loop with the state predictions inside the optimization problem using the hybrid model identified in Section

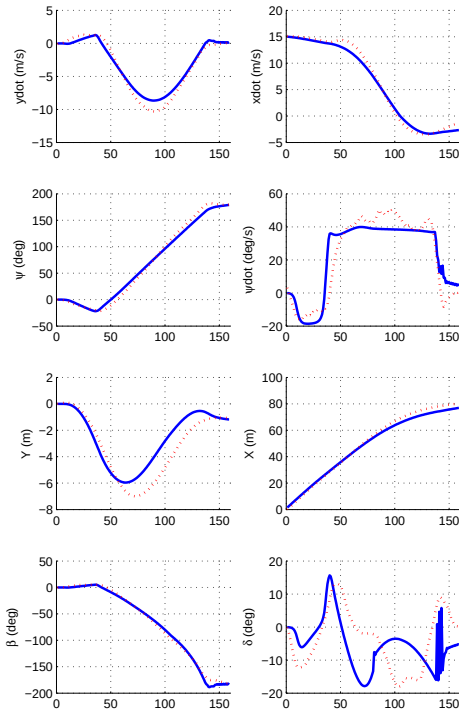


Figure 5.1: State and input trajectory: NPSOL solution of new optimization formulation (blue solid line) vs expert driver trajectories (red dotted line).

4.1.3. Figure 5.2 shows that $\tilde{u}$ for the latter two formulations are smaller. We remark that there is a tradeoff between controller intervention and tracking error and is highly dependent on the weights between the respective terms in the cost function.

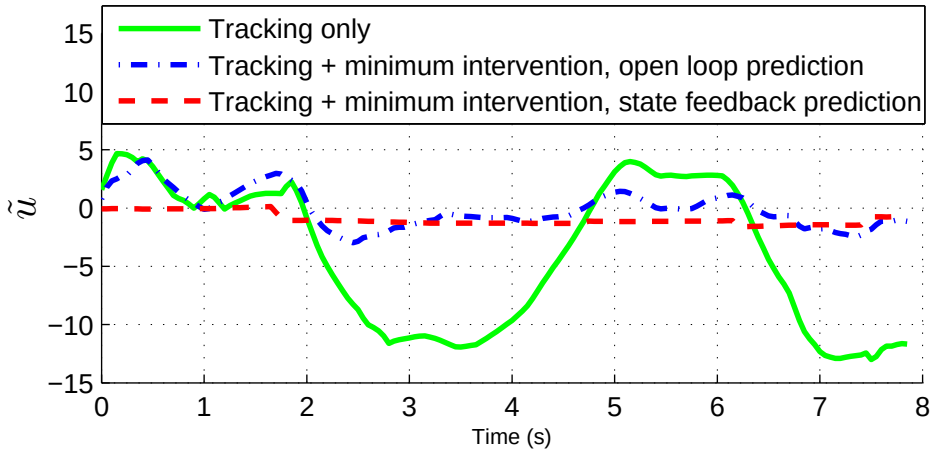


Figure 5.2: Controller intervention $\tilde{u}$: difference between controller input and predicted driver input. Predictions are open loop or through state feedback.

We have so far discussed the application of integrating driver models in closed-loop with the MPC, which was computationally possible because of the simplicity of the models involving only state feedback gains. However, for more complicated driver models, it will be too computationally extensive to integrate it in closed-loop with the MPC, making the MPC optimization problem intractable for the search algorithms. We will now discuss driver models for distracted driving, and use it in open-loop with the MPC controller.

5.2.2 Open-loop approach to integrating distracted driver models into semi-autonomous system

One major problem with the above closed-loop approach to driver-controller fusion is the increase in computational complexity due to the additional equality constraint introduced for the behavior predictions of the driver. To eliminate this problem, the open-loop approach can be used where a sequence of inputs can be generated prior to running the MPC.

For the distracted driving scenario, we are interested in making predictions of future states of the vehicle resulting from the driver's control input. To capture the effects of driver distractions on the predicted states, we need a more complex model which makes closed-loop predictions much more difficult to perform within the MPC.

We will now introduce the cluster-based driver model where predictions are performed from observed skeletal configurations of the driver, vehicle and environment states as well as past steering behavior. And present the results of fusing this cluster-based, open-loop driver model into the MPC framework of autonomous driving. We would like to credit the modeling framework, controller design and experimental design to the authors in [19]. My own contribution to this work lies in the data collection and the qualitative assessments of the semi-autonomous controller.

Cluster-based probabilistic driver model

The behavior of the distracted driving scenario on the simulator is highly variable and dependent on many factors including vehicle states and driver pose. The capability of the cluster-based model outlined in Section 2.2.2 to handle uncertainty with probability distributions, makes the model a perfect candidate to use for this particular application. Following the experimental setup for distracted driving on the simulator described in Section 3.1.6, the cluster-based framework will be used to model driving behavior and future predictions during different scenarios.

Feature vector: First we need to define the feature vectors that are used as observations O for the clusters. The feature vectors can be divided into four groups: vehicle states, environment, past input and driver states. The vehicle states correspond to those listed in Section 3.1.1 and is provided by the high-fidelity vehicle model of Carsim. The environment, including the curvature of the road, the existence of obstacles and the states of near-by vehicles, is predefined by the experimenter and experimental design, [19] provides a detailed description of the experiment scenarios and how the environment is coded for different trials. Temporal trajectories of past steering inputs are also included in the feature vector. Driver states are skeletal and joint predictions obtained from the

Microsoft Kinect. Ideally, the skeletal information would be able to distinguish between distracted and attentive states by detecting the position of the hands with respect to the wheel. Unfortunately, this may not necessarily be the case since people have different texting preferences, and having the hands away from the wheel may not always correspond to the subject texting, and vice versa, having the hands close to the wheel doesn't mean that the person isn't texting. To minimize this confusion, we ask the subject to text to the side with their hand far away from the wheel. If eye-tracking and head-tracking was involved, then the driver states would be distinguished more clearly. Nevertheless, we present the results we have with the current setup here.

Model formulation and identification: Taking large amounts of data, accumulating to 1 hour of driving on the CarSim simulator, the feature vectors or observations are used to feed the k-means clustering algorithm [82]. Instead of using steering angles, the prediction output of the clustering based model are directly mapped to future vehicle states. We fit a marginal distribution $p((X(k)-x(0))|O,I)$ to cover the set of possible future vehicle states for each identified cluster. The marginal distribution captures the probability of the difference between the future states of the vehicle $X(k)$ over the next T horizon and the current state $x(0)$ given the set of observation clusters O and the current measured observation I.

With the marginal distributions fitted for each cluster, we can compute the *vehicle prediction multifunction* (VPM) denoted by $\Delta(\alpha,k,O,I) \subset 2^{R^n}$ as the solution to,

$$\Delta(\alpha,k,O,I) = \arg\min_{\Omega\subset R^n} |\Omega| \tag{5.5a}$$

$$\text{s.t.} P((X(k)-x(0)) \subset \omega|O,I) \geq \alpha \tag{5.5b}$$

$$\forall k \in \{0,...,T\} \tag{5.5c}$$

where $\omega = \{x \in R^n | z_1 \leq x \leq z_2\}$ represents the upper and lower bounds for the future state. The VPM is the smallest size set that contains all α probable vehicle trajectories in k times steps, given prior observations and current information.

Results of cluster-based probabilistic driver model: The results shown in Figure 5.3 and 5.4 illustrates a comparison between two types of models in predictions of future vehicle states, and the probability distributions associated with the identified cluster. The first one uses a worst case scenario where state are the reachable sets (RS) treating drivers as a fixed disturbance at maximum steering range. The second one uses the VPM - the cluster-based driver model with marginal distributions. We can see that in both the curved and straight roads, with and without obstacles, future vehicle states in the lateral direction has a much larger range when predicted using the RS model.

Our VPM model of distracted driving behaviors will be a better model than the model computed with RS since in that worst case, the RS model is overly conservative and always predict the driver to be unsafe. We can see especially in the attentive states of the driver that our model predicted the driver will be safe whereas the RS model always predicted the same regardless of whether the driver was attentive or not.

Integrating distracted driver model into MPC

With the driver models established in the previous section, we can now use it to make open-loop predictions and integrate it into the MPC framework for semi-autonomous systems. First we present the modified semi-autonomous framework and then the results.

