MODELING HUMAN DRIVING FROM DEMONSTRATIONS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF AERONAUTICS AND ASTRONAUTICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Raunak Bhattacharyya
February 2021

This dissertation is online at: http://purl.stanford.edu/ws309yz7055

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Mykel Kochenderfer, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Dorsa Sadigh

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Mac Schwager

Approved for the Stanford University Committee on Graduate Studies.

Stacey F. Bent, Vice Provost for Graduate Education

This signature page was generated electronically upon submission of this dissertation in electronic format. An original signed hard copy of the signature page is on file in University Archives.

Preface

For autonomous agents to coexist and cooperate with humans, it is important for them to anticipate human behavior. Models of human behavior can be used by autonomous agents to plan in response to human actions and proactively coordinate with humans. Such models can also be used to build simulators for testing autonomous agents by replicating the environment of operation.

Modeling human behavior is challenging because of multiple reasons such as stochasticity, multi-modality, unobservable intents, high dimensional state action spaces, and nonlinear dynamics. In the literature, both ontological and phenomenological approaches have been used to model human behavior. While ontological approaches use rules, phenomenological approaches are data-driven. This thesis presents techniques to model human behavior from demonstrations. The techniques proposed in this thesis are evaluated on their ability to model human driving behavior which is important in autonomous driving for both planning and safety validation.

First, this thesis adapts the technique of Generative Adversarial Imitation Learning to the problem of driver modeling. It extends the GAIL formulation to work in the multi-agent setting where observations gathered from multiple agents are used to inform the training process of a learning agent. The proposed method is shown to better imitate demonstrated driving as opposed to single agent learning method.

Since driving has associated rules, the second part of this thesis introduces a method to provide domain knowledge to the imitation learning agent through reward augmentation. The proposed method, which relies on reward augmentation, is shown to provide better emergent driving performance and overall traffic flow in the recreated traffic simulations.

Many of the applications of autonomous agents are safety-critical including that of autonomous driving. This makes it important for models to be interpretable. This thesis proposes a hybrid rule-based and data-driven method that relies on the technique of particle filtering to learn parameters of underlying rule-based models from human driving demonstrations. The proposed method is demonstrated on the problem of highway merging and shown to generate realistic driving behavior as assessed by a driving Turing test.

Acknowledgments

I want to begin by thanking my advisor, Prof. Mykel Kochenderfer for shaping my thinking: viewing research problems not as problems but as opportunities for learning, developing a love for reading textbooks, realizing that the goal of a Ph.D. is to reach towards the objective truth and thus being unafraid of critiquing my own work, writing the thesis and preparing the defense with a first year graduate student as being the intended target audience in mind, and developing respect and appreciation for students and collaborators. I would also like to thank Mykel for giving me a wealth of opportunities: the opportunity to be a part of an awesome lab like SISL, opportunities to be a teaching assistant and a research mentor, opportunities to travel to conferences and industry workshops to present my research, and to grow both as a researcher and as a person.

Next, I want to thank my reading committee members: Prof. Mac Schwager and Prof. Dorsa Sadigh. I am very grateful to Prof. Schwager for his questions starting right from quals all the way through to my defense, which have always motivated me to deepen my own understanding. Prof. Dorsa Sadigh gave me insights and opportunities to present my work and collaborate with some of her students. I would like to thank my defense committee members Prof. Monroe Kennedy and Prof. Grace Gao for their valuable time and feedback.

My thanks to the Toyota Research Institute (TRI) and Toyota Racing Development (TRD) for their research funding support, and for providing real world problems to motivate my Ph.D. research.

I am extremely thankful to my department administrators Patrick, Robin, Corinna, Van and Jenny for all their help.

I am grateful to my labmates at the Stanford Intelligent Systems lab for creating a welcoming and collaborative atmosphere for learning and research. From them, I have learned many of the skills that are the fundamental to doing research: clear understanding of the underlying mathematics, a process to make writing enjoyable with tangible goals and checkpoints, writing code that can be used by more than just the developer, effective project management to be able to take on larger problems, developing general awareness of what broad research problems are being tackled in the field, and being able to bridge different research worlds.

I am also grateful to the broader Artificial Intelligence and Robotics community at Stanford for the endless opportunities for learning. Through their classes and their insight, Prof. Ben Van Roy, Prof. Marco Pavone, Prof. Tengyu Ma, Prof. Stefano Ermon, Prof. Mac Schwager, Prof. Dorsa Sadigh, and Prof. Mykel Kochenderfer have been instrumental in shaping my understanding of learning and planning for intelligent agents. Stats consulting is a hidden gem which was very helpful in my development of the necessary mathematical skills. Stanford is a phenomenal place for exploration. I am grateful for the opportunity to take excellent classes in History, Sociology, and Tennis, experience various theater performances, be a supporter at competitive sports events, perform music for a play, teach AI to high school students, and make friends from the law school, school of education, and the school of medicine. All these thing combine to make campus life a very full and exciting life indeed.

I was at Georgia Tech for my Master's studies. I am thankful to Prof. Amy Pritchett and my labmates at the Cognitive Engineering Center for their support. I am thankful to my undergrad alma mater, IIT Bombay for starting me out on this journey. My thanks to all my amazing friends from the Bay Area, from Atlanta and from back home in Bombay for their companionship.

Finally, I thank my family for their constant love, encouragement, and support. I am grateful to my father for his unshakeable belief in me, and for always making the time for me to discuss everything. I am grateful to my mother for her unbounded optimism and calm assurance that everything is just going to be alright. I am grateful to both my mother and father for the infinite energy and enthusiasm they bring to everything. They are truly the foundation and the inspiration behind all that I do.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

For autonomous agents to coexist and cooperate with human agents, it is important for them to understand human motion. Anticipation is a key ability for advanced autonomous systems, especially if they operate in densely crowded environments and alongside humans. Prediction plays an important part in human motion analysis: foreseeing how a scene involving multiple agents will unfold over time allows incorporation of this knowledge in a proactive manner, i.e. allowing for enhanced ways of predictive planning, model predictive control, active perception or human-robot interaction. Humans are naturally able to navigate through many social interaction scenarios (e.g. traversing through a dense crowd or negotiating traffic on a highway onramp) because of an inherent ability to reason about other people's actions in terms of their mental states (Gweon and Saxe, 2013). Currently, most autonomous systems do not have such reasoning capabilities which forces them to operate in low-risk roles with minimal human interaction. However, this will need to change with the rising growth of automation in transportation, warehouses, and manufacturing. This thesis presents techniques to model human behavior from demonstrations.

1.1 Human Motion Modeling

Modeling human behavior is challenging because of multiple reasons. First, human behavior is stochastic, i.e., it is inconsistent across settings and different instants

even with all other factors equal. Addressing this inherent stochasticity of human behavior is one of the fundamental challenges in human-robot interaction (HRI). Second, human behavior is multi-modal. Even when the broader intent is known, there are often multiple courses of action that a person may pursue to accomplish their goals. Third, human behavior is influenced by latent factors such as intent and trait that are not directly observable and need to be inferred. Fourth, human behavior is governed by high-dimensional states and nonlinearities in the dynamics. Fifth, the presence and actions of surrounding agents, social relations between agents, social rules and norms influence motion behavior. Sixth, the environment with its geometry and semantics influences motion behavior. Finally, to be effective in practice, motion prediction should be robust and operate in real-time.

Modeling approaches to human motion prediction have been categorized into a taxonomy based on how the models represent human motion: physics-based, pattern-based and planning-based (Rudenko et al., 2018). Physics-based models generate future human motion considering a hand-crafted, explicit dynamical model. They follow a reactive sense-predict scheme. Among the simplest ones are kinematic models that ignore the forces that govern the motion. Popular examples include the constant velocity and constant acceleration models. Dynamic models account for forces that are the key descriptor of motion. However, forces that govern the motion of other agents are not directly observable from sensory data. This makes dynamic models more challenging for motion prediction. Both kinematic and dynamical models ignore the motion of other agents and only make one step ahead predictions.

There are several ways to incorporate local agent interaction models into physics-based approaches for prediction, one popular example being the social force (SF) model (Helbing and Molnar, 1995). Due to the simplicity of these models, they are very useful in simulating large-scale interactions, such as crowd dynamics in panic situations (Helbing et al., 2000) or traffic flow (Treiber et al., 2000). However, these models are sometimes brittle and do not generalize well to novel scenarios because of their reliance on a few parameters.

Pattern-based methods learn human motion behaviors by fitting different function approximators (e.g., hidden Markov models, neural networks, Gaussian processes) to

training data of observed agent trajectories, and follow a sense-learn-predict scheme. Recurrent Neural Networks (RNN) for sequence learning have recently become a widely popular modeling approach for predicting human (Alahi et al., 2016; Vemula et al., 2018) and vehicle (Altché and La Fortelle, 2017; Ding et al., 2019; Park et al., 2018) motion. Other approaches use RNN as models of spatio-temporal graphs for problems that require both spatial and temporal reasoning (Ivanovic et al., 2020; Jain et al., 2016; Vemula et al., 2018). These methods have been used to jointly predict transitions in human crowds (Vemula et al., 2018).

Planning-based approaches solve a sequential decision making problem by explicitly reasoning about the agent's motion goals and follow a sense-reason-predict scheme. By placing an assumption of rationality on the human, the models used to represent human motion must take into account the impact of current actions on the future. Common approaches here are hand-craft cost functions and cost-to-go value estimates. To account for presence of other agents, several authors propose to modify individual optimal policies locally with physics-based methods (Van Den Berg et al., 2008; Wu et al., 2018). Game theoretic approaches model the interaction dynamics by making assumptions on whether the other agent is cooperative (Nikolaidis et al., 2017) or adversarial (M. Wang et al., 2019) and leverage this information for robot planning.

Inverse planning methods assume that the reward or cost function, which depends on social and contextual features and defines the rational behavior, can be learned from observations. Instead of first learning a reward function and then applying planning techniques to generate motion predictions, imitation learning approaches directly extract a policy from the data. This method has been successfully applied to learning human highway driving behavior (Kuefler et al., 2017) and training joint pedestrian motion models (A. Gupta et al., 2018).

1.2 Vehicle Motion Modeling

Autonomous vehicles have the potential to increase the safety and efficiency of the transportation system, improve the quality and productivity of the time spent in

cars, and transform transportation into a utility available to anyone, anytime. This requires advances in many aspects of vehicle autonomy, ranging from vehicle design to control, perception, planning, coordination, and human interaction. Achieving the vision of fully capable automated vehicles will require overcoming many technical, legal, and social challenges (Maurer et al., 2016). One of these challenges is safety validation, which requires reliable models of human driving behavior. The techniques proposed in this thesis are evaluated on their ability to model human driving behavior.

Existing approaches to human driving behavior modeling can be categorized according to multiple considerations (Brown et al., 2020) such as vehicle dynamics and motion prediction paradigm. Vehicle dynamics models describe the input-output behavior of the vehicle. Dynamics models propagate the state of the vehicle forward in response to control inputs. Examples include four-wheel dynamic models, bicycle dynamic models, bicyle kinematic models and unicycle dynamic models. The control inputs are usually acceleration and yaw rate (turn rate).

In applications where low-level vehicle dynamics can be abstracted away, state-transition models are used. Both discrete and probabilistic state-transition models can be used depending on the need to capture the uncertainty in the future state. Some state-transition models are learned, in the sense that the observed correlation between consecutive predicted states results entirely from training on large datasets (Altché and La Fortelle, 2017; M. Bansal et al., 2018; Diehl et al., 2019; Driggs-Campbell et al., 2015; Luo et al., 2018). Such data-driven models come in many forms. Some incorporate an explicit transition model where the parameters are learned, whereas others simply output a full trajectory.

While vehicle dynamics models govern how the vehicle state is propagated in response to control inputs, motion hypothesis generation considers how these control inputs are generated. The motion hypothesis is an estimate of the states of all the agents from the next timestep onward to a prediction horizon. Existing approaches can be grouped according to three paradigms: open-loop independent trajectory prediction, game theoretic prediction, and closed-loop forward simulation. Under the open-loop independent trajectory prediction paradigm, models predict the trajectory independently for each agent in the scene. These approaches are interaction-unaware

because they are open-loop. The simplest motion prediction models in this paradigm are based on various combinations of constant velocity, acceleration, yaw rate or steering angle. More advanced models rely on deep neural networks (Altché and La Fortelle, 2017; Deo et al., 2018; Kim et al., 2017; Krajewski et al., 2019; Yoon and Kum, 2016), Gaussian processes (Armand et al., 2013) and Gaussian mixture models (Wiest et al., 2012). Since these models ignore interaction between agents, their predictive power diminishes with increasing prediction horizon.

Under the game-theoretic paradigm, the predicted motion of some agents is explicitly conditioned on the predicted motion of other agents in the scene (Fisac et al., 2019; González et al., 2019; Isele, 2019; Okuda et al., 2017; Pruekprasert et al., 2019). Thus, agents are modeled as looking ahead to consider the ramifications of their actions. This notion of looking ahead makes game theoretic models more deeply interaction aware than forward simulation models. However, game theoretic models suffer from tractability issues making them unsuitable for building traffic simulators.

In the closed-loop forward simulation paradigm, the model computes a control action for each agent at each time step based on the observations received up to and including that time step, then propagates the entire scene forward in time (Abbeel and Ng, 2004; Bhattacharyya et al., 2020b; Gipps, 1981; Kuefler et al., 2017; Levine and Koltun, 2012; Morton et al., 2016). This process is repeated until the prediction horizon is reached. The closed loop action policy used in forward simulation can be either deterministic, commonly used in motion prediction, or stochastic, commonly used in traffic simulation. Simple examples include heuristic control laws like the Intelligent Driver Model. More complex models include closed loop policies based on neural networks (Kuefler et al., 2017; Morton et al., 2016), dynamic Bayesian networks (Wheeler et al., 2015) and random forests. Since forward simulation uses closed-loop policies, models are nominally interaction-aware. The control action can depend on the actions of other agents via the observation received by the ego vehicle.

1.3 Contributions

This thesis makes the following contributions:

Driver modeling using Generative Adversarial Imitation Learning

In this thesis, we view driving as a sequential decision making problem under uncertainty. Markov Decision Processes (MDPs) are commonly used to model sequential decision making problems under uncertainty. Driving presents two challenges from the perspective of MDP solution techniques. First, the cost function is unknown. Second, the state and action spaces are continuous making tabular solution methods unsuitable. To deal with the first problem, we rely on learning from demonstrations (also known as imitation learning). This approach relies on using demonstration data from expert agents, i.e., human drivers to solve the driving MDP. To tackle the second problem, we use deep neural network driving policies that can work with continuous state action spaces without requiring tabular enumeration.

Prior work to learning driving from demonstrations has used multiple approaches. Behavior cloning treats the problem as supervised learning where a regressor is learned to map states to actions (Pomerleau, 1989). However, since there can never be enough data from all possible states, the approach fails by running into cascading errors. Inverse reinforcement learning approaches have also been tried that try to recover the cost function from the expert demonstrations and then use it to solve the underlying MDP (Abbeel et al., 2008; Abbeel and Ng, 2004; Ng and Russell, 2000). However, these approaches are computationally expensive since they involve a cost recovery step and an MDP solution step. Further, they impose restrictive assumptions on possible cost functions and require hand designed features. In this thesis, we use Generative Adversarial Imitation Learning (GAIL) that bypasses learning the cost function and directly tries to learn a policy from demonstrations. Further, GAIL leverages deep neural networks thereby removing the need for hand designing features. We demonstrate two extensions to GAIL that specifically help modeling vehicle motion from demonstrations. GAIL was originally proposed to work in situations where there is a singe learning agent for instance continuous control tasks in Mujoco (Todorov et al., 2012). However, it does not work well for multiple interacting agents as seen by experiments on driver modeling. This is because the training and test distributions are

different thereby causing a covariate shift problem. To alleviate this problem, this thesis proposes a method based on parameter sharing to learn from demonstrations provided by multiple interacting agents.

The learning agent in GAIL does not explicitly understand the domain of learning. The learning problem is posed in general terms where an underlying MDP is assumed. For structured domains such as driving which have governing rules, we propose Reward Augmented Imitation Learning (RAIL) that provides domain knowledge to inform the learning agent about specific rules of the road.

Hybrid rule-based and data-driven driver modeling

To be able to use our driver models for safety validation of autonomous driving algorithms, we need them to be interpretable. Since neural driving policies do not provide interpretability, we explore the use of rule-based driving policies in the second part of this thesis. Previous approaches to rule-based driving relied on heuristics to set the parameters (Kesting et al., 2007; Treiber et al., 2000). However, these heuristics were brittle and could not explain actual driving behavior well. In this thesis, we propose a methodology for data-driven online parameter estimation of rule-based models. We cast the parameter estimation problem as Bayesian inference over the parameters given the driving demonstration trajectory. We use particle filtering to find distributions over model parameters from driving demonstration data.

For safety validation of autonomous driving in simulation, the learned driver models have to generate realistic traffic simulations. We sample parameters from the learned distributions to obtain driver models, and use them to generate novel scenarios via rollouts. We evaluate their realism using a driving Turing test in highway merging scenarios, where we showed human participants videos of ground truth, and synthetic driving trajectories.

1.4 Overview

This thesis studies the problem of human motion modeling from demonstrations. The problem of human driver behavior modeling has been used as a case study throughout this thesis to demonstrate the techniques. The contributions made by this thesis have been published in various conference proceedings (Bhattacharyya et al., 2019, 2018, 2020a).

Chapter 2 introduces the technique of Generative Adversarial Imitation Learning (GAIL). Building on previous work in imitation learning, GAIL removes the restriction on possible cost function classes which enables attacking real-world problems involving continuous state and action spaces. Further, GAIL allows the use of deep driving policies that do not require hand-crafted feature designing.

Chapter 3 presents an approach that enables GAIL to be scaled up to multi-agent settings and incorporate domain knowledge in the form of reward augmentation. These ideas are demonstrated on the problem of modeling highway driving behavior from real world demonstration data.

Many of the applications of autonomous agents are safety-critical including that of autonomous driving. This makes it important for models to be interpretable.

Chapter 4 proposes a hybrid rule-based and data-driven method that relies on the technique of particle filtering to learn parameters of underlying rule-based models from human driving demonstrations. The method is shown to generate more realistic driving behavior as compared to the black-box imitation learning methods. The proposed method is also demonstrated on the problem of highway merging and shown to generate realistic driving behavior as assessed by a driving Turing test.

Chapter 5 concludes the thesis by providing a summary and identifying directions of future work.

Chapter 2

Generative Adversarial Imitation Learning

Sequential decision making problems, such as driving, are generally formulated using Markov decision processes (MDPs). In order to solve an MDP, a cost function must be specified; however, the cost function may be unknown and difficult to articulate for many real-world problems including driving. In such circumstances, imitation learning, also known as learning from demonstration, is a promising approach (Argall et al., 2009; Schaal, 1999). Imitation learning uses expert demonstrations to learn a policy that behaves similarly to the expert with respect to performance on the unknown cost function.

The supervised learning approach to imitation learning, behavioral cloning (BC), learns a policy by minimizing a loss function over the set of demonstrations with respect to the policy (Pomerleau, 1989). Behavioral cloning trains the policy on the distribution of states encountered by the expert. During testing, however, the policy acts within the environment for long time horizons, and small errors in the learned policy or stochasticity in the environment can cause the agent to encounter a different distribution of states from what it observed during training. This problem, referred to as covariate shift, generally results in the policy making increasingly large errors from which it cannot recover (Ross and Bagnell, 2010; Ross et al., 2011). Behavioral cloning can be effective when a large number of demonstrations are available, but in

many environments it is not possible to obtain sufficient quantities of data.

There are two primary approaches for addressing the covariate shift problem. First, if an expert and environment are available at training time, we can use Dataset Aggregation (DAgger) (Ross et al., 2011). In this work, we do not assume access to an expert. A second class of methods learn a replacement for the cost function that generalizes to unobserved states, allowing the policy to learn from interaction with the environment, and thereby encountering the same distribution of states observed at test time. Inverse reinforcement learning (Ng and Russell, 2000) and apprenticeship learning (Abbeel and Ng, 2004; Ho et al., 2016; Syed and Schapire, 2008) are examples of this second approach. In this work, we adopt a specific definition for apprenticeship learning following that of Ho et al. (2016).

The goal in apprenticeship learning is for an agent to perform no worse than the expert on the true, unknown cost function. Traditional approaches to apprenticeship learning have three primary disadvantages. First, they often fail at imitating the expert as a consequence of restricting the class of cost functions. Second, the class of cost functions is often defined as the span of a set of basis functions that must be defined manually (as opposed to learned from the observations). Third, these methods generally involve running reinforcement learning repeatedly, and have a large computational cost as a result.

Generative Adversarial Imitation Learning (GAIL) (Ho and Ermon, 2016) is a method that tries to address these drawbacks. Using ideas from Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), GAIL removes the restriction that the cost belong to a highly limited class of functions, instead allowing it to be learned using expressive function approximators such as neural networks. Furthermore, using Trust Region Policy Optimization (TRPO) (Schulman et al., 2015), GAIL works with direct policy search as opposed to finding intermediate value functions (Abbeel and Ng, 2004; Ng and Russell, 2000).

In this chapter, we review GAIL, describing its connection to, and advantages over, previous apprenticeship learning approaches. In the following chapter, we apply GAIL to real-world driving data in order to learn models of human driving behavior.

2.1 Preliminaries

An infinite horizon, discounted MDP is defined by the tuple $(\mathcal{S}, \mathcal{A}, P, c, \rho_0, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the transition probability distribution, $c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the cost function, $\rho_0 : \mathcal{S} \to \mathbb{R}$ is the distribution of the initial state s_0, and $\gamma \in (0, 1)$ is the discount factor.

A stochastic policy, $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$, defines the probability of taking each action from each state. The set Π contains all stationary stochastic policies that take actions in $\mathcal{A}$ given states in $\mathcal{S}$. We use π_E to refer to the expert policy. In practice, π_E will only be provided as a set of trajectory samples obtained by executing π_E in the environment.

The expectation with respect to a policy $\pi \in \Pi$ is used to denote an expectation with respect to the trajectory it generates: $\mathbb{E}_\pi[c(s, a)] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t c(s_t, a_t)\right]$ where $s_0 \sim p_0$, $a_t \sim \pi(\cdot \mid s_t)$, and $s_{t+1} \sim P(\cdot \mid s_t, a_t)$ for $t \geq 0$. The γ-discounted causal entropy of the policy π is $H(\pi) = \mathbb{E}_\pi[-\log \pi(a \mid s)]$.

The state-occupancy distribution:

$$\rho_\pi(s) = (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t p(s_t = s \mid \pi), \tag{2.1}$$

gives the average discounted probability of the agent being in state s. The state-action occupancy distribution of a policy π is then defined as $\rho_\pi(s, a) = \pi(a \mid s)\rho_\pi(s)$. This can be interpreted as the distribution of state and actions that an agent encounters when following policy π starting from state $s_0 \sim \rho_0$. The state-action occupancy distribution allows us to write the expected trajectory cost of a policy as

$$\mathbb{E}_\pi[c(s, a)] = \mathbb{E}\left[\sum_{s,a} \rho_\pi(s, a) c(s, a)\right], \tag{2.2}$$

for any cost function c.

2.2 Imitation Learning

The goal of imitation learning (IL) is to learn a policy π that imitates an expert policy π_E given demonstrations from that expert (Ross et al., 2011; Schaal, 1999). A demonstration is defined as a sequence of state-action pairs that result from a policy interacting with the environment: $\tau = \{s_1, a_1, s_2, a_2, \ldots\}$.

Behavioral cloning learns a policy by minimizing some loss function ℓ over the set of demonstrations with respect to the policy (Ross et al., 2011):

$$\pi_{sup} = \operatorname*{argmin}_{\pi} \mathbb{E}_{s \sim \rho_{\pi_E}}[\ell(\pi, s)], \tag{2.3}$$

where ℓ is typically the cross-entropy loss when using discrete actions and the negative log likelihood of a multivariate Gaussian distribution when using continuous actions.

During training, behavioral cloning samples states from the state-occupancy distribution of the expert, ρ_{π_E}. However, when interacting with the environment, the policy samples states from the state-occupancy distribution of the learned policy, $\rho_{\pi_{sup}}$. This change in distribution between training and test time is referred to as covariate shift (Shimodaira, 2000), and results in the agent making increasingly large errors from which it cannot recover.

Allowing the agent to interact with the environment at training time addresses the underlying cause of covariate shift, but this interaction requires an explicit or implicit reward function since the agent may encounter states not contained in the training data. There are various approaches to addressing this problem, which we detail next.