Formulation: To integrate the above introduced VPM model of distracted driving, we start from the semi-autonomous framework (5.1), introduce a slack variable for the state constraint, formally define the equalities associated with driver predictions, and reformulate the framework as,

$$\begin{aligned}
\min_{u,\varepsilon} \quad & \lambda\varepsilon + \sum_{k=0}^{\bar{N}} (\|u^c(k)\|_R + \|\Delta u^c(k)\|_S) && \text{(5.6a)}\\
\text{subj. to} \quad & \xi_{k+1,t} = f_{\text{sys}}(\xi_{k,t}, u_{k,\text{sys}}) \quad k = t,...,t+H_p-1 && \text{(5.6b)}\\
& u(k) = u^c(k) + u^{\text{driver}}(k) && \text{(5.6c)}\\
& u_{k,\text{driver}} = \Pi(\Delta(1,k,O,I)) \quad k = t,...,t+H_p-1 && \text{(5.6d)}\\
& h(x(k),u(k)) \leq 1\varepsilon && \text{(5.6e)}\\
& \Delta u_{k+1,t} = u_{k+1} - u_k \quad k = t,...,t+H_p-2 && \text{(5.6f)}\\
& u_{k,t} \in \mathcal{U} \quad k = t,...,t+H_p-1 && \text{(5.6g)}\\
& \Delta u_{k,t} \in \Delta\mathcal{U} \quad k = t+1,...,t+H_p-1 && \text{(5.6h)}\\
& \xi_{t,t} = \xi(t) && \text{(5.6i)}\\
& \xi_{N,t} \in \mathcal{X}_f && \text{(5.6j)}
\end{aligned}$$

where the slack variable ε is a new optimization variable added to Constraint (5.6e) and penalized in the cost function (5.6a). The cost function also penalizes for the controller intervention and the change in controller intervention u^c and Δu^c, combined with the predicted driver steering results in the control input u applied to the vehicle system (3.1). The driver steering in (5.6d) is predicted using the probability density obtained from the cluster-based framework described in []. The function Π is constructed to pick the expected value of $\Delta(1,k,O,I)$ for each $k \in \{0,...,\bar{N}\}$, and finds the steering angle which results in the trajectory. Notice this sequence of state trajectory and the corresponding steering angle is computed before the MPC is solved, and therefore driver predictions are utilized in the open-loop sense.

Results of semi-autonomous controller performance Subjects were asked to go through 4 training trials, and the data collected from these trials were used to run the k-means clustering algorithm in order to fit the associated driver model. As mentioned previously, Figure 5.3 and 5.4 shows the results of the prediction of the model, which can be used in an open-loop approach in the semi-autonomous framework.

After obtaining the driver model, we asked the same subject to go through another 4 testing trials where the semi-autonomous controller is applied with the integration of open-loop driver

predictions. The weights of the cost function (5.6a) used in the experiment which has the semi-autonomous controller running are $R = \begin{bmatrix} 1 & 0 \\ 0 & 0.1 \end{bmatrix}$, $S = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.01 \end{bmatrix}$, and $\lambda = 1000$. The MPC controller is solved using NPSOL [90] at a sampling rate of 200ms.

We define the occurrence controller intervention to be when $u^c \neq 0$ in (5.6c). From all the trials, we found that the controller using the VPM model intervened only 13% of the time. Compare this to the RS model where it would always intervene since the driver is always predicted to be unsafe. Figure 5.5 shows examples of instances where controller intervention occurred. The dotted red lines, solid blue line, dotted blue line and red line represented the past steering angle, future steering angle, expected steering from VPM and future controller steering angle respectively.

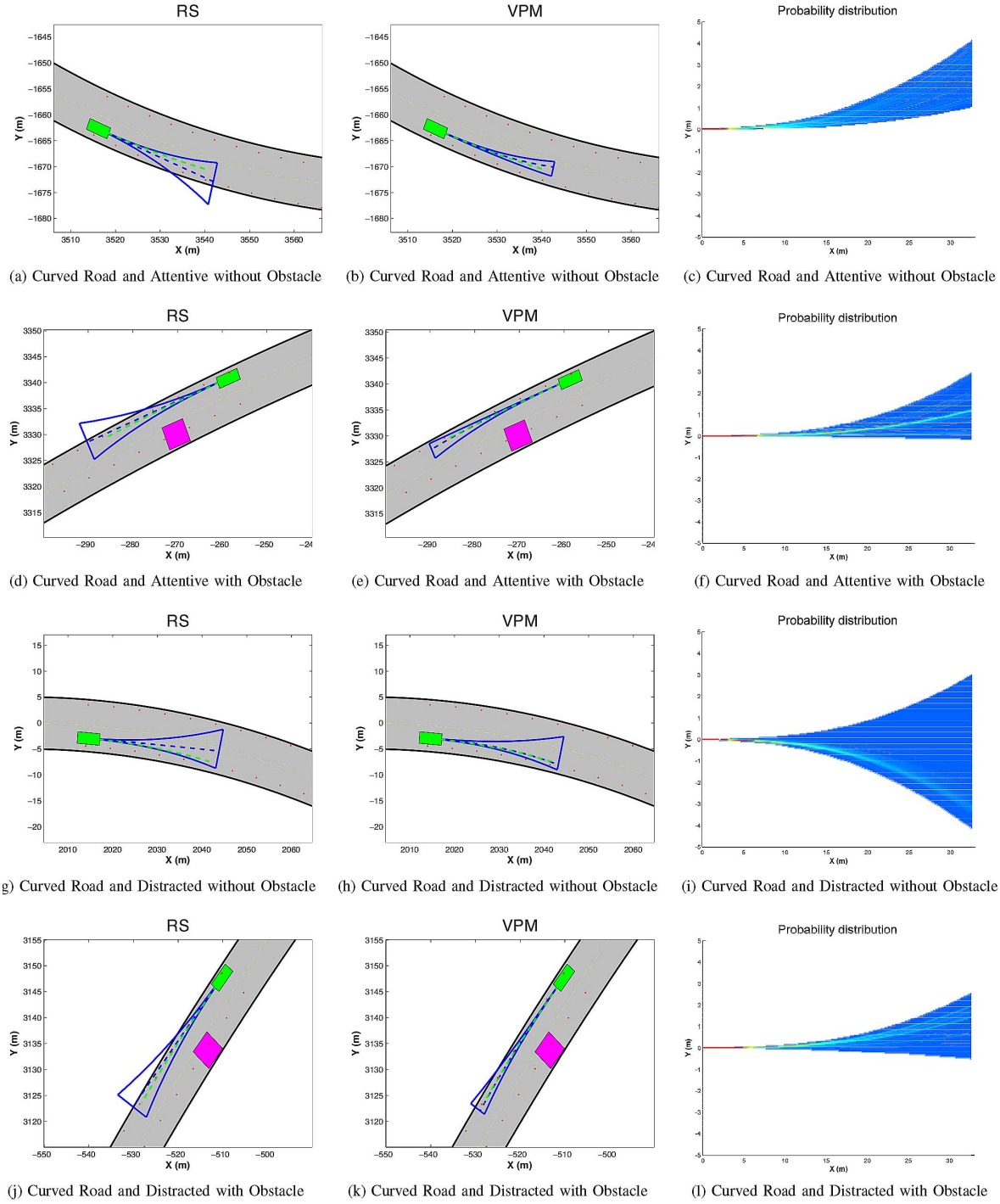
We also conducted surveys on the subjects and found that 75% of the subjects felt the controller was effective in preventing collisions, 21% neutral and only 4% felt it wasn't effective. 50% of the subjects felt the controller was too aggressive in intervening, 21% felt neutral and 29% felt it wasn't. Lastly, 58% of the subjects felt the knowledge that there was a safety controller changed their driving behavior and they were more risky in responding to texts.

5.3 Conclusions

We have demonstrated using examples from driving applications the effectiveness of using the extended MPC framework as a unifying way to integrate computational models of human behaviors into semi-autonomous systems. We proposed that since the MPC framework already includes state predictions in its infrastructure, it becomes a natural candidate to also include driver predictions in the control algorithm, which only involves adding extra constraints and cost terms corresponding to the predictions and controller intervention.

We demonstrated that using this MPC framework, we are not limited to using a particular human modeling structure for making predictions. Instead the controller can handle any open-loop sequence of model predictions over the next horizon, regardless of how they are generated. Therefore, however complex the model might be, i.e. optimization-based models, probabilistic Bayesian models, or even the descriptive cognitive architectures, as long as an open-loop sequence of predictions are generated, the MPC semi-autonomous controller can take the predictions into account.

Using examples from the switched system in the extreme driving maneuver, we demonstrated how even a switched strategy exhibited by the driver can be integrated in closed-loop with the MPC. We also showed that the complex cluster-based probabilistic model of distracted drivers can be included in open-loop, and that the results showed significantly less controller interventions that were unnecessary as compared to the worst case scenarios.



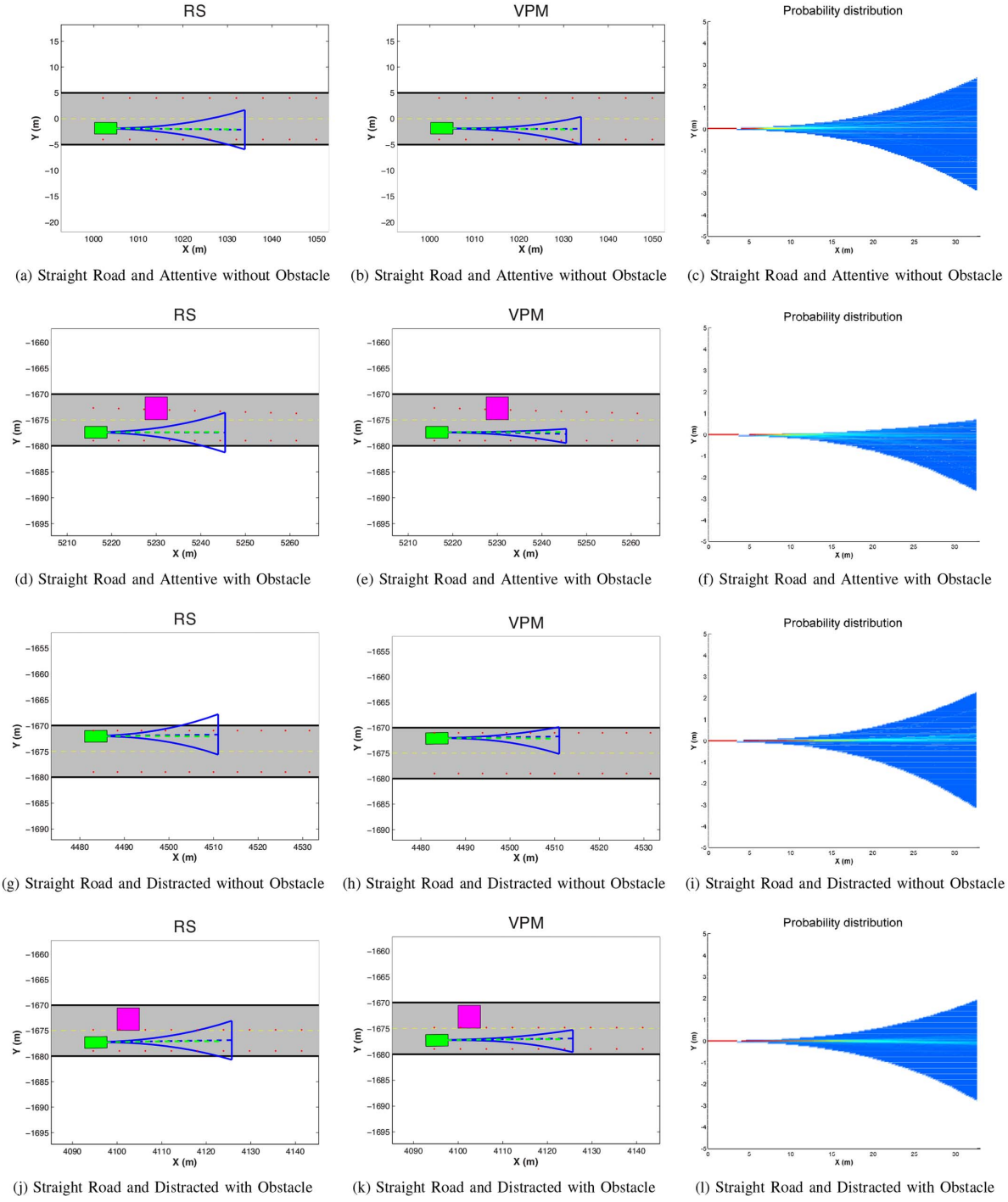
(a) Curved Road and Attentive without Obstacle (b) Curved Road and Attentive without Obstacle (c) Curved Road and Attentive without Obstacle

(d) Curved Road and Attentive with Obstacle (e) Curved Road and Attentive with Obstacle (f) Curved Road and Attentive with Obstacle

g) Curved Road and Distracted without Obstacle (h) Curved Road and Distracted without Obstacle (i) Curved Road and Distracted without Obstacle

(j) Curved Road and Distracted with Obstacle (k) Curved Road and Distracted with Obstacle (l) Curved Road and Distracted with Obstacle

Figure 5.3: Probability distribution of vehicle states with the identified cluster given observations, and predicted future state of the vehicle based on two different driver models: RS and VPM, for both distracted and attentive states of the driver, and on curved roads.



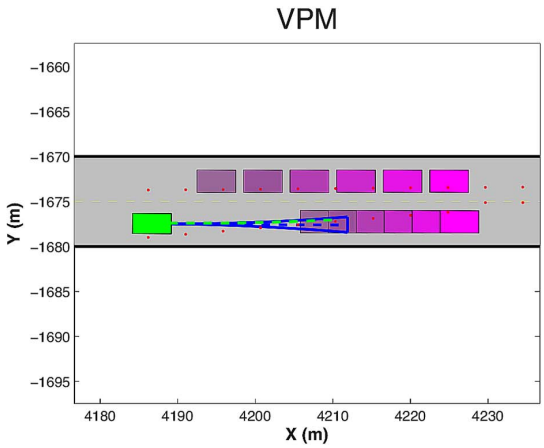
(a) Straight Road and Attentive without Obstacle (b) Straight Road and Attentive without Obstacle (c) Straight Road and Attentive without Obstacle

(d) Straight Road and Attentive with Obstacle (e) Straight Road and Attentive with Obstacle (f) Straight Road and Attentive with Obstacle

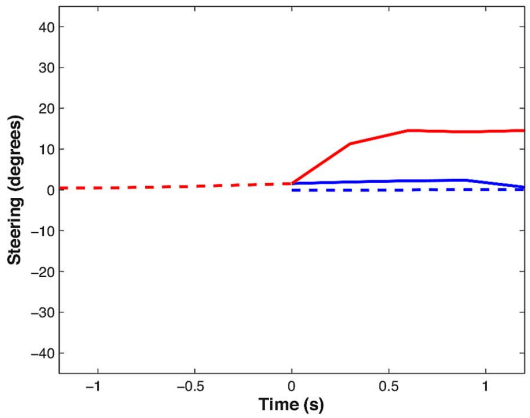
(g) Straight Road and Distracted without Obstacle (h) Straight Road and Distracted without Obstacle (i) Straight Road and Distracted without Obstacle

(j) Straight Road and Distracted with Obstacle (k) Straight Road and Distracted with Obstacle (l) Straight Road and Distracted with Obstacle

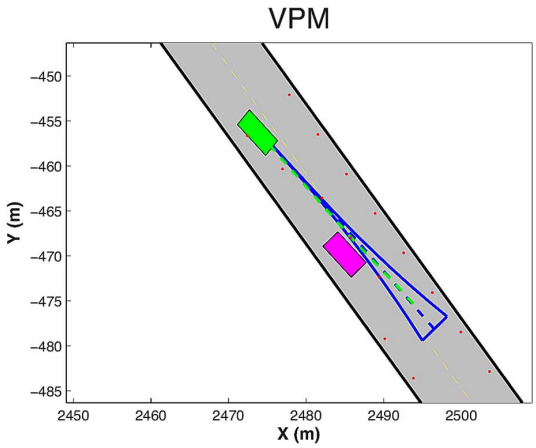
Figure 5.4: Probability distribution of vehicle states with the identified cluster given observations, and predicted future state of the vehicle based on two different driver models: RS and VPM, for both distracted and attentive states of the driver, and on straight roads.