2.3 Apprenticeship Learning

The goal of apprenticeship learning (Abbeel and Ng, 2004) is to find a policy that performs no worse than the expert under the true cost function:

$$\mathbb{E}_{\pi}[c^{\text{true}}(s, a)] \leq \mathbb{E}_{\pi_E}[c^{\text{true}}(s, a)]. \tag{2.4}$$

The problem is that the true cost function c^{true} is unknown. Hence, the desired goal is recast as:

$$\mathbb{E}_{\pi}[c(s,a)] \leq \mathbb{E}_{\pi_E}[c(s,a)],\ \forall c \in \mathcal{C} \tag{2.5}$$

where $\mathcal{C}$ is a restricted class of cost functions. Under the assumption that $c^{\text{true}} \in \mathcal{C}$, if the goal in eq. (2.5) is met, the policy also satisfies the goal established in eq. (2.4).

If we can satisfy eq. (2.5) for the worst possible cost function, i.e., find a policy that performs no worse than the expert on the worst possible cost function in $\mathcal{C}$, we can guarantee that it will perform no worse than the expert on the (unknown) true cost function. Thus, for a given policy π that is yet to be determined, we are interested in finding the worst possible cost function. This is made possible by posing the following optimization problem:

$$c^{\text{worst}}(s,a) = \max_{c \in \mathcal{C}} \mathbb{E}_{\pi}[c(s,a)] - \mathbb{E}_{\pi_E}[c(s,a)]. \tag{2.6}$$

Once the worst-case cost function c^{worst} is known, finding a policy can be posed as the following optimization problem:

$$\pi = \arg\min_{\pi \in \Pi} \mathbb{E}_{\pi}[c^{\text{worst}}(s,a)]. \tag{2.7}$$

The policy found from eq. (2.7) is guaranteed to perform no worse than the expert with respect to the worst-case cost function, and hence guaranteed to perform no worse than the expert on the true cost function c^{true} if $c^{\text{true}} \in \mathcal{C}$.

We can add the expert incurred cost into the objective function without changing the resulting optimum, as follows:

$$\pi = \arg\min_{\pi \in \Pi} \mathbb{E}_{\pi}[c^{\text{worst}}(s,a)] - \mathbb{E}_{\pi_E}[c^{\text{worst}}(s,a)]. \tag{2.8}$$

Since the worst-case cost function is found by solving a maximization problem in eq. (2.6), the overall objective function can be rewritten as:

$$\pi = \arg\min_{\pi \in \Pi} \max_{c \in \mathcal{C}} \mathbb{E}_{\pi}[c(s,a)] - \mathbb{E}_{\pi_E}[c(s,a)]. \tag{2.9}$$

Equation (2.9) establishes a general framework for defining apprenticeship learning algorithms. To use this framework, we must provide: a cost function class $\mathcal{C}$, and an optimization algorithm.

The unknown, true cost function is typically assumed to be a linear combination of known functions that are called basis cost functions. Classic apprenticeship learning algorithms (Abbeel and Ng, 2004; Syed and Schapire, 2008) restrict $\mathcal{C}$ to convex sets given by linear combinations of basis cost functions. However, when the true cost function does not lie within the cost function classes, we lose the guarantee that the learning agent will perform no worse than the expert.

2.4 Generative Adversarial Imitation Learning

Generative Adversarial Imitation Learning is derived from an alternative approach to imitation learning called Maximum Causal Entropy IRL (MaxEntIRL) (Bloem and Bambos, 2014; Ziebart et al., 2008). While apprenticeship learning attempts to find a policy that performs at least as well as the expert across cost functions, MaxEntIRL seeks a cost function for which the expert is uniquely optimal. This latter objective turns out to be equivalent, under certain assumptions, to finding a policy with an occupancy distribution matching that of the expert. This section describes the derivation of this connection, the resulting imitation learning algorithm, and its connection with Generative Adversarial Networks.

2.4.1 Derivation of GAIL

GAIL is derived from a cost-regularized MaxEntIRL objective (Ziebart et al., 2010):

$$\mathrm{IRL}_\psi(\pi_E) = \arg\max_{c \in \mathcal{C}} -\psi(c) + \left(\min_{\pi \in \Pi} -\mathcal{H}(\pi) + \mathbb{E}_\pi[c(s,a)] \right) - \mathbb{E}_{\pi_E}[c(s,a)]. \quad (2.10)$$

where $\mathcal{H}(\pi) \equiv \mathbb{E}_\pi[-\log \pi(a \mid s)]$ is the discounted causal entropy of the policy taken with respect to the state-action distribution of the policy, and $\psi : \mathcal{C} \to \mathbb{R}^*$ is a function assigning a value in the extended reals to each cost function c. The regularization

function, ψ, plays an important role in the derivation of GAIL and in its connection with the apprenticeship learning methods of section 2.3. Specifically, Ho and Ermon (2016) characterize the result of running reinforcement learning on a cost output from MaxEntIRL:

$$\mathrm{RL} \circ \mathrm{IRL}_{\psi}(\pi_E) = \arg\min_{\pi \in \Pi} -\mathcal{H}(\pi) + \psi^*(\rho_\pi - \rho_{\pi_E}). \tag{2.11}$$

Here, $\psi^*(\rho_\pi - \rho_{\pi_E}) \equiv \sup_{c \in \mathcal{C}} (\rho_\pi - \rho_{\pi_E})^T c - \psi(c)$ denotes the convex conjugate of ψ, which attempts to find a cost function that places high cost on state-action pairs more frequently visited by π than by π_E. As a result, minimizing with respect to π attempts to match the occupancy distributions of the two policies.

Ho and Ermon (2016) show that different cost function regularizers result in different imitation learning algorithms. For example, they show (under assumptions) that when ψ is constant across cost functions, this results in exact occupancy distribution matching. This is accomplished by showing that MaxEntIRL is dual to the following optimization problem:

$$\min_{\rho \in \mathcal{D}} -\bar{H}(\rho) \text{ subject to } \rho(s,a) \quad = \rho_E(s,a),\ \forall s \in \mathcal{S}, a \in \mathcal{A} \tag{2.12}$$

where $\bar{H}$ denotes the entropy of the occupancy distribution. Solving this optimization problem is intractable for large or continuous MDPs because it involves satisfying a constraint for each point in $\mathcal{S} \times \mathcal{A}$, many of which will require ρ_π to be zero due to the limited size of the dataset of expert demonstrations.

An alternative setting of the cost function regularizer results in the apprenticeship learning algorithms from section 2.3. Let $\psi(c) = \delta_{\mathcal{C}}(c)$ where $\delta_{\mathcal{C}}(c) = 0$ if $c \in \mathcal{C}$ and ∞ otherwise, for a restricted class of cost functions $\mathcal{C}$. This results in eq. (2.11) reducing

to (entropy-regularized) apprenticeship learning as follows:

$$\pi^{\text{apprenticeship}} = \arg\min_{\pi \in \Pi} -\mathcal{H}(\pi) + \psi^*(\rho_\pi - \rho_{\pi_E}) \tag{2.13}$$

$$= \arg\min_{\pi \in \Pi} -\mathcal{H}(\pi) + \delta^*_{\mathcal{C}}(\rho_\pi - \rho_{\pi_E}) \tag{2.14}$$

$$= \arg\min_{\pi \in \Pi} -\mathcal{H}(\pi) + \max_{c \in \mathcal{C}} -\delta_{\mathcal{C}}(c) + \sum_{s,a} c(s,a)\big(\rho_\pi(s,a) - \rho_{\pi_E}(s,a)\big) \tag{2.15}$$

$$= \arg\min_{\pi \in \Pi} -\mathcal{H}(\pi) + \max_{c \in \mathcal{C}} \mathbb{E}_\pi[c(s,a)] - \mathbb{E}_{\pi_E}[c(s,a)] \tag{2.16}$$

$$\tag{2.17}$$

This regularizer restricts the cost function to $\mathcal{C}$, which is traditionally taken to be a small subspace spanned by finitely many basis cost functions. From eq. (2.11) we see that the information contained in the expert policy (or demonstrations sampled using that policy) must be encoded in the cost function. When the "true" cost function is not in this space, information about the expert policy can be lost, which partially explains why traditional apprenticeship learning algorithms can fail to imitate the expert well.

Given that the desire is for an imitation learning algorithm that can: 1) scale to large state action spaces to work for practical problems, and 2) can allow for imitation without restricting cost functions to lie in a small subspace of finitely many linear basis cost functions, GAIL proposes a new cost function regularizer ψ_{GA}. This regularizer allows scaling to large state action spaces and removes the requirement to specify basis cost functions. While existing apprenticeship learning formalisms used the cost function as the descriptor of desirable behavior, GAIL relies instead on the divergence between the demonstration occupancy distribution and the learning agent's occupancy distribution. The subsequent discussion will derive the form of ψ_{GA} and establish its connection to GANs.

2.4.2 Connection to Generative Adversarial Networks

Equation (2.11) establishes an optimization problem formulation for the imitation learning problem via the convex conjugate of the cost function regularizer. A cost function regularizer $\psi(c)$ needs to be instantiated to reach an imitation learning algorithm. Consider the binary classification problem of classifying state-action pairs (s, a) that have been drawn from the expert occupancy distribution ρ_{π_E} or the learning agent's occupancy distribution ρ_π. For this binary classification task, assuming a loss function ϕ to score the training examples, the minimum expected risk is defined as:

$$R_\phi(\pi, \pi_E) = \sum_{s,a} \min_{\gamma \in \mathbb{R}} \rho_\pi(s, a)\phi(\gamma) + \rho_{\pi_E}(s, a)\phi(-\gamma). \tag{2.18}$$

Proposition A.1 of Ho and Ermon, 2016 shows that this minimum expected risk is connected to the convex conjugate of the cost function regularizer as

$$\psi^*_\phi(\rho_\pi - \rho_{\pi_E}) = -R_\phi(\rho_\pi, \rho_{\pi_E}). \tag{2.19}$$

This allows us to build a bridge from binary classification to imitation learning. The logistic loss function $\phi(x) = \log(1 + e^{-x})$ is chosen. In this case, using the logistic loss, the connection is (as shown by corollary A.1.1 from Ho and Ermon, 2016):

$$\begin{aligned}
\psi^*_{GA}(\rho_\pi - \rho_{\pi_E}) &= -R_\phi(\rho_\pi, \rho_{\pi_E}) && (2.20)\\
&= \sum_{s,a} \max_{\gamma \in \mathbb{R}} \rho_\pi(s, a)\big(-\phi(\gamma)\big) + \rho_{\pi_E}(s, a)\big(-\phi(-\gamma)\big) && (2.21)\\
&= \sum_{s,a} \max_{\gamma \in \mathbb{R}} \rho_\pi(s, a) \log\left(\frac{1}{1 + e^{-\gamma}}\right) + \rho_{\pi_E}(s, a) \log\left(\frac{1}{1 + e^{\gamma}}\right) && (2.22)\\
&= \sum_{s,a} \max_{\gamma \in \mathbb{R}} \rho_\pi(s, a) \log\big(\sigma(\gamma)\big) + \rho_{\pi_E}(s, a) \log\big(1 - \sigma(\gamma)\big) && (2.23)\\
&= \sum_{s,a} \max_{d \in (0,1)} \rho_\pi(s, a) \log d + \rho_{\pi_E}(s, a) \log(1 - d) && (2.24)\\
&= \max_D \sum_{s,a} \rho_\pi(s, a) \log(D(s, a)) + \rho_{\pi_E}(s, a) \log(1 - D(s, a)). && (2.25)
\end{aligned}$$

If we use the result from eq. (2.20) to substitute the expression for the convex conjugate of the cost regularizer into our central optimization objective eq. (2.11), we obtain the GAIL objective function:

$$\min_{\pi} \max_{D} \sum_{s,a} \rho_{\pi}(s,a) \log(D(s,a)) + \rho_{\pi_E}(s,a) \log(1 - D(s,a)). \tag{2.26}$$

This optimization objective established in eq. (2.26) provides a connection to GANs (Goodfellow et al., 2014). In GANs, the goal is to model the distribution $p_{\text{data}}(x)$. The generative modeling objective is formulated as

$$\min_{G} \max_{D} \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]. \tag{2.27}$$

Here, G is the generator that maps input noise variables z to the data space as $G(z)$ and D is the discriminator which outputs a single scalar $D(x)$ that represents the probability that x came from the data rather than p_g, a binary classification task. This objective is solved using simultaneous gradient descent wherein the parameters of G and D are updated. This is accomplished by sampling two sets of data, one from the training samples and the other from the noise prior.