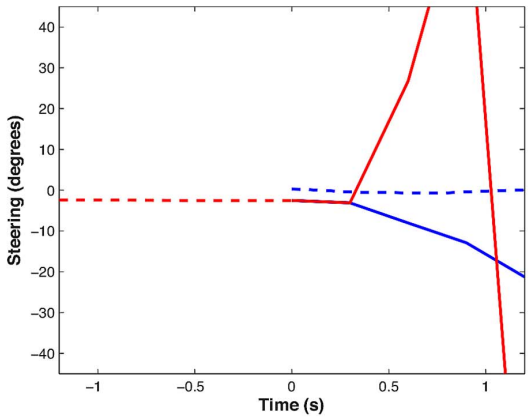
(a) Straight Road with Obstacle



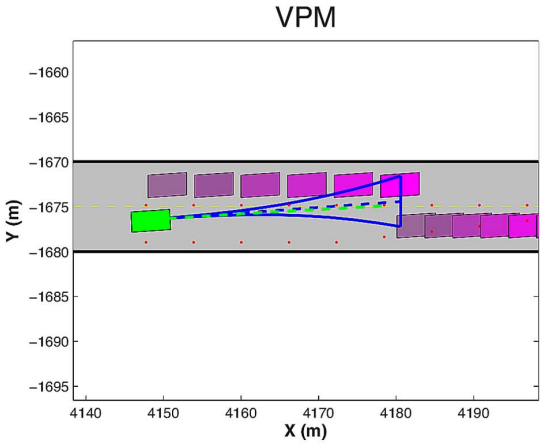
(b) Straight Road with Obstacle



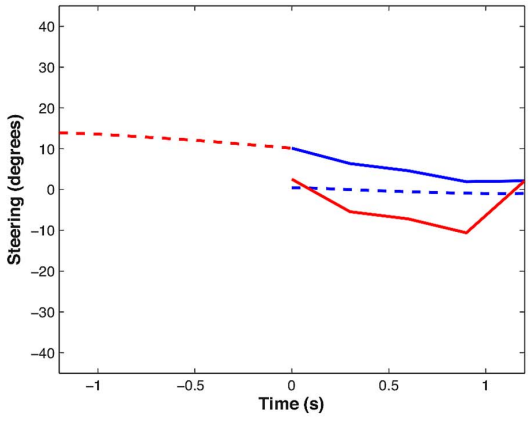
(c) Straight Road with Obstacle



(d) Straight Road with Obstacle



(e) Straight Road with Obstacle



(f) Straight Road with Obstacle

Figure 5.5: Results from semi-autonomous MPC framework with cluster-based probabilistic driver model during distracted driving.

Chapter 6

Conclusions

In this thesis, we integrated advanced control algorithms with computational models of human decision making in dynamic contexts in order to improve human-machine interactions in intelligent systems. We addressed the need to make connections between the human behavioral studies performed in psychology with real-world applications. To support these research directions, we have used advanced test apparatus and sensor equipment to collect experimental data and observe, directly and indirectly, the cognitive states of the human subject. To connect studies of human behavior from the psychological perspective to engineering applications, we performed experiments in the real-world and designed game-based scenarios which paralleled the corresponding driving applications. Parametric models were used instead of data-driven models in order to generalize behaviors to a wider context. With the experimental data, we used both the direct and optimization-based approaches to parametrically model the cognitive processes associated with the decision making in the presented dynamic scenarios. Using these advanced modeling techniques, we were able to connect model structure and parameters with psychological principles, and simulate how subjects switch between different controls strategies that is a combination of both modeling approaches. To integrate human models into controller design, we introduced an advanced framework using model predictive control to incorporate any computational model into the control algorithm.

6.1 Using control-based frameworks to model cognitive processes in a dynamic system

In Chapter 2, we introduced a framework capturing the interactions between the human agent with the dynamic system, and paralleled the typical components in a control system with the underlying cognitive processes associated with human behaviors. In order to properly model behaviors of the human in dynamic contexts, all interconnecting components of the control system need to be considered. State observers take noisy measurements from the sensory nervous system and combine sensor fusion with prior knowledge to produce an estimated state feedback from the associated dynamic system. This piece of information is crucial to the control algorithm component which maps

state feedback to the required input to the actuators in order to achieve a set of goals determined by a higher-level reference generator. This mapping, termed the decision policy, is the main cognitive process addressed in this thesis. The decision policy needs to address any discrepancies resulted from the various uncertainties associated with state estimation, actuation, or prior knowledge of the world, either through corrective action or internal predictions.

One limitation of the above presented framework is the singleton module of the decision policy. This structure oversimplifies the cognitive processes associated with the overall decision makings in dynamic environments and constrains the model to deal with a single layer of abstraction. As discussed in Section 1.3.1, there are multiple layers of abstraction to consider when modeling human behaviors, and lower-level decisions are often coupled with a hierarchy of higher-level decision making. Therefore, as an extension to the analysis of the framework, multiple levels of decision policies can be simultaneously posed to encompass this broader context.

6.2 Using advanced technologies to bridge psychological principles with real-world applications

In Chapter 3, we presented the experiments designed to connect studies of human behavior from the psychological perspective to engineering applications. Our real-world application consists of the task of driving under various circumstances. Our test vehicle, equipped with an inertial measurement unit, differential GPS and forward looking cameras, provided us with the necessary measurements of vehicle states, road and traffic conditions in driving scenarios on real roads. With access to two testing facilities the Smithers testing center and the Hyundai-Kia Motor California Proving Ground, we were able to collect data from various driving scenarios including highway driving, distracted driving, and extreme maneuvers in slippery conditions. This broad range of driving contexts allows us to explore the different factors associated with human behaviors from various modeling methodologies. Experimental setups on the virtual simulator with skeletal tracking allowed us to test the implementation of our semi-autonomous controller which guarantees safety of a distracted driver in dangerous situations.

To study the fundamental principles behind human decision making in a controlled environment and draw connections to the above discussed driving applications, we designed game-based experiments which parallels similar scenarios in the context of driving. The projectile game is designed to present a nonlinear mapping function between the control input and the state trajectories with uncertainty and constraints, which matches the nonlinearity and uncertainty associated with different road conditions in driving, as well as constraints in traffic conditions. With the dual-task game, we simulated a scenario which parallels the cognitive processes involved with texting while driving, where attention allocation has to be divided between a primary and a secondary task.

Many of the decision making experiments did not directly measure the cognitive states of the human subject. In our modeling paradigm, we followed the assumption that humans will intrinsically perform mental simulations of states in order to search for a feasible solution to the problem. To validate this conjecture, we used head-mounted eye-tracking technology to directly measure the

cognitive states of the human, in terms of attention allocation and state predictions. One limitation is that in some contexts, humans will use their peripheral vision to obtain information, therefore even though the point-of-gaze provides valuable information, it is still unclear how the use peripheral vision can be measured.

Even with the combination of both game-based and driving experiments, the problem context still sits at very far ends of the spectrum which relate simple cognitive problems with real-world applications. Game-based experiments, although provide a controlled setting for creating arbitrary scenarios and collect consistent data, suffers from the lack of sensory feedback and motor control involved in real-world applications. It will be beneficial to design experimental contexts which lie somewhere in between the two extremes. With recent advancements in virtual reality [135], we now have the capability to include 3D vision with virtual reality glasses such as the Oculus Rift [136], tracking full body and hand movements with non-invasive skeletal tracking technologies [102], force and haptic feedback through virtual gloves [137]. Designing experiments with these new technologies will provide the same controlled setting as computer games at the same time provide subjects with a more realistic experience of the world.

6.3 Using advanced models to understand cognitive principles in decision policies

In Chapter 4 we used data collected from the experiments described in Chapter 3, and the two approaches of modeling detailed in Chapter 2 to analysis human behaviors with principles of psychology in different decision making contexts, involving both continuous and discrete decisions. The first approach of direct methods maps observed input to the control decisions directly without extra computations involved with solving for an optimal solution. This methodology is suitable for mapping lower-level decisions which are repetitive and requires no planning. The second approach used an optimization-based framework, where subjects are postulated to perform mental simulations and state predictions in order to satisfy constraints and minimize the cost function.

Dynamic problems involving continuous decision: Analyzing subject's response in the projectile game, we observed that a switched control strategy was used due to nonlinearity and uncertainty associated with the function mapping inputs to states of the system. With old scenarios where the trajectory feedback was closer to the target position, linearization about the current input values produced less errors than if the feedback was further away, typically the case with new scenarios. In addition, subjects showed preference towards changing the velocity input over the angle since velocity appears linearly in the function. Therefore, it is more likely that subjects adopted the direct method in these scenarios. On the contrary, in new scenarios where linearization fails, subjects adopted a strategy which involved mentally simulating predicted trajectories in order to hit the target and avoid obstacle. Subjects were shown to minimize a combination of minimum change in input and time to hit the target in order to minimize the effect of input uncertainty on the state trajectories. Reaction times also supported the switching between strategies.

Similar to the effect of nonlinearity in the projectile game, subjects also showed a switched control strategy in extreme driving where both the linear and the saturated regions of the tire need to be utilized. A hybrid model was used to fit the steering behavior of drivers and it was shown that a two-mode model was sufficient. We also showed by using model predictive control that subjects need to have prior knowledge of the nonlinearity in the tire in order to follow the trajectory of the required maneuver. This showed that forward models of the system are considered in the decision making process. The existence of forward models and state predictions were validated through analyzing eye-tracking data in the context of driving and juggling.

Even though we were able to show that uncertain knowledge of the forward model and its associated nonlinearity lead to a switched control strategy in both the game-based scenario and real-world driving, the inputs of the projectile game are applied at discrete intervals, whereas the steering inputs of driving are being adjusted constantly. In addition, the constraints associated with the obstacles in the projectile game are stationary, but in the context of on-road traffic, constraints have a temporal component associated it with. Therefore, to improve our aim to connect game-based principles to the real-world, we need to redesign our experiment to allow, first the continuous application of inputs and second, the incorporation of moving obstacles.

Dynamic problems involving discrete decisions: Optimization-based methods of the Markov decision process was used to study the discrete decision of attention allocation in multi-task problems in the context of texting while driving. Model simulations and inverse reinforcement learning from data of the dual-task game showed a shift in the weights in the cost function, that as the subjects become more familiar with the transition probabilities associated with the game, they become more comfortable in switching to the secondary task more often and at higher risks. Applying the same framework to the context of distracted driving in the test center, the value function showed a negative correlation with the vehicle's longitudinal velocity and complies with the attention durations on the texting task measured with the eye-tracker. We showed that the same decision making framework applies to both the game-based experiment and real-world driving.

To further improve the connection between the game-based experiments with the context of driving, on top of attention allocation in the secondary task, the lower-level decision making associated with the primary task for both contexts should match. Therefore, further experiments should be conducted where the dual-task game is redesigned to involve a primary task which parallel steering inputs, or real driving experiments should be performed to involve lane switching.

6.4 Using advanced control frameworks to integrate human behaviors with intelligent systems

In Chapter 5 we pieced together the previously discussed modeling efforts and addressed our original motivation of improving human-machine interactions through the integration of human models with control algorithms. We introduced a unifying framework using model predictive control (MPC) extended to include both predictions of human behavior and the minimization of controller

intervention. We presented both simulation results and real implementations of a safety controller on the simulator the effectiveness of the extended MPC which included predictions of driver models both in closed-loop and open-loop. We showed that with the inclusion of the driver model, the controller was able to intervene on less occasions. We demonstrated that using this MPC framework, we are not limited to using a particular human modeling structure for making predictions. Instead the controller can handle any open-loop sequence of model predictions over the next horizon, regardless of how they are generated. Therefore, even though the presented examples involved only predictions from a single driver, the framework should be able to be extended to include other drivers or decision making agents in the environment, and design a system which can interact with multiple agents simultaneously. The limiting factor in the inclusion of additional models, posed as constraints, will be the feasibility and computational complexity involved with the new optimization problem.

6.5 Summary of original contributions

Lastly, we provide a summary of the above discussed contributions and emphasize the originality of the research in this thesis. We used parametric approaches to model and analyze human behaviors. Inspired from the field of controls theory and computer science, we used model structures typically used in control systems to model human decision making in dynamic problems, and analyzed the model parameters from the perspectives of cognitive psychology. We differ from the data-driven approach which mainly focuses on the accuracy of identification and predictions and does not take into account the cognitive factors influencing human behaviors. We also differ from the hand reaching experiments in cognitive psychology by dealing with tasks that involve a dynamic environment such as the effects of gravity in the projectile game and in driving. We attempted to bridge the cognitive principles studied in simple controlled experiments in psychology with real-world dynamic tasks by designing paralleling experiments in both contexts. For example, the dual-task game is designed to mirror the decision of attention allocation in the context of texting while driving.

To model the decision policies involved in a typical dynamic task, we proposed that modeling methods can be divided into two approaches: direct and optimization-based methods. We related the principles behind these approaches to cognitive processes and discussed how lower-level familiar tasks are associated with the direct method, whereas higher-level planning tasks are associated with the optimization-based approach. Through the projectile game, we were able to propose the novel idea that subjects used a combination of the two approaches depending on the scenarios presented to them. With the help of advanced eye-tracking technology, we were able to directly measure the cognitive states of the human subject in texting, driving and juggling, and used these observations to support the idea of forward models and mental simulations essential to the optimization-based approaches.

Our novel control framework, using an extended version of model predictive control, is able to directly include any computational models of human behavior into controller design and minimize the amount of controller intervention. This framework structure is able to include complex models

as long as the model can generate sequences of predictions for the next horizon.

6.6 Future work

In Section 6.1, we proposed a higher-fidelity representation of the human decision making framework as having a hierarchical nature where lower-level decision making are coupled with higher-level abstractions similar to the framework used in [67] where autonomous driving involved both path-planning and path-following. Similarly, our study of both continuous and discrete decision variables in Section 6.3 can be viewed as a decomposition of a hierarchical task, where at the higher-level, discrete decisions of attention allocation is decided between the primary and secondary task, and at the lower-level, continuous decisions of steering inputs are applied to the primary task. Future experiments, designed both from a game-based perspective and real-world driving, can be performed to draw parallels between psychological principles and engineering applications in this hierarchical context. In addition, the use of virtual reality technology can be included to further bridge the two extremities of the research fields.

Bibliography

[1] Annabelle Gawer and Michael A Cusumano. *Platform leadership: How Intel, Microsoft, and Cisco drive industry innovation*. Harvard Business School Press Boston, 2002.

[2] Hans Meixner. *Sensors, Micro-and Nanosensor Technology: Trends in Sensor Markets*. Vol. 8. John Wiley & Sons, 2008.

[3] Lawrence A Klein. *Sensor and data fusion: a tool for information assessment and decision making*. Vol. 324. Spie Press Bellingham‸ eWA WA, 2004.

[4] Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[5] Michael A Goodrich and Alan C Schultz. "Human-robot interaction: a survey". In: *Foundations and trends in human-computer interaction* 1.3 (2007), pp. 203–275.

[6] Consumer Reports. *Avoiding crashes with self-driving cars*. Feb. 2014. URL: http://www.consumerreports.org/cro/magazine/2014/04/the-road-to-self-driving-cars/index.htm (visited on 12/01/2015).

[7] Ravi Shanker et al. *Autonomous Cars: Self-Driving the New Auto Industry Paradigm*. Morgan Stanley Research, 2013.

[8] Sebastian Thrun et al. "Stanley: The robot that won the DARPA Grand Challenge". In: *Journal of field Robotics* 23.9 (2006), pp. 661–692.

[9] Chris Urmson et al. "Autonomous driving in urban environments: Boss and the urban challenge". In: *Journal of Field Robotics* 25.8 (2008), pp. 425–466.

[10] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The 2005 DARPA grand challenge: the great robot race*. Vol. 36. Springer Science & Business Media, 2007.

[11] Mark Campbell et al. "Autonomous driving in urban environments: approaches, lessons and challenges". In: *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 368.1928 (2010), pp. 4649–4672.

[12] Joel C McCall and Mohan M Trivedi. "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation". In: *Intelligent Transportation Systems, IEEE Transactions on* 7.1 (2006), pp. 20–37.

[13] Sanghyun Hong et al. "Tyre–road friction coefficient estimation based on tyre sensors and lateral tyre deflection: modelling, simulations and experiments". In: *Vehicle System Dynamics* 51.5 (2013), pp. 627–647.

[14] Ronald K Jurgen. "Adaptive cruise control". In: *Training* 2012 (2006), pp. 11–19.

[15] Shengbo Li et al. "Model predictive multi-objective vehicular adaptive cruise control". In: *Control Systems Technology, IEEE Transactions on* 19.3 (2011), pp. 556–566.

[16] Johan Bengtsson. *Adaptive cruise control and driver modeling*. Department of Automatic Control, Lund Institute of Technology, 2001.

[17] Andrew Gray et al. "Robust predictive control for semi-autonomous vehicles with an uncertain driver model". In: *Intelligent Vehicles Symposium (IV), 2013 IEEE*. IEEE. 2013, pp. 208–213.

[18] Ashwin Carvalho et al. "Robust vehicle stability control with an uncertain driver model". In: *Control Conference (ECC), 2013 European*. IEEE. 2013, pp. 440–445.

[19] Victor Shia et al. "Semiautonomous vehicular control using driver modeling". In: *Intelligent Transportation Systems, IEEE Transactions on* 15.6 (2014), pp. 2696–2709.