Unlike GANs, GAIL considers the environment as a black box, and thus the objective is not differentiable with respect to the parameters of the policy. Therefore, simultaneous gradient descent is not suitable for solving the GAIL optimization objective. Instead, optimization over the GAIL objective is performed by alternating between a gradient step to increase eq. (2.26) with respect to the discriminator parameters D, and a Trust Region Policy Optimization (TRPO) step (Schulman et al., 2015) to decrease eq. (2.26) with respect to the parameters θ of the policy π_θ.

TRPO is a model-free reinforcement learning algorithm. It does not have explicit access to a model of the environment, i.e, a function which predicts the state transitions or rewards. Model-free RL algorithms can be broadly categorized into those that learn Q-values and those that directly learn policies. TRPO belongs to the latter family, known as policy optimization.

2.4.3 Policy Optimization

Policy optimization directly search in the policy space by optimizing the policy parameters by either gradient ascent on the performance objective, or indirectly by maximizing local approximations to the performance objective. This optimization is almost always performed on-policy, meaning each update only uses data collected according to the most recent version of the policy.

Since the goal is to model human behavior which is stochastic, we have a stochastic policy that provides a probability distribution given a state and is denoted by $\pi(a \mid s)$. We write the policy as $\pi_\theta(a \mid s)$ when it is parametrized by parameter vector θ. Here, we are in the premise of learning from obervations where trajectories of expert demonstrations are provided. Let $\tau = (s_0, a_0, s_1, a_1, ..., s_T)$ denote a trajectory, which is a sequence of states and actions generated by a policy π_θ interacting with the environment. Let $p(\tau \mid \theta)$ be the probability of the trajectory τ, and let $R(\tau)$ denote the total reward of the trajectory.

In an MDP, the goal is to maximize the expected return using a stochastic, parametrized policy yielding the objective:

$$\max_\theta J(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)]. \tag{2.28}$$

The policy gradient approach approaches this objective by performing a gradient ascent on the policy parameters as follows:

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\pi_\theta)\Big|_{\theta_k}, \tag{2.29}$$

where k denotes the current iterate of the policy parameters and $k+1$ denotes the new parameters after taking a gradient step.

The expression for the gradient is given by

$$\nabla_\theta J(\pi_\theta) = \nabla_\theta \, \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)]. \tag{2.30}$$

Expressing the expectation in the integral form we have:

$$\nabla_\theta J(\pi_\theta) = \nabla_\theta \int_\tau p(\tau \mid \theta) R(\tau) \tag{2.31}$$

$$= \int_\tau \nabla_\theta p(\tau \mid \theta) R(\tau). \tag{2.32}$$

The derivative of the probability of a trajectory can be expressed as follows (known as the log derivative trick):

$$\nabla_\theta p(\tau \mid \theta) = p(\tau \mid \theta) \nabla_\theta \log p(\tau \mid \theta). \tag{2.33}$$

The probability of a trajectory τ is

$$p(\tau \mid \theta) = \rho_0(s_0) \Pi_{t=0}^{T} p(s_{t+1} \mid s_t, a_t) \pi_\theta(a_t \mid s_t). \tag{2.34}$$

The log probability of the trajectory is therefore,

$$\log p(\tau \mid \theta) = \log \rho_0(s_0) + \sum_{t=0}^{T} \big(\log p(s_{t+1} \mid s_t, a_t) + \log \pi_\theta(a_t \mid s_t) \big). \tag{2.35}$$

Taking the gradient of this expression, and observing that some of the terms do not depend on the parameter θ, we obtain the gradient of the log probability of the trajectory to be:

$$\nabla_\theta \log p(\tau \mid \theta) = \cancel{\nabla_\theta \log \rho_0(s_0)} + \sum_{t=0}^{T} \big(\cancel{\nabla_\theta \log p(s_{t+1} \mid s_t, a_t)} + \nabla_\theta \log \pi_\theta(a_t \mid s_t) \big) \tag{2.36}$$

$$= \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t). \tag{2.37}$$

Putting all of the above together, we can derive an expression for the gradient of the policy with respect to the parameter θ as follows:

$$\nabla_\theta J(\pi_\theta) = \nabla_\theta \, \mathbb{E}_{\tau \sim \pi_\theta}[R(\tau)] \tag{2.38}$$

$$= \int_\tau \nabla_\theta p(\tau \mid \theta) R(\tau) \quad \text{Expand expectation} \tag{2.39}$$

$$= p(\tau \mid \theta) \nabla_\theta \log p(\tau \mid \theta) R(\tau) \quad \text{Log derivative trick} \tag{2.40}$$

$$= \mathbb{E}_{\tau \sim \pi_\theta}[\nabla_\theta \log p(\tau \mid \theta) R(\tau)] \quad \text{Return to expectation form} \tag{2.41}$$

$$= \mathbb{E}_{\tau \sim \pi_\theta}[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) R(\tau)] \quad \text{Gradient of log probability} \tag{2.42}$$

GAIL can also be derived more directly from a f-divergence minimization perspective (Ghasemipour et al., 2020), which is less general than cost-regularized MaxEntIRL. The f-divergence framework does not allow for minimizing certain distances between occupancy distributions, for example the Wasserstein distance, which has been shown to result in more reliable training of GANs (Arjovsky et al., 2017). However, the Wasserstein distance can be used within the cost-regularized MaxEntIRL framework (Xiao et al., 2019), and we use this version of GAIL in the multi-agent setting.

2.5 Extension to Multiple Agents

For safety validation in simulation, it is crucial to simulate the behavior of not just a single vehicle, but entire traffic scenes to be able to recreate driving behavior arising out of interaction between agents.

The task of simultaneously controlling multiple vehicles operating on a single roadway can be viewed as a multiagent control problem. However, when evaluated in a multi-agent setting, the policies learned through single-agent imitation learning fail to exhibit realistic behavior, rendering them inadequate for use in simulation. During test time, the policy learned from a single agent's experience observes nearby vehicles acting differently than during training, and again makes small errors that compound over time.

This motivated the development of Parameter-Sharing GAIL (PS-GAIL) (Bhattacharyya et al., 2018), which enables scaling of the imitation learning approach to multiple agents. In line with recent work in multi-agent imitation learning (Gruver et al., 2020; Song et al., 2018; Yu et al., 2019), we formulate multi-agent driving as a Markov game (Littman, 1994) consisting of M agents and an unknown reward function. We make three simplifying assumptions:

1. Homogeneous agents: agents have the same observation and action spaces:

$$\mathcal{O}_i = \mathcal{O}_j \text{ and } \mathcal{A}_i = \mathcal{A}_j \ \forall \text{ agents } i, j.$$

2. Identical cost function: the (unknown) cost function is the same for all agents:

$$\mathcal{R}_i = \mathcal{R}_j \ \forall \text{ agents } i, j.$$

3. Cooperative agents: agents have common behavior such as actions, domain knowledge, and goals (an example is avoiding collision). In particular, drivers are not adversarial (as in the case of racing).

These assumptions are idealizations and do not hold for real-world driving scenes. For example, different vehicles may permit different accelerations, a driver may only want to change lanes if other drivers are not doing so, and individuals may value different driving qualities such as smoothness or proximity to other vehicles differently. Nevertheless, these assumptions often do apply approximately, and, as we later show, allow for learning of realistic driving policies.

A naive approach to learning human driver policies would be to train a policy in an environment where it controls a single vehicle on the roadway and all remaining vehicles follow a predetermined trajectory. Unfortunately, this approach is often incapable of producing policies that can reliably control many vehicles on the same roadway. By introducing such a controller to other vehicles after training, we reintroduce covariate shift. As a result, small errors in the behavior of a single vehicle can destabilize neighboring vehicles, ultimately leading to the failure of many agents in the scene.

Existing training schemes for reinforcement learning that are applicable to multi-agent domains are centralized learning (Foerster et al., 2016; Sukhbaatar et al., 2016) and concurrent learning (Diallo et al., 2017; Palmer et al., 2018). Centralized training learns a single policy that controls every agent by mapping a joint observation to a joint action over all the agents. However, this quickly runs into scalability issues with increasing number of agents due to exponential growth in the observation and action spaces. In the concurrent training approach, every agent learns its own policy. This addresses the growth of the state and action spaces, but still scales poorly because the number of parameters to be learned grows with the number of agents.

Centralized learning and decentralized execution is an approach that finds the middle ground between centralized and concurrent approaches to learning in multi-agent domains. Parameter sharing is an example of this approach that has shown recent promise due to more centralization during the learning process (Terry et al., 2020). Further, direct policy search based methods have shown to perform better than value based methods in deep reinforcement learning problems involving continuous action spaces (Nguyen et al., 2020), such as driving.

Combining the benefits of parameter sharing with direct policy search, J. Gupta et al. (2017) introduced an algorithm called Parameter Sharing Trust Region Policy Optimization (PS-TRPO), which is a policy gradient approach that combines parameter sharing and TRPO. PS-TRPO was shown to produce decentralized parameter-sharing neural network policies that exhibit emergent cooperative behavior without explicit communication between agents. PS-TRPO is highly sample-efficient because it reduces the number of parameters by a factor of M, and shares experience across all agents. Furthermore, it mitigates issues resulting from non-stationary learning dynamics by collecting a new batch of data for each round of policy optimization. Notably, PS-TRPO still allows agents to exhibit different behavior because each agent receives unique observations.

For a policy π_θ with parameters θ, PS-TRPO performs an update to the policy parameters by approximately solving the constrained optimization problem:

$$\underset{\theta}{\text{maximize}} \quad \mathbb{E}_{o,a\sim\pi_{\theta_k}}\left[\frac{\pi_\theta(a \mid o)}{\pi_{\theta_k}(a \mid o)} A_{\theta_k}(o,a)\right] \tag{2.43}$$

$$\text{subject to} \quad \mathbb{E}_o\left[D_{KL}(\pi_{\theta_k}(\cdot \mid o) \| \pi_\theta(\cdot \mid o))\right] \leq \Delta_{KL}, \tag{2.44}$$

where π_{θ_k} is a rollout-sampling policy, and $A_{\theta_k}(o,a)$ is an advantage function quantifying how much the value of an action a taken in response to an observation o differs from the baseline value estimated for o. D_{KL} is the KL-divergence between the two policy distributions, and Δ_{KL} is a step size parameter that controls the maximum change in policy per optimization step.

Our proposed approach, PS-GAIL, combines GAIL with PS-TRPO to generate policies capable of controlling multiple vehicles, enabling more stable simulation of entire road scenes. The approach is described in algorithm 1. We begin by initializing the shared policy parameters and select a step size parameter. At each iteration, the shared policy is used by each agent to generate trajectories. Rewards are then assigned to each state-action pair in these trajectories by the critic. Subsequently, observed trajectories are used to perform a TRPO (Schulman et al., 2015) update for the policy, and an Adam (Kingma and Ba, 2014) update for the critic. PS-GAIL can be viewed as a special case of the algorithms presented by Song et al. (Song et al., 2018) since all agents share the same policy and receive rewards from the same critic.

Algorithm 1 PS-GAIL

Input: Expert trajectories $\tau_E \sim \pi_E$, Shared policy parameters Θ_0, Discriminator parameters ψ_0, Trust region size Δ_{KL}, Curriculum distribution $\mathcal{C}$
for $k \leftarrow 0, 1, \ldots$
  Sample number of agents from curriculum $m \sim \mathcal{C}(k)$
  Rollout trajectories for all m agents $\vec{\tau} \sim \pi_{\theta_k}$
  Score $\vec{\tau}$ with critic, generating reward $\tilde{r}(s_t, a_t; \psi_k)$
  Batch trajectories obtained from all m agents
  Take a TRPO step to find $\pi_{\theta_{k+1}}$ maximizing eq. (2.43)
  Update the critic parameters ψ by maximizing eq. (2.26)

2.6 Discussion

This chapter presented the idea of imitation learning, a method that uses expert demonstrations to obtain policies for MDPs which lack associated cost functions. First, this chapter reviewed existing approaches to imitation learning such as behavior cloning and apprenticeship learning. Next, this chapter examined Generative Adversarial Imitation Learning, a approach that enables imitation learning for real world problems. Specifically, this chapter highlighted the connection between GAIL and GANs, and illustrated how the GAIL framework is a more general framework that subsumes existing apprenticeship learning approaches. However, while GAIL was originally proposed for a single learning agent, many real world problems including driving involve multiple interacting agents. This chapter proposed a multi-agent extension to GAIL called PS-GAIL that shall be demonstrated in the next chapter for modeling multiple drivers in highway driving.