[20] Andrew Gray et al. "Stochastic predictive control for semi-autonomous vehicles with an uncertain driver model". In: *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*. IEEE. 2013, pp. 2329–2334.

[21] Gill Pratt and Justin Manzo. "The DARPA robotics challenge". In: *Robotics & Automation Magazine, IEEE* 20.2 (2013), pp. 10–12.

[22] Boston Dynamics. *Atlas - The Agile Anthropomorphic Robot*. URL: `http://www.bostondynamics.com/robot_Atlas.html` (visited on 12/01/2015).

[23] Danna Voth. "A new generation of military robots". In: *Intelligent Systems, IEEE* 19.4 (2004), pp. 2–3.

[24] Masato Hirose and Kenichi Ogawa. "Honda humanoid robots development". In: *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 365.1850 (2007), pp. 11–19.

[25] Joel C McCall and Mohan M Trivedi. "Driver behavior and situation aware brake assistance for intelligent vehicles". In: *Proceedings of the IEEE* 95.2 (2007), pp. 374–387.

[26] Brenna D Argall et al. "A survey of robot learning from demonstration". In: *Robotics and autonomous systems* 57.5 (2009), pp. 469–483.

[27] Peter Pastor et al. "Learning and generalization of motor skills by learning from demonstration". In: *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE. 2009, pp. 763–768.

[28] John Schulman et al. "Learning from demonstrations through the use of non-rigid registration". In: *Proceedings of the 16th International Symposium on Robotics Research (ISRR)*. 2013.

[29] Jun Nakanishi et al. "Learning from demonstration and adaptation of biped locomotion". In: *Robotics and Autonomous Systems* 47.2 (2004), pp. 79–91.

[30] Christopher G Atkeson et al. "Using humanoid robots to study human behavior". In: *IEEE Intelligent Systems and their applications* 15.4 (2000), pp. 46–56.

[31] Nuria Oliver and Alex P Pentland. "Graphical models for driver behavior recognition in a smartcar". In: *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*. IEEE. 2000, pp. 7–12.

[32] John Krumm. *A markov model for driver turn prediction*. Tech. rep. SAE Technical Paper, 2008.

[33] Andrew Gray et al. "Integrated threat assessment and control design for roadway departure avoidance". In: *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*. IEEE. 2012, pp. 1714–1719.

[34] Stéphanie Lefevre et al. "Lane keeping assistance with learning-based driver model and model predictive control". In: *Proceedings of the 12th international symposium on advanced vehicle control*. 2014.

[35] Stéphanie Lefèvre, Ashwin Carvalho, and Francesco Borrelli. "Autonomous car following: A learning-based approach". In: *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE. 2015, pp. 920–926.

[36] Stéphanie Lefèvre et al. "Driver models for personalised driving assistance". In: *Vehicle System Dynamics* (2015), pp. 1–16.

[37] Pieter Abbeel and Andrew Y Ng. "Apprenticeship learning via inverse reinforcement learning". In: *Proceedings of the twenty-first international conference on Machine learning*. ACM. 2004, p. 1.

[38] Wayne D Gray. "Integrated models of cognitive systems." In: *AFOSR Cognitive Modeling Workshop, Mar, 2005, Saratoga Inn, Saratoga Springs, NY, US; The chapters in this volume are the result of the aforementioned workshop.* Oxford University Press. 2007.

[39] Dario D Salvucci. "Modeling driver behavior in a cognitive architecture". In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 48.2 (2006), pp. 362–380.

[40] Dario D Salvucci and Niels A Taatgen. *The multitasking mind*. Oxford University Press, USA, 2010.

[41] David Marr. "Vision: A computational investigation into the human representation and processing of visual information". In: *WH San Francisco: Freeman and Company* (1982).

[42] James L McClelland. "Connectionist models and Bayesian inference". In: *Rational models of cognition* (1998), pp. 21–53.

[43] Thomas L Griffiths et al. "Probabilistic models of cognition: Exploring representations and inductive biases". In: *Trends in cognitive sciences* 14.8 (2010), pp. 357–364.

[44] James L McClelland et al. "Letting structure emerge: connectionist and dynamical systems approaches to cognition". In: *Trends in cognitive sciences* 14.8 (2010), pp. 348–356.

[45] Simon Haykin. *Neural Networks: A comprehensive foundation*. 2nd ed. Prentice Hall, 1994.

[46] Kazuo Tanaka and Michio Sugeno. "Stability analysis and design of fuzzy control systems". In: *Fuzzy sets and systems* 45.2 (1992), pp. 135–156.

[47] Yoshua Bengio. "Learning deep architectures for AI". In: *Foundations and trends® in Machine Learning* 2.1 (2009), pp. 1–127.

[48] Yoshua Bengio, Aaron Courville, and Pascal Vincent. "Representation learning: A review and new perspectives". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 35.8 (2013), pp. 1798–1828.

[49] Quoc V Le et al. "Building high-level features using large scale unsupervised learning". In: *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE. 2013, pp. 8595–8598.

[50] Jeffrey L Elman. *Rethinking innateness: A connectionist perspective on development*. MIT press, 1998.

[51] Henry Markram. "The blue brain project". In: *Nature Reviews Neuroscience* 7.2 (2006), pp. 153–160.

[52] A Aldo Faisal, Luc PJ Selen, and Daniel M Wolpert. "Noise in the nervous system". In: *Nature Reviews Neuroscience* 9.4 (2008), pp. 292–303.

[53] JH Byrne and N. Dafny (eds.) "Neuroscience Online: An Electronic Textbook for the Neurosciences". In: *Department of Neurobiology and Anatomy, The University of Texas Medical School at Houston* (1997).

[54] Vittorio Gallese and Alvin Goldman. "Mirror neurons and the simulation theory of mind-reading". In: *Trends in cognitive sciences* 2.12 (1998), pp. 493–501.

[55] Marc Jeannerod. "Neural simulation of action: a unifying mechanism for motor cognition". In: *Neuroimage* 14.1 (2001), S103–S109.

[56] Beatriz Calvo-Merino et al. "Action observation and acquired motor skills: an FMRI study with expert dancers". In: *Cerebral cortex* 15.8 (2005), pp. 1243–1249.

[57] Ed Vul et al. "Explaining human multiple object tracking as resource-constrained approximate inference in a dynamic probabilistic model". In: *Advances in neural information processing systems*. 2009, pp. 1955–1963.

[58] Gene F Franklin, J David Powell, and Michael L Workman. *Digital control of dynamic systems*. 3rd ed. Addison-Wesley Menlo Park, 1998.

[59] Michel Desmurget and Scott Grafton. "Forward modeling allows feedback control for fast reaching movements". In: *Trends in cognitive sciences* 4.11 (2000), pp. 423–431.

[60] Daniel M Wolpert and Mitsuo Kawato. "Multiple paired forward and inverse models for motor control". In: *Neural networks* 11.7 (1998), pp. 1317–1329.

[61] R Christopher Miall. "Connecting mirror neurons and forward models". In: *Neuroreport* 14.17 (2003), pp. 2135–2137.

[62] Robert J van Beers, Pierre Baraduc, and Daniel M Wolpert. "Role of uncertainty in sensorimotor control". In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 357.1424 (2002), pp. 1137–1145.

[63] Konrad P Körding and Daniel M Wolpert. "Bayesian integration in sensorimotor learning". In: *Nature* 427.6971 (2004), pp. 244–247.

[64] Daniel M Wolpert and Michael S Landy. "Motor control is decision-making". In: *Current opinion in neurobiology* 22.6 (2012), pp. 996–1003.

[65] John R Anderson. *Cognitive psychology and its implications*. 6th ed. Macmillan, 2005.

[66] Stephen H Scott. "Optimal feedback control and the neural basis of volitional motor control". In: *Nature Reviews Neuroscience* 5.7 (2004), pp. 532–546.

[67] Yiqi Gao et al. "Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads". In: *ASME 2010 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers. 2010, pp. 265–272.

[68] Lauralee Sherwood. *Human physiology: from cells to systems*. 9th ed. Cengage Learning, 2015.

[69] Rudolf Emil Kalman. "Contributions to the theory of optimal control". In: *Bol. Soc. Mat. Mexicana* 5.2 (1960), pp. 102–119.

[70] Richard A Schmidt and Tim Lee. *Motor control and learning*. 5th ed. Human Kinetics, 2011.

[71] Eduardo F Camacho and Carlos Bordons. *Model predictive control*. 2nd ed. Springer-Verlag London, 2007.

[72] Andrew J Pick and David J Cole. "A mathematical model of driver steering control including neuromuscular dynamics". In: *Journal of Dynamic Systems, Measurement, and Control* 130.3 (2008), p. 031004.

[73] Reza Shadmehr and Steven P Wise. *The computational neurobiology of reaching and pointing: a foundation for motor learning*. MIT press, 2005.

[74] Leif Johnson et al. "Predicting human visuomotor behaviour in a driving task". In: *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 369.1636 (2014), p. 20130044.

[75] Alessandro Alessio and Alberto Bemporad. "A survey on explicit model predictive control". In: *Nonlinear model predictive control*. Springer, 2009, pp. 345–369.

[76] Robert Tibshirani. "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), pp. 267–288.

[77] Lennart Ljung. *System Identification: Theory for the User*. 2nd ed. Prentice Hall, 1999.

[78] Jan Lunze and Françoise Lamnabhi-Lagarrigue. *Handbook of hybrid systems control: theory, tools, applications*. Cambridge University Press, 2009.

[79] Simone Paoletti et al. "Identification of hybrid systems a tutorial". In: *European journal of control* 13.2 (2007), pp. 242–260.

[80] G Ferrari-Trecate. "Hybrid identification toolbox (HIT)". In: *Zurich, Switzerland* (2005).

[81] Rui Xu, Donald Wunsch, et al. "Survey of clustering algorithms". In: *Neural Networks, IEEE Transactions on* 16.3 (2005), pp. 645–678.

[82] John A Hartigan and Manchek A Wong. "Algorithm AS 136: A k-means clustering algorithm". In: *Applied statistics* (1979), pp. 100–108.

[83] Nello Cristianini and John Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[84] Hyeran Byun and Seong-Whan Lee. "Applications of support vector machines for pattern recognition: A survey". In: *Pattern recognition with support vector machines*. Springer, 2002, pp. 213–236.

[85] Isabelle Guyon et al. "Gene selection for cancer classification using support vector machines". In: *Machine learning* 46.1-3 (2002), pp. 389–422.

[86] Thorsten Joachims. *Learning to classify text using support vector machines: Methods, theory and algorithms*. Kluwer Academic Publishers, 2002.

[87] Jason Weston and Chris Watkins. *Multi-class support vector machines*. Tech. rep. Citeseer, 1998.

[88] Chih-Chung Chang and Chih-Jen Lin. "LIBSVM: A library for support vector machines". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 2.3 (2011), p. 27.

[89] Andrew Gray et al. "Predictive control for agile semi-autonomous ground vehicles using motion primitives". In: *American Control Conference (ACC), 2012*. IEEE. 2012, pp. 4239–4244.

[90] Philip E Gill et al. *User's Guide for NPSOL (Version 4.0): A Fortran Package for Nonlinear Programming.* Tech. rep. DTIC Document, 1986.

[91] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[92] Andrew Y Ng, Stuart J Russell, et al. "Algorithms for inverse reinforcement learning." In: *Icml*. 2000, pp. 663–670.

[93] Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. "Maximum margin planning". In: *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 729–736.

[94] Egbert Bakker, Lars Nyborg, and Hans B Pacejka. *Tyre modelling for use in vehicle dynamics studies*. Tech. rep. SAE Technical Paper, 1987.

[95] Tesla Motors. *Model S Specifications*. URL: https://www.teslamotors.com/support/model-s-specifications (visited on 12/01/2015).

[96] Motor Trend. *2013 Motor Trend Car of the Year: Contenders and Finalists - Photos*. URL: http://www.motortrend.com/news/2013-motor-trend-car-of-the-year-contenders-and-finalists/photos/ (visited on 12/01/2015).

[97] Andrew Duchowski. *Eye tracking methodology: Theory and practice*. Vol. 373. Springer Science & Business Media, 2007.

[98] SensoMotoric Instruments. URL: http://www.smivision.com/ (visited on 12/01/2015).

[99] EyeTracking. *Hardware: Eye Tracking Systems*. URL: http://www.eyetracking.com/Hardware/Eye-Tracker-List (visited on 12/01/2015).

[100] Zhengyou Zhang. "Microsoft kinect sensor and its effect". In: *MultiMedia, IEEE* 19.2 (2012), pp. 4–10.

[101] Ramanarayan Vasudevan et al. "Safe semi-autonomous control with enhanced driver modeling". In: *American Control Conference (ACC), 2012*. IEEE. 2012, pp. 2896–2903.

[102] Alper Yilmaz, Omar Javed, and Mubarak Shah. "Object tracking: A survey". In: *Acm computing surveys (CSUR)* 38.4 (2006), p. 13.

[103] Karen Aldana. *U.S. Department of Transportation Releases Policy on Automated Vehicle Development*. URL: http://www.nhtsa.gov/About+NHTSA/Press+Releases/U.S.+Department+of+Transportation+Releases+Policy+on+Automated+Vehicle+Development (visited on 12/01/2015).

[104] Efstathios Velenis, Emilio Frazzoli, and Panagiotis Tsiotras. "On steady-state cornering equilibria for wheeled vehicles with drift". In: *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*. IEEE. 2009, pp. 3545–3550.

[105] Kristie Young, John D Lee, and Michael A Regan. *Driver distraction: Theory, effects, and mitigation*. CRC Press, 2008.

[106] Joanne L Harbluk, Y Ian Noy, and Moshe Eizenman. *The impact of cognitive distraction on driver visual behaviour and vehicle control*. Tech. rep. 2002.

[107] Peter J Beek and Arthur Lewbel. "The science of juggling". In: *Scientific American* 273.5 (1995), pp. 92–97.

[108] M Rodrigues and P Simeão Carvalho. "Teaching physics with Angry Birds: exploring the kinematics and dynamics of the game". In: *Physics Education* 48.4 (2013), p. 431.

[109] Jessica B Hamrick et al. "Think again? The amount of mental simulation tracks uncertainty in the outcome". In: *Proceedings of the 37th Annual Conference of the Cognitive Science Society*. 2015.

[110] Andrew Gray et al. "Semi-autonomous vehicle control for road departure and obstacle avoidance". In: *IFAC Control of Transportation Systems* (2012), pp. 1–6.

[111] Eric L Amazeen, Polemnia G Amazeen, and Peter J Beek. "Eye movements and the selection of optical information for catching". In: *Ecological Psychology* 13.2 (2001), pp. 71–85.

[112] Joost C Dessing, Frédéric P Rey, and Peter J Beek. "Gaze fixation improves the stability of expert juggling". In: *Experimental brain research* 216.4 (2012), pp. 635–644.

[113] K Sjöstrand. "Matlab implementation of LASSO". In: *LARS, the elastic net and SPCA, Version* 2 (2005).

[114] Stefan Schaal and Christopher G Atkeson. "Robot juggling: implementation of memory-based learning". In: *Control Systems, IEEE* 14.1 (1994), pp. 57–71.

[115] Anca D Dragan, Kenton CT Lee, and Siddhartha S Srinivasa. "Legibility and predictability of robot motion". In: *Human-Robot Interaction (HRI), 2013 8th ACM/IEEE International Conference on*. IEEE. 2013, pp. 301–308.

[116] Emanuel Todorov and Michael I Jordan. "Optimal feedback control as a theory of motor coordination". In: *Nature neuroscience* 5.11 (2002), pp. 1226–1235.

[117] Johan Löfberg. "YALMIP: A toolbox for modeling and optimization in MATLAB". In: *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*. IEEE. 2004, pp. 284–289.

[118] Jonathan Currie and David I. Wilson. "OPTI: Lowering the Barrier Between Open Source Optimizers and the Industrial MATLAB User". In: *Foundations of Computer-Aided Process Operations*. Ed. by Nick Sahinidis and Jose Pinto. Savannah, Georgia, USA, 2012.

[119] Andreas Wächter and Lorenz T Biegler. "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming". In: *Mathematical programming* 106.1 (2006), pp. 25–57.

[120] Christopher G Lucas et al. "A rational model of function learning". In: *Psychonomic bulletin & review* (2015), pp. 1–23.

[121] Edmund Donges. "A two-level model of driver steering behavior". In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 20.6 (1978), pp. 691–707.

[122] Charles C Macadam. "Understanding and modeling the human driver". In: *Vehicle System Dynamics* 40.1-3 (2003), pp. 101–134.

[123] Dianne Parker et al. "Driving errors, driving violations and accident involvement". In: *Ergonomics* 38.5 (1995), pp. 1036–1048.