Chapter 3

Generative Modeling of Driver Behavior

This chapter demonstrates the GAIL methodology to model the behavior of human drivers from driving demonstration data. Reliable models of human driving behavior are important for building simulation platforms for validating the safety of autonomous driving algorithms. The autonomous driving literature has established that it is infeasible to build a statistically significant case for the safety of a system solely through real-world testing (Koopman and Wagner, 2016). Validation through simulation is an alternative to real-world testing, with the ability to evaluate vehicle performance in large numbers of scenes quickly, safely, and economically (Morton et al., 2018). In such safety critical settings, representative models of human driving behavior are essential in the validation of autonomous driving systems.

Driver modeling falls within the paradigm of learning from demonstrations. There is no reason to assume that the cost function of the human drivers lies within a small function class. Instead, the cost function could be quite complex, which makes GAIL a suitable choice for driver modeling. Human driving situations are inherently multi-agent in nature. Typical human driving scenes are composed of several vehicles that interact to exhibit emergent patterns of traffic behavior that cannot be easily predicted from the properties of the individual vehicles alone. For example, given two very similar initial scenes, the vehicles can reach very different configurations after

just a few seconds because small changes in states can quickly compound into large differences in the resulting vehicle behaviors and motion pattern. Reliable human driver models must be capable of imitating these emergent properties of traffic behavior.

We frame highway driving as a sequential decision-making task in which the driver obeys a stochastic policy $\pi(a \mid s)$, mapping observed road conditions s to a distribution over driving actions a. The state space represents the driving scene, the actions are driving actions, and the transition model is governed by the vehicle dynamics and the actions taken by surrounding vehicles. However, the cost function of the MDP is unspecified because it is often difficult for humans to articulate, let alone mathematically formulate, the cost function that they are following while driving. Given a class of policies π_θ parameterized by θ, we seek to find the policy that best recreates human driving behavior. The goal is to infer this policy from a dataset consisting of a sequence of state-action tuples (s_t, a_t).

3.1 Dataset

We use the public Next-Generation Simulation (NGSIM) dataset for US Highway 101 (Colyar and Halkias, 2007). NGSIM provides 45 minutes of driving at 10 Hz. The US Highway 101 dataset covers an area in Los Angeles approximately 640 m in length with five mainline lanes and a sixth auxiliary lane for highway entrance and exit.

Traffic density in the dataset transitions from uncongested to full congestion and exhibits a high degree of vehicle interaction as vehicles merge on and off the highway and must navigate in congested flow. The diversity of driving conditions and the forced interaction of traffic participants makes these sources particularly useful for behavioral studies. The trajectories were smoothed using an extended Kalman filter on a bicycle model and projected to lanes using centerlines extracted from the NGSIM roadway geometry files. Cars, trucks, buses, and motorcycles are in the dataset, but only car trajectories were used for model training.

This dataset is split into three consecutive 15 min sections of driving data that

represent different vehicle densities and traffic conditions. We use the first section as the training dataset, from which we learn our policies. The remaining two sections are used for testing and evaluating the quality of the resulting policies. This allows us to assess the generalization capability of the learned driving policies.

3.2 Simulator

In order to learn the policy in an environment with human drivers, we use a simulator that allows for playing back real trajectories and simulating the movement of controlled vehicles given actions selected by a policy. The process proceeds as follows:

1. The initial scene is sampled from a dataset of real driver trajectories. This state includes the position, orientation, and velocity of all vehicles in the scene.

2. A subset of the vehicles in the scene are randomly selected to be controlled by the policy. For single-agent training only one vehicle is selected, whereas for multi-agent training M vehicles are controlled by the policy.

3. For each vehicle, a set of features are extracted and passed to the policy as the observation. Table 3.1 describes the features provided to the policy. These features represent the scene information, and thus act as observations of the state of the driving MDP.

4. At every timestep, the policy outputs longitudinal acceleration and turn-rate values as the vehicle action in response to the observed features. These values are used to propagate the vehicle forward in time according to the vehicle dynamics.

5. The simulation is carried out, and associated metrics of both imitation performance and driving performance are extracted.

Table 3.1: Observation features

Feature	Description
Ego Vehicle	Lane-relative velocity, heading, offset. Vehicle length and width. Longitudinal and lateral acceleration. local and global turn and angular rate. Lane curvature, distance to left and right lane makers and road edges.
Leading Vehicle	Relative distance, velocity, and absolute acceleration of vehicle of fore vehicle, if it exists.
LIDAR Range and Range Rate	20 artificial LIDAR beams output in regular polar intervals, providing the relative position and velocity of intercepted objects.
Temporal	Timegap and time-to-collision.
Indicators	Collision occurring, ego vehicle out-of-lane, and negative velocity.

3.3 Policy Representation

Our learned policy must be able to simulate human driving behavior, which involves:

- *Non-linearity* in the desired mapping from states to actions (e.g., large corrections in steering to avoid collisions caused by small changes in the current state).
- *High-dimensionality* of the state representation, which must describe properties of the ego-vehicle, in addition to surrounding cars and road conditions.
- *Stochasticity* because humans may take different actions each time they encounter a given traffic scene.

To address the first and second points, we represent all learned policies π_θ using neural networks. Neural networks have gained widespread popularity due to their

ability to learn robust hierarchical features from complicated inputs (Krizhevsky et al., 2012; H. Lee et al., 2009), and have been used in automotive behavioral modeling for action prediction in car-following contexts (Hongfei et al., 2003; Khodayari et al., 2012; Lefèvre et al., 2014; Morton et al., 2016; Panwai and Dia, 2007), lateral position prediction (Q. Liu et al., 2014), and maneuver classification (Boyraz et al., 2007).

To addresss the third point, we interpret the network's real-valued outputs given input s_t as the mean μ_t and logarithm of the diagonal covariance $\log \nu_t$ of a Gaussian distribution. This enables stochasticity in the driving action provided by the neural network policy in response to a particular driving scene. Actions are chosen by sampling $a_t \sim \pi_\theta(a_t \mid s_t)$.

We evaluate both feedforward and recurrent network architectures. Feedforward neural networks directly map inputs to outputs. The most common architecture, multilayer perceptrons (MLPs), consist of alternating layers of tunable weights and element-wise nonlinearities. However, the feedforward MLP is limited in its ability to adequately address partially observable environments. In real world driving, sensor error and occlusions may prevent the driver from seeing all relevant parts of the driving state. By maintaining sufficient statistics of past observations in memory, recurrent policies (Wierstra et al., 2010) disambiguate perceptually similar states by acting with respect to histories of, rather than individual, observations. In this work, we represent recurrent policies using Gated Recurrent Unit (GRU) networks (Cho et al., 2014) due to their comparable performance with fewer parameters than other architectures.

We use similar architectures for the feedforward and recurrent policies. The recurrent policies consist of five feedforward layers that decrease in size from 256 to 32 neurons, with an additional GRU layer consisting of 32 neurons. Exponential linear units (ELU) were used throughout the network, which have been shown to combat the vanishing gradient problem while supporting a zero-centered distribution of activation vectors. The MLP policies have the same architecture, except the GRU layer is replaced with an additional feedforward layer. For each network architecture, one policy is trained through BC and one policy is trained through GAIL. In all, we trained four neural network policies: GAIL GRU, GAIL MLP, BC GRU, and BC

MLP.

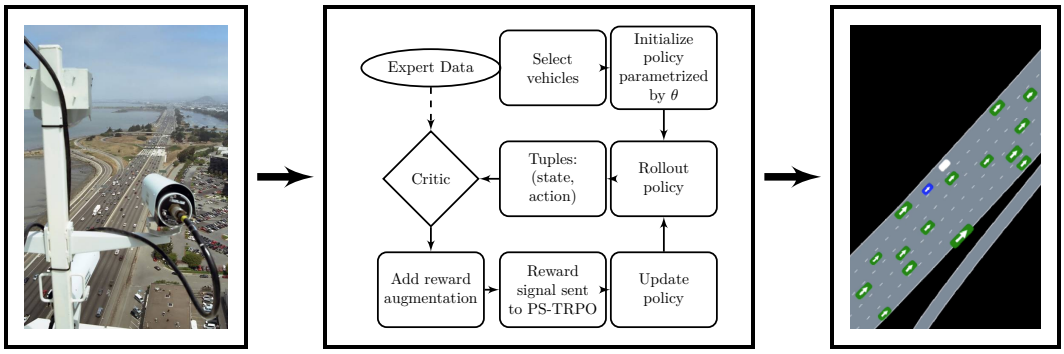

Figure 3.1: The imitation learning pipeline: the driving demonstration trajectories (left panel) are fed into our imitation learning module (middle panel) to create driving policies that can be used for validation of autonomous vehicles in simulation (right panel).

3.4 Metrics

Section 3.3 shows the imitation learning pipeline starting from driving demonstration data to driving policies. We assess the imitation performance of our driving policies via different metrics. First, to measure imitation of local vehicle behaviors, we use a set of Root Mean Square Error (RMSE) metrics that quantify the distance between the real trajectories in the dataset and the trajectories generated by our learned driving policies. We calculate the RMSE between the original human driven vehicle and its replacement policy driven vehicle in terms of the position, speed, and lane offset.

Second, to assess the undesirable traffic phenomena that arise out of vehicular interactions as compared to local, single vehicle imitation, we extract metrics that quantify collisions, hard-braking, and offroad driving. We also extract these metrics of undesirable traffic phenomena for the NGSIM driving data and compare them against the metrics obtained from rollouts generated by our driving policies.

3.5 Single Agent Imitation

First, we report results obtained from experiments conducted on learning driving from a single agent (Kuefler et al., 2017). Here, one vehicle is randomly sampled from the NGSIM demonstration data and its trajectory is used to train the critic. The effectiveness of the resulting driving policy trained using GAIL in imitating human driving behavior is assessed by validation in rollouts conducted on the simulator described in section 3.2. The resulting driving behavior was compared against various driver modeling baselines using the metrics discussed in section 3.4.

The first baseline is a static Gaussian (SG) model, which is an unchanging Gaussian distribution $\pi(a \mid s) = \mathcal{N}(a \mid \mu, \Sigma)$ fit using maximum likelihood estimation on the demonstration data. The second baseline model is a Behavioral Cloning (BC) approach using mixture regression (MR) (Lefèvre et al., 2014). The model has been used for model-predictive control and has been shown to work well in simulation and in real-world drive tests. Our MR model is a Gaussian mixture over the joint space of the actions and features, trained using Expectation Maximization. The stochastic policy is formed from the weighted combination of the Gaussian components conditioned on the features. Greedy feature selection is used during training to select a subset of predictors up to a maximum feature count threshold while minimizing the Bayesian information criterion (Schwarz, 1978).

The final baseline model uses a rule-based controller to govern the lateral and longitudinal motion of the ego vehicle. The longitudinal motion is controlled by the Intelligent Driver Model (Treiber et al., 2000). The inputs to the model are the vehicle's current speed $v(t)$ at time t, relative speed $r(t)$ with respect to the leading vehicle, and distance headway $d(t)$. The model then outputs an acceleration according to

$$a_{\text{IDM}} = a_{\max}\left(1 - \left(\frac{v(t)}{v_{\text{des}}}\right)^4 - \left(\frac{d_{\text{des}}}{d(t)}\right)^2\right), \tag{3.1}$$

where the desired distance is

$$d_{\text{des}} = d_{\min} + \tau.v(t) - \frac{v(t) \cdot r(t)}{2\sqrt{a_{\max} \cdot b_{\text{pref}}}}. \tag{3.2}$$

The model has several parameters that determine the acceleration output based on the scene information. Here, v_{des} refers to the free speed velocity, d_{min} refers to the minimum allowable separation between the ego and leader vehicle, τ refers to the minimum time separation allowable between ego and leader vehicle, a_{max} and b_{pref} refer to the limits on the acceleration and deceleration, respectively.

For the lateral motion, MOBIL (Kesting et al., 2007) is used to select the desired lane, with a proportional controller used to track the lane centerline. A small amount of noise is added to both the lateral and longitudinal accelerations to make the controller nondeterministic.