[124] Ari Katila et al. "Does increased confidence among novice drivers imply a decrease in safety?: the effects of skid training on slippery road accidents". In: *Accident Analysis & Prevention* 36.4 (2004), pp. 543–550.

[125] P. Carlo Cacciabue. *Modelling Driver Behaviour in Automotive Environments: Critical Issues in Driver Interactions with Intelligent Transport Systems*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007. ISBN: 1846286174.

[126] Daniel Kahneman and Amos Tversky. "Prospect theory: An analysis of decision under risk". In: *Econometrica: Journal of the Econometric Society* (1979), pp. 263–291.

[127] Kevin A Smith et al. "Physical predictions over time". In: *Proceedings of the 35th annual meeting of the cognitive science society*. 2013, pp. 1–6.

[128] Kevin A Smith and Edward Vul. "Prospective uncertainty: The range of possible futures in physical predictions". In: *Proceedings of the 37th Annual Conference of the Cognitive Science Society*. 2015.

[129] Michael F Land. "Eye movements and the control of actions in everyday life". In: *Progress in retinal and eye research* 25.3 (2006), pp. 296–324.

[130] Sofia Crespi et al. "Spotting expertise in the eyes: Billiards knowledge as revealed by gaze shifts in a dynamic visual prediction task". In: *Journal of vision* 12.11 (2012), p. 30.

[131] Brian Scassellati and Bradley Hayes. "Human-robot collaboration". In: *AI Matters* 1.2 (2014), pp. 22–23.

[132] Erico Guizzo. "How googles self-driving car works". In: *IEEE Spectrum Online, October* 18 (2011).

[133] Alistair Barr and Mike Ramsey. *Google Tries to Make Its Cars Drive More Like Humans*. URL: `http://www.wsj.com/articles/google-tries-to-make-its-cars-drive-more-like-humans-1443463523` (visited on 12/01/2015).

[134] Rami Y Hindiyeh and J Christian Gerdes. "Equilibrium analysis of drifting vehicles for control design". In: *ASME 2009 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers. 2009, pp. 181–188.

[135] Grigore Burdea and Philippe Coiffet. "Virtual reality technology". In: *Presence: Teleoperators and virtual environments* 12.6 (2003), pp. 663–664.

[136] VR Oculus. "Oculus rift-virtual reality headset for 3d gaming". In: *URL: http://www. oculusvr. com* (2012).

[137] Grigore C Burdea. *Force and touch feedback for virtual reality*. Wiley New York, 1996.

Appendix A

Appendix

A.1 Baseline solutions to the projectile problem

The projectile problem involves kinematic equations with gravity in the vertical direction as described in Section 3.2.2. Recall from Equation (2.9), the optimization problem is posed as,

$$\min_x J(x) \tag{A.1a}$$
$$\text{subject to} f(x) = 0 \tag{A.1b}$$
$$g(x) \leq 0 \tag{A.1c}$$

Although the decision making model posed in Section 4.2.1 for the human subject includes uncertainty in the formulations, but we will try and solve the deterministic problem here since we are only solving for the baseline solutions.

We will be solving four different sets of baseline solutions for four different *cost functions* $J(x)$:

- Minimum change in initial angle input: $J_1 = \|\Delta\alpha\|$
- Minimum change in initial velocity input: $J_2 = \|\Delta v\|$
- Minimum time to hit target $J_3 = T_{\text{targ}}$
- Maximum time to hit target $J_4 = -T_{\text{targ}}$

The *equality constraints* $f(x) = 0$ as described in Section 2.3 are kinematic relationships, repeated here for clarity,

$$x(t) = v\cos(\alpha)t \tag{A.2a}$$
$$y(t) = v\sin(\alpha)t + 0.5gt^2 \tag{A.2b}$$
$$p(t) = (x(t), y(t)) \tag{A.2c}$$

The *inequality constraints* $g(x) \leq 0$ are associated with successfully hitting the target at the same time avoiding the obstacle, again repeated for clarity,

$$\exists\, t > 0,\ \|p(t) - p_{\text{targ}}\| < R_{\text{targ}} + R_{\text{proj}} \tag{A.3a}$$
$$\forall\, t > 0,\ \|p(t) - p_{\text{obst}}\| > R_{\text{obst}} + R_{\text{proj}} \tag{A.3b}$$
$$y \leq Y_{\text{screen}} \tag{A.3c}$$
$$u \in \mathcal{U},\ \mathcal{U} = \{u : \underline{u} \leq u \leq \bar{u}\} \tag{A.3d}$$

However the obstacle avoidance equation in (A.3b) makes the inequality constraint non-convex, and this makes the optimization problem very difficult to solve. Therefore, we will relax the non-convex inequality constraint (A.3b) in order to more easily find the optimal solution to the relaxed problem.

Because the projectile game scenarios in the game are setup with only one obstacle, and in a way such that the obstacle is always obstructing the direct line of path such that it is impossible to pass underneath the obstacle and still hit the target. We will increase and transform the avoidance region as shown in Figure A.1. Therefore, trajectory path directly before the obstacle will also be infeasible.

Next we will relax the condition that the trajectory path has to avoid the obstacle for all t, as well as discretizing the obstacle into finite points as shown in Figure A.1. With the discretized coordinates on the edges of the obstacle, now we just need to make sure the trajectory stays above those points. Combining the discretized constraints with the kinematic equations in (A.2),

$$t_{\text{obst},i} = \{t : x(t) = x_{\text{obst},i}\} \qquad i = 1, ..., N \tag{A.4}$$
$$y(t_{\text{obst},i}) \geq y_{\text{obst},i} \qquad i = 1, ..., N \tag{A.5}$$

Even though the new relaxed optimization problem is still non-convex, the relaxation in the non-convex obstacle constraint makes it easier for numerical solvers to find the optimal solutions. Before relaxation, it was difficult for solvers to find a feasible solution in the first place.

With the new formulation, we run the relaxed optimization problem four times to find optimal solutions for the four different cost functions J_1 to J_4, using YALMIP optimization toolbox for Matlab [117], and the solver IPOPT [118]. IPOPT was able to quickly find the feasible and optimal solution.

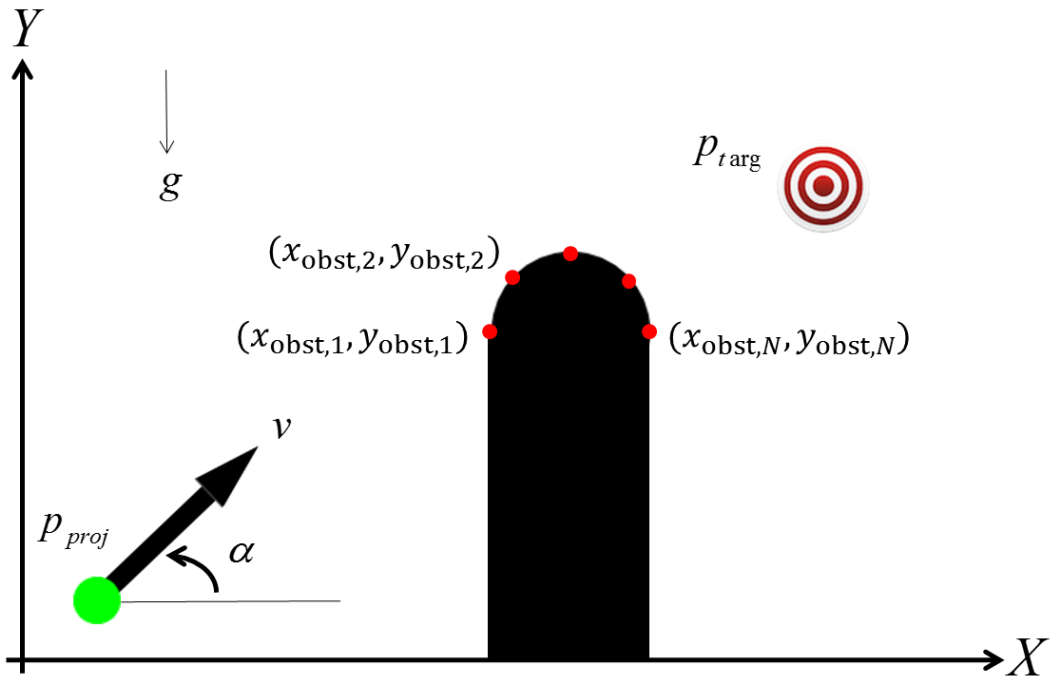


Figure A.1: Relaxed obstacle constraint for the projectile problem.