To extract the metrics of driving performance, the ego vehicle is driven using a bicycle model with acceleration and turn rate sampled from the policy network. Figure 3.2 shows the discrepancy between rollouts and the ground truth demonstration through root mean square error metrics. The RMSE results show that the BC models have competitive short-horizon performance, but accumulate error over longer time horizons. GAIL produces more stable trajectories and its short term predictions perform well.

Figure 3.3 shows the undesirable driving metrics obtained from simulation. The GAIL policies outperform the BC policies. Compared to BC, the GAIL GRU policy has the closest match to the data everywhere except for hard brakes, as it rarely takes extreme actions. Mixture regression largely performs better than SG and is on par with the BC policies, but is still susceptible to cascading errors. Offroad duration is perhaps the most striking statistic; only GAIL (and of course IDM + MOBIL) stay on the road for extended stretches. SG never brakes hard as it only drives straight, causing many collisions as a consequence. It is interesting that the collision rate for IDM + MOBIL is roughly the same as the collision rate for GAIL GRU, despite the fact that IDM + MOBIL should not collide. The inability of other vehicles within the simulation environment to fully react to the ego-vehicle may explain this phenomenon.

The results demonstrate that GAIL-based models capture many desirable properties of both rule-based and machine learning methods, while avoiding common pitfalls. With the exception of the rule-based controller, GAIL policies achieve the lowest collision and off-road driving rates, considerably outperforming baseline and similarly

structured BC models. Furthermore, extending GAIL to recurrent policies leads to improved performance. This result is an interesting contrast with the BC policies, where the addition of recurrence tends not to yield better results. Thus, we find that recurrence by itself is insufficient for addressing the detrimental effects that cascading errors can have on BC policies.

3.6 Multi Agent Imitation

In this subsection, we describe experiments and results conducted for multiple learning agents using the parameter sharing approach (PS-GAIL) described in Section 2.5.

In the multi-agent setting, multiple vehicles are sampled from the demonstration NGSIM data, and a policy with shared parameters is learned by batching together the observations and actions from all the vehicles. Importantly, the dynamics of the environment change along with the agent policies. Our training procedure must therefore account for non-stationary environment dynamics.

We mitigate this problem by introducing a curriculum that scales the difficulty of the multi-agent learning problem during training. J. Gupta et al. (2017) define a multi-agent curriculum, $\mathcal{C}$, as a multinomial distribution over the number of agents controlled by the policy each episode. The curriculum gradually shifts probability mass to larger numbers of agents. In practice, we use a simplified curriculum that increments the number of controlled agents by a fixed number every K iterations during training, in which case $\mathcal{C}(k)$ is a deterministic function of the iteration k.

We use recurrent neural network (RNN) policies, in all cases consisting of 64 Gated Recurrent Units (GRUs). The observation is passed directly into the RNN without any initial reduction in dimensionality. We use recurrent policies in order to address the partial observability of the state caused by occluded vehicles. In the multi-agent setting, a single shared policy selects actions for all vehicles, following the parameter sharing approach previously described. Policy optimization is performed using an implementation of TRPO from rllab (Duan et al., 2016) with a step size of 0.1.

We use two training phases for all of the models. The first phase consists of 1000

iterations with a low discount of 0.95 and a small batch size of 10 000 observation-action pairs. The second phase fine-tunes the models, running for 200 iterations with a higher discount of 0.99 and larger batch size of 40 000. For the multi-agent model, we add 10 agents to the environment every 200 iterations of the first training phase. We use 100 agents in the fine-tune phase for the multi-agent GAIL models.

The critic acts as the surrogate reward function in the environment. The observation-action pairs for each vehicle at each timestep are passed to the critic, which outputs a scalar value that is then used as the reward for that vehicle. The critic is implemented as a feed-forward neural network consisting of (128,128,64) ReLU units. We implemented the critic as a Wasserstein GAN with a gradient penalty (WGAN-GP) of 2 (Gulrajani et al., 2017). Similar to Li et al. (2017), we used a replay memory for the critic in order to stabilize training, which contains samples from the three most recent epochs. For each training epoch of the policy, the critic is trained for 40 epochs using the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.0004, dropout probability of 0.2, and batch size of 2000. Half of each batch consists of NGSIM data, with the remaining half comprised of data from policy rollouts. Finally, the reward values output from the critic are adaptively normalized to have zero mean and unit variance prior to being passed to TRPO.

The difficulty of the multi-agent task scales with the number of agents controlled in the environment. Figure 3.4 shows the performance of the two models as a function of the number of agents driven by our learned driving policy. The indicated number of agents are randomly sampled and replaced in the environment with the policy, while the remaining agents are left as originally recorded in NGSIM. Here, the single-agent policy refers to the policy trained using data obtained from one vehicle, and then deployed on multiple vehicles during validation. The results indicate that while the single-agent policy deteriorates rapidly with increasing number of agents, the multi-agent policy declines in performance much more gradually.

3.6.1 Reward Augmentation

Both single-agent GAIL and PS-GAIL are methodologies that are domain agnostic. However, for the specific task of driver modeling, providing the learning agent with domain knowledge proves useful. Reward Augmented Imitation Learning (RAIL) provides external penalties during training (Bhattacharyya et al., 2019) that specifically encapsulate rules of the road. These include penalties for going off the road, braking hard, and colliding with other vehicles. All of these are undesirable driving behaviors and therefore should be discouraged in the learning agent. These penalties help to improve the state space exploration of the learning agent by discouraging bad states such as those that could potentially lead to collisions. In RAIL, part of the reinforcement learning cost signal comes from the critic based on imitating the expert, and another cost signal comes from the externally provided penalties specifying the prior knowledge of the expert (Li et al., 2017).

For reward augmentation, we require that π belongs to a certain set such that undesired actions are discouraged. For example, the vehicle should not drive off road, collide with others, or brake too hard. Such undesired state-action pairs are denoted as belonging to the set U. The constraint on the policy is denoted by Π:

$$\Pi = \{\bar{\pi} : \pi = \pi_i, \forall\, i, \text{ and } \mathbb{P}_{\bar{\pi}}\left[\mathbf{a} \mid \mathbf{s}\right] = 0, \forall\, (\mathbf{s}, \mathbf{a}) \in U\} \tag{3.3}$$

Considering Wasserstein distance (Arjovsky et al., 2017), the following constrained minimax problem for imitation learning is formulated:

$$\min_{\bar{\pi} \in \Pi} \max_{D} \{\mathbb{E}_{\bar{\pi}_E}[D(\mathbf{s}, \mathbf{a})] - \mathbb{E}_{\bar{\pi}}[D(\mathbf{s}, \mathbf{a})]\} \tag{3.4}$$

where the critic, D, learns to output a high score when encountering pairs from $\bar{\pi}_E$, and a low score when encountering generated pairs from $\bar{\pi}$. D should be optimized for all functions.

The constrained minimax is solved by transforming the problem to an unconstrained form. The constraint for parameter sharing is naturally encoded by sharing the same policy for all agents. The constraint for reward augmentation is enforced by

adding a reward augmentation regularizer in the function. Thus, the unconstrained problem becomes:

$$\min_{\pi} \max_{D} \mathbb{E}_{\bar{\pi}_E}[D(\mathbf{s}, \mathbf{a})] - \mathbb{E}_{\pi}[D(\mathbf{s}, \mathbf{a})] + r\mathbb{E}_{\pi}[\mathbf{1}_U] \tag{3.5}$$

where r is the penalty, and $\mathbf{1}_U$ is an indicator function that is non-zero if and only if $(\mathbf{s}, \mathbf{a}) \in U$. The penalty r can either be a constant value or a barrier function. We have binary penalty when r is constant, and smooth penalty when r is a continuous function that reaches zero on the boundary of the set U. Note that the term $\mathbb{E}_{\pi}[D(\mathbf{s}, \mathbf{a})]$ is different from $\mathbb{E}_{\bar{\pi}}[D(\mathbf{s}, \mathbf{a})]$, where the former notation requires that all agents use the same policy π with shared parameters.

We explore a binary penalty and a smoothed penalty as the two forms of reward augmentation provided to the imitation learning agent. The first method of reward augmentation that we employ is to penalize states in a binary manner, where the penalty is applied when a particular event is triggered. To calculate the augmented reward, we take the maximum of the individual penalty values. For example, if a vehicle is driving off the road and colliding with another vehicle, we only penalize the collision. This will also be important when we discuss *smoothed penalties*.

We explore penalizing three different behaviors. First, we give a large penalty R to each vehicle involved in a collision. Next, we impose the same large penalty R for a vehicle that drives off the road. Finally, performing a hard brake (acceleration of less than $-3\,\mathrm{m/s^2}$) is penalized by only $R/2$. The penalty formula is shown in eq. (3.6). We denote the smallest distance from the ego vehicle to any other vehicle on the road as d_c (meters), where $d_c \geq 0$. We also define the closest distance from the ego vehicle to the edge of the road (meters): $d_{\text{road}} = \min\{d_{\text{left}}, d_{\text{right}}\}$. We allow d_{road} to be negative if the vehicle is off the road. Finally, let a be the acceleration of the vehicle in $\mathrm{m/s^2}$. A negative value of a indicates that the vehicle is braking. Now, we

can formally define the binary penalty function:

$$\text{Penalty} = \begin{cases} R & \text{if } d_c = 0 \\ R & \text{if } d_{\text{road}} \leq -0.1 \\ \frac{R}{2} & \text{if } a \leq -3 \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

The relative values of the penalties indicate the preferences of the designer of the imitation learning agent. For example, in this case study, we penalize hard braking less than the other undesirable traffic phenomena.

We hypothesize that providing advanced warning to the imitation learning agent in the form of smaller, increasing penalties as the agent approaches an event threshold will address the credit assignment problem in reinforcement learning. In this case, we provide a *smooth penalty* for off-road driving and hard braking, where the penalty is linearly increased from a minimum threshold to the previously defined event threshold for the binary penalty.

For off-road driving, we linearly increase the penalty from 0 to R when the vehicle is within 0.5 m of the edge of the road. For hard braking, we linearly increase the penalty from 0 to $R/2$ when the acceleration is between $-2\,\text{m/s}^2$ and $-3\,\text{m/s}^2$.

The driving performance of driving policies trained using PS-GAIL and RAIL was assessed by performing experiments in the simulator. Figure 3.5 shows root mean square error results for prediction horizons up to 20 s. These plots indicate that the multi-agent learning approaches PS-GAIL and RAIL capture expert behavior more faithfully than single-agent GAIL. This performance discrepancy is especially pronounced for longer prediction horizons, where the errors for single-agent policies begin to accumulate rapidly. Further, reward augmentation results in better local imitation performance, as seen by the lowest RMSE values.

The superior performance of PS-GAIL and RAIL is further illustrated by Figure 3.6. These validation results empirically demonstrate that PS-GAIL and RAIL policies are less likely to lead vehicles into collisions, extreme decelerations, and off-road driving. This serves as further illustration that the PS-GAIL training procedure

encourages stabler interactions between agents, thereby making them less likely to encounter extreme or unlikely driving situations. The inclusion of domain knowledge is especially significant here as seen by the reduction in the values of the undesirable metrics of driving.

3.7 Discussion

This chapter applied GAIL to the problem of driver modeling using driving demonstration data from the NGSIM dataset. This chapter first investigated the performance of GAIL as compared to behavior cloning and rule-based approaches for modeling single agent driving. In addition to establishing superior performance of GAIL, this investigation also showed that recurrent driving policies perform better than multi layer perceptron policies. Subsequently, this chapter showed that while GAIL fails to model the behavior of multiple driving agents, PS-GAIL is able to do so by learning a policy with shared parameters across driving agents. Both GAIL and PS-GAIL were agnostic to the driving problem and treated the demonstrations simply as trajectories of state-action pairs. This chapter proposed RAIL that enables providing domain knowledge to the learning agent, in this case in the form of penalty functions for violating driving rules. RAIL was shown to achieve more realistic driving behavior than PS-GAIL.

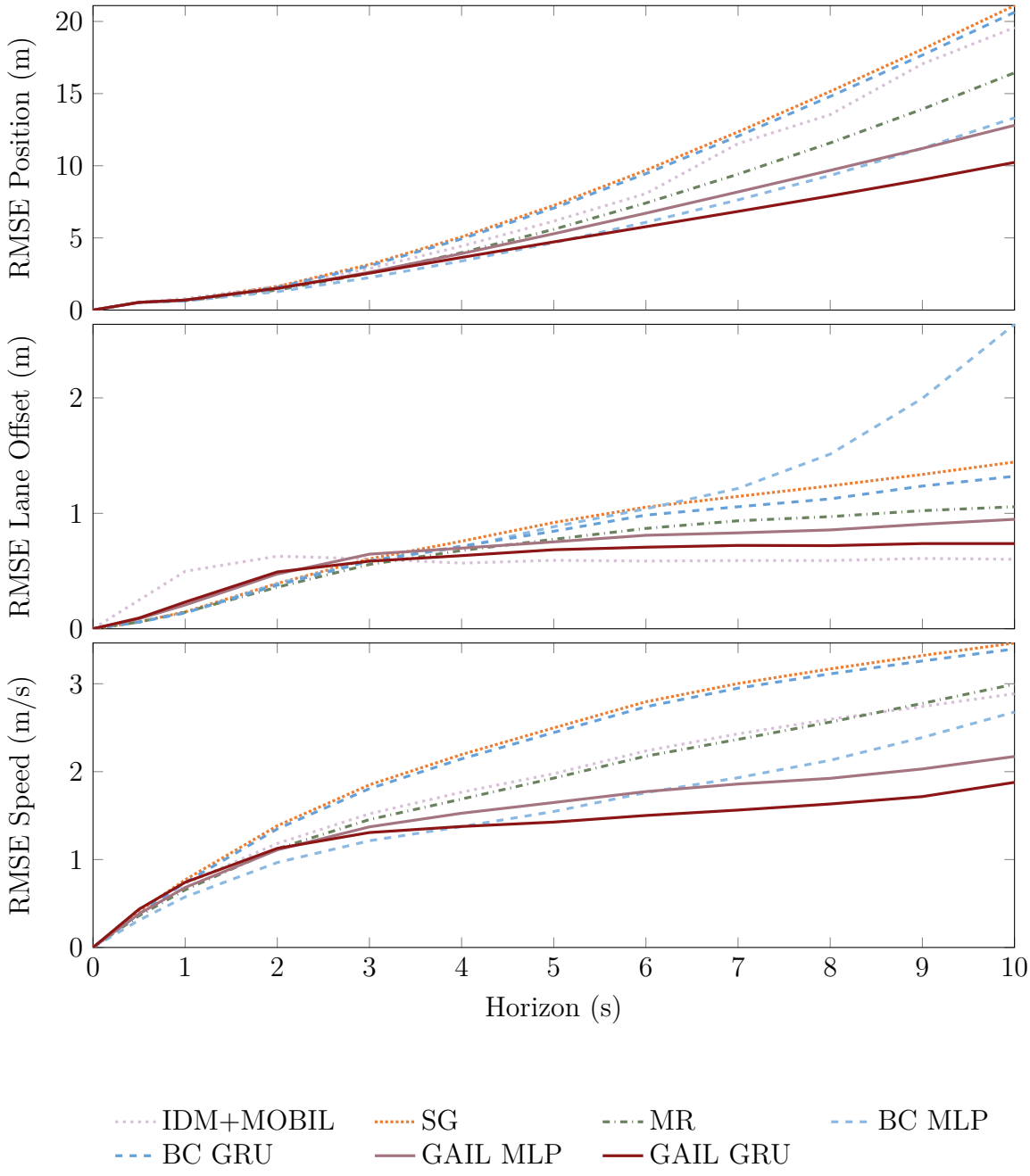

Figure 3.2: The root mean square error in position, velocity and lane offset for each candidate model versus prediction horizon. Policies trained using GAIL outperform the other methods.

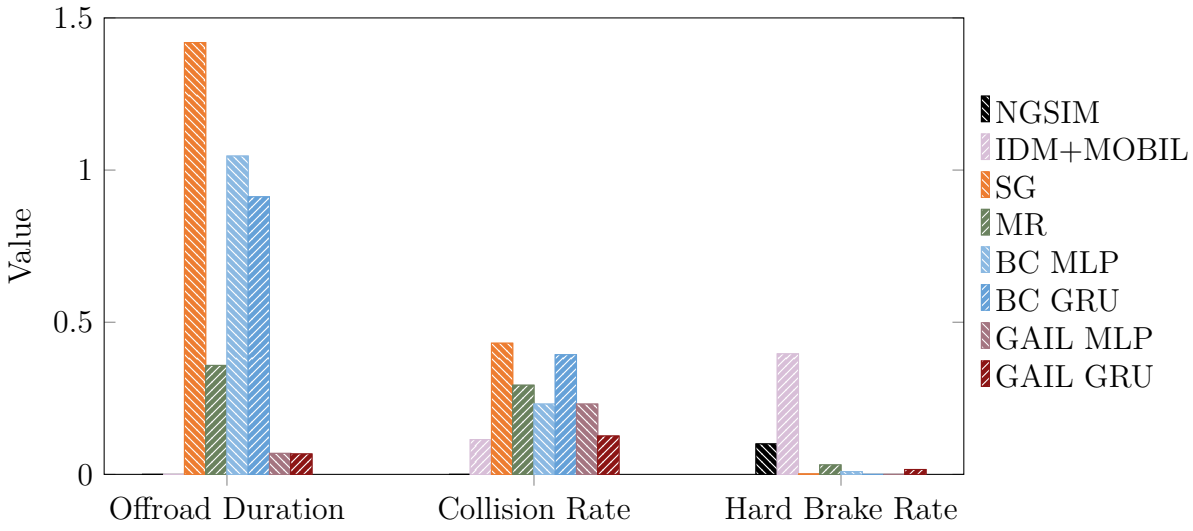

Figure 3.3: Metrics of undesirable traffic phenomena such as collisions, off the road driving, and hard decelerations. The GAIL based driving policies perform better than BC.

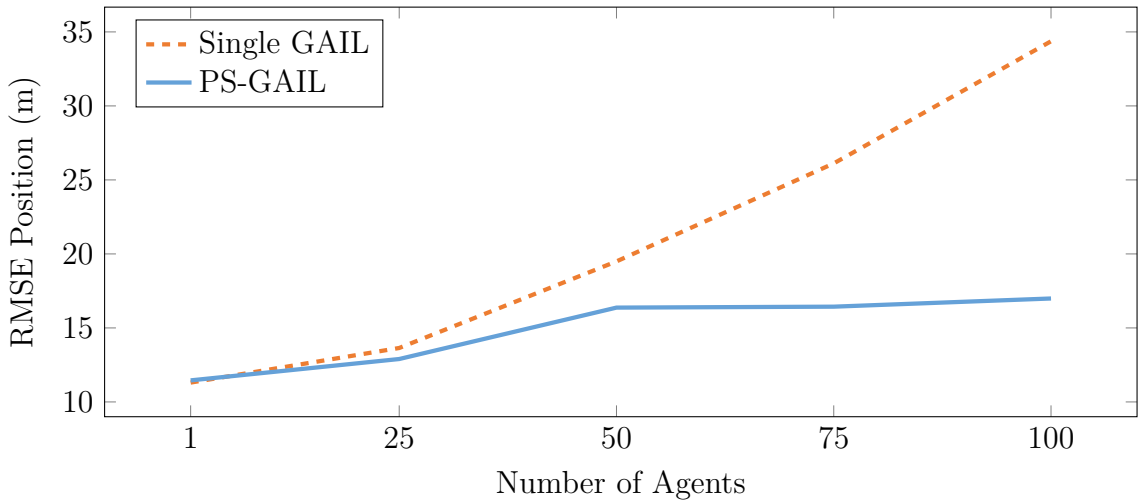

Figure 3.4: Average RMSE position value across all timesteps of an episode as a function of the number of controlled agents. As the policy controls more vehicles, single-agent GAIL performance deteriorates rapidly, while PS-GAIL performance decays more slowly.

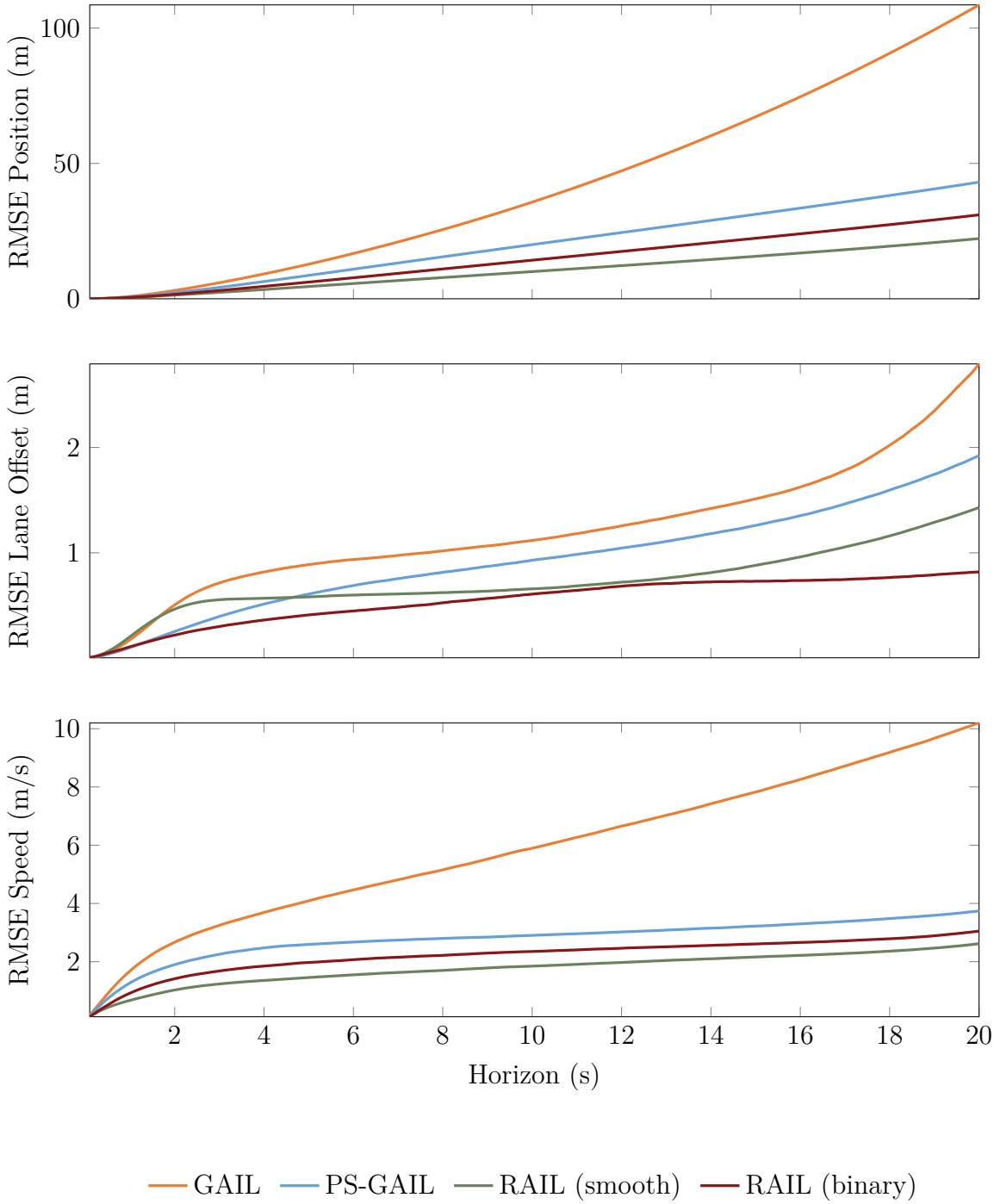

Figure 3.5: A comparison of root mean square error in position, lane offset and speed for single-agent, multi-agent and reward augmented GAIL versus prediction horizon. Policies trained using reward augmented GAIL show better prediction performance.

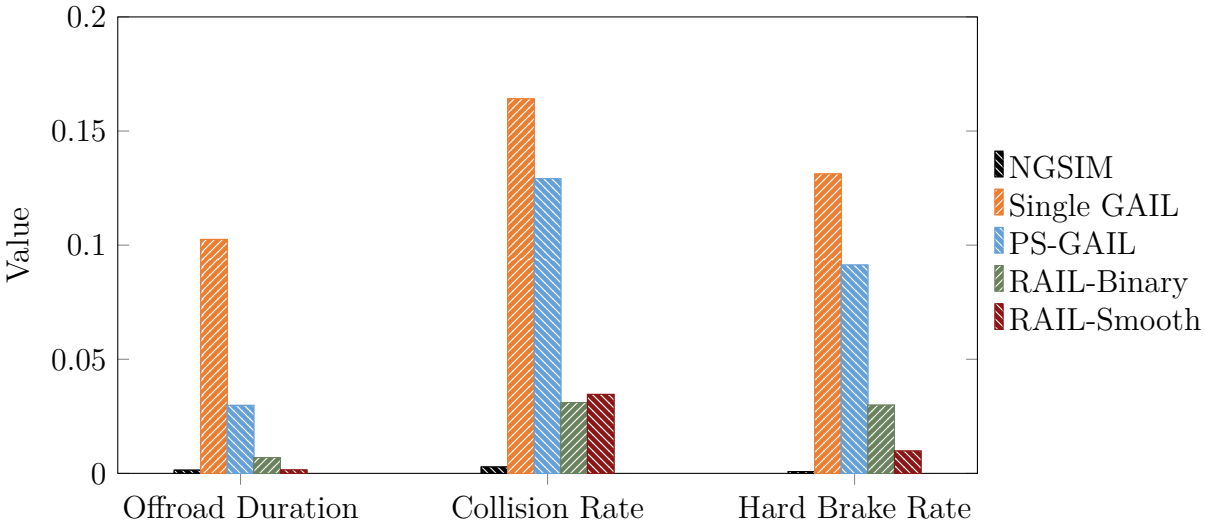

Figure 3.6: Metrics of undesirable traffic phenomena. These are explicitly penalized in the reward augmentation formulation. RAIL results in policies with much lower values of collisions, offroad driving and hard braking as compared to the PS-GAIL baseline.

Chapter 4

Driver Modeling using Particle Filtering

The previous two chapters discussed the use of Generative Adversarial Imitation Learning (GAIL) to model human motion from demonstration trajectories. Deep neural driving policies were learnt to drive multiple vehicles on straight highways. The appeal of this methodology was the ability to learn driving without knowing the underlying rules of driving and mapping relatively low level features to driving actions.

However, many of the intended domains of human behavior modeling are safety-critical such as indoor robotics, robot navigation in crowds, and driving, where faulty behavior by the autonomous agent can lead to loss of life and property. Therefore, interpretability of human models is key to understanding the decisions made by autonomous agents. For example, in the driving experiments reported in the previous chapter, vehicles showed collisions. However, it was difficult to determine the cause of these collisions as deep neural driving policies are not easily interpretable.

Rule-based models tend to be more interpretable to human designers compared to black-box artefacts such as neural networks. However, rule-based models are generally deterministic and do not take advantage of the variability we see in large datasets, instead relying on heuristics to assign the parameters of the model (Helbing and Molnar, 1995; Treiber et al., 2000).

In this chapter, we propose a methodology that combines rule-based modeling with data-driven learning. The parameters of an underlying rule-based model are learned online from human demonstration data using particle filtering. The proposed methodology is especially suited to human modeling because human behavior is inherently stochastic, i.e., given the same situation, humans may not necessarily take the same action every time (G. Bansal et al., 2019; Pentland and A. Liu, 1999). In our method, we incorporate this stochasticity as part of model parametrization. Further, our method results in a distribution over model parameters that can be sampled from to generate novel scenarios. In the next chapter, we demonstrate the proposed methodology to model real world driving behavior from demonstration trajectories.

4.1 Background

Rule-based modeling is an approach that uses a set of rules that indirectly specifies a mathematical model. For instance, if an autonomous vehicle equipped with a rule-based driving model observes an orange traffic light while driving, its rule enforces the model to set its acceleration to zero. Although rule-based models are interpretable, they are brittle and struggle to generalize to diverse scenarios since they only rely on predefined rules. For instance, in autonomous driving, different drivers inherently have different driving patterns (Brown et al., 2020). Therefore, we need a model that generalizes human behavior while accounting for individual variations.

In contrast to rule-based models, we can also develop a model to learn purely from data. Such black-box models do not have a prescribed set of rules. The advantage of data-driven models is that they can learn arbitrarily complex patterns from large amounts of data. However, they have two main disadvantages. First, it can be challenging to explicitly incorporate physical knowledge or structure in such models. For instance, as humans, we know that vehicles should not collide with each other. However, as found by (Bhattacharyya et al., 2020b), it is challenging for completely data-driven techniques to learn such rules. Second, because the data-driven models are not interpretable, it is difficult to verify and validate them, making them less

attractive for safety-critical applications such as autonomous driving. In such applications, engineers should be able to stress test the system before deployment by taking into account various possible failure modes (R. Lee et al., 2018). In case of failure, they should be able to understand the underlying reason for the failure. In the field of system identification, attempts to combine black-box models with white-box models are known as gray-box modeling. A common approach is to combine a partial theoretical structure with data to complete the model. Such models have been used for modeling nonlinear system dynamics (J. Gupta et al., 2020; Menda et al., 2020; Pearson and Pottmann, 2000).

With the aim of operating robots around humans, both rule-based and data-driven techniques have been used to model human behavior. These studies have focused on various aspects of human behavior. Z. Wang et al. (2013) and Völz et al. (2016) attempt to infer the intentions of humans in a collaborative robot manipulation task using Bayesian estimation and pedestrian crossing using convolutional neural networks, respectively. Such signals about how humans would behave in the future can help robot decision-making (Bai et al., 2015; Z. Wang et al., 2013). In a similar problem, Chandra et al. (2020) attempt to model future actions given the observations in previous time steps using recurrent neural networks. Some of these models are purely rule-based (Helbing and Molnar, 1995) while some are purely data-driven (Chandra et al., 2020).

Another aspect of modeling human behavior is modeling its intrinsic decisions. There have been attempts to model the rationality (Reddy et al., 2018) and legibility (Dragan et al., 2013) in human-robot interaction. In imitation learning, the objective is to learn a policy to imitate a set of human demonstrations. Behavioral cloning (Ross et al., 2011) is one way to learn such demonstrations from data. However, such supervised learning techniques have proven to be less successful in applications such as modeling multi-agent traffic due to compounding errors and not taking into account multi-agent interactions (Bhattacharyya et al., 2020b). Techniques such as inverse reinforcement learning (Abbeel and Ng, 2004; Gombolay et al., 2018) and inverse reward design (Hadfield-Menell et al., 2017) attempt to directly model the underlying reward function of humans. However, it is not clear how to incorporate

traffic rules and road geometry into these models.

The intelligent driver model (IDM) is a widely used rule-based dynamics model used for human driving behavior (Treiber et al., 2000). It can be used to drive a vehicle at a desired speed in a specific lane while maintaining a minimum spacing with the leading vehicle. Augmenting IDM with MOBIL (Kesting et al., 2007), another rule-based model, can be used to switch lanes. Even though, by construction, these models are guaranteed to avoid collisions, they do not, 1) consider interactions with other vehicles beside the leading vehicle, 2) reflect natural human driving styles as parameters are arbitrarily set by the model user, and 3) account for individual driving behavior because these deterministic models can only have a single set of scalar-valued parameters.

4.2 Motion Modeling using Particle Filtering

This section introduces the problem of learning human behavior models from demonstrations and outlines our methodology for interpretable human behavior modeling.

4.2.1 Problem Definition

We are given a batch of trajectories $\mathbf{y}_{1:T} \in \mathbb{R}^{D \times T}$ of human demonstrations over a time horizon T. We assume that the human demonstration follows a dynamical system with state $\mathbf{x} \in \mathbb{R}^D$ that evolves according to the following equations:

$$\begin{aligned} \mathbf{x}_{t+1} &= f_{\theta_t}(\mathbf{x}_t, \mathbf{w}_t) \\ \mathbf{y}_t &= g_{\theta_t}(\mathbf{x}_t, \mathbf{v}_t), \end{aligned} \tag{4.1}$$

where $\mathbf{w}_t$ is the process noise and $\mathbf{v}_t$ is the observation noise at time t. We assume that we are provided a class of parametrized models $f_\theta(x, t)$ and $g_\theta(x, t)$ that approximate the dynamical system's evolution and observation processes, respectively.

Our goal is to learn distributions over the parameters of the functions f and g. These distributions represent the variation in the possible human behaviors.

Algorithm 2 Human behavior model parameter estimation using particle filtering

Input: Demonstration trajectories of length T, Starting scene with K humans, Initial set of particle sets $\{\Theta_1, \Theta_2, \ldots, \Theta_K\}$
for $k \leftarrow 1, 2, \ldots, K$ {humans}
  for $t \leftarrow 0, 1, \ldots, T$ {time-steps}
    $\mathbf{y}_{t+1}^k \leftarrow$ ground truth observation of the kth human at $t+1$
    for $i \leftarrow 1, 2, \ldots, I_k$ {particles}
      $\theta_i \leftarrow$ random particle in Θ_k
      $\mathbf{y}_{t+1}^{k,i} \sim$ Observation evolution {Sampled next observation for kth human using the ith particle}
      $w_i \leftarrow O\big(\mathbf{y}_{t+1}^k \mid \mathbf{y}_{t+1}^{k,i}\big)$ {Probability density of true next observation given sampled next observation}
    $\Theta_k \leftarrow$ Obtain I_k samples from Θ_k according to $[w_1, w_2, \ldots, w_{I_k}]$ {Resampling}
    Step trajectory forward by one time-step
Combine particles obtained from all humans and use as prior for next epoch

4.2.2 Method

We view the problem of finding parameter distributions from the lens of state estimation. The parameters of the model evolve according to a hidden Markov model where the transition distribution is governed by $p(\theta_{t+1} \mid \theta_t)$ and the observation distribution is governed by $p(\mathbf{y}_t \mid \theta_t)$ as shown in Figure 4.1. Now, we can view the problem as a state estimation problem where we want to find the posterior distribution $p(\theta_T \mid \mathbf{y}_{1:T})$.

The distribution over parameters can be written as

$$p(\theta_T \mid \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T), \tag{4.2}$$

where $(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T)$ denotes a sequence of observations from demonstration data. This inference problem can be solved using recursive Bayesian estimation, where the recursive update equation is given by

$$p(\theta_T \mid \mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t \mid \theta_T) p(\theta_T \mid \mathbf{y}_{1:t-1})}{\int_{\theta_T} p(\mathbf{y}_t \mid \theta_T) p(\theta_T \mid \mathbf{y}_{1:t-1}) \mathrm{d}\theta_T}. \tag{4.3}$$

The partition function (the denominator) in equation 4.3 cannot be evaluated analytically for general nonlinear distributions. Rather than imposing restrictive

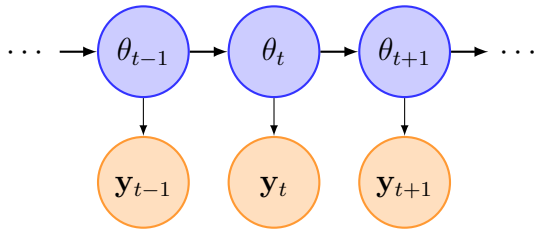

Figure 4.1: Hidden Markov model showing the human behavior modeling problem in the state estimation framework. θ_t and $\mathbf{y}_t$ are the parameters and observations at time t, respectively. The objective is to learn a distribution over the latent parameters using data from multiple human demonstrations.

assumptions on the form of the distribution, we use particle filtering (J. S. Liu and Chen, 1998; Thrun, 2002) to approximately solve the inference problem. A particle filter approximates a continuous probability distribution with a collection of sampled particles. The parameters also include the inherent stochasticity in human behavior, and the particles represent our uncertainty over this stochasticity. It is important to note that this method uses particle filtering in a novel way. While particle filtering is typically used to do state estimation, in this method it is used to estimate the parameters of an underlying model.

We perform particle filtering over the trajectory provided by one human demonstrator to reach a distribution over the parameters for that particular demonstrator. We then mix the distributions obtained from multiple human demonstrators. To maintain the simplicity of our approach, in this work, we combine the sample-based representation of the distributions obtained from different demonstration trajectories. This constitutes one epoch of our approach. For the subsequent epoch, the distribution learned from the previous epoch is used as a prior from which the initial particle set is sampled. Our algorithm is shown in Algorithm 3.

Figure 4.2 illustrates the particle filtering procedure in our driver modeling case study. For the purposes of illustration, we assume that the parameter space of the driver model is 2 dimensional. First, a set of particles is sampled from a uniform distribution. Each particle is then used to hallucinate the vehicle one step forward.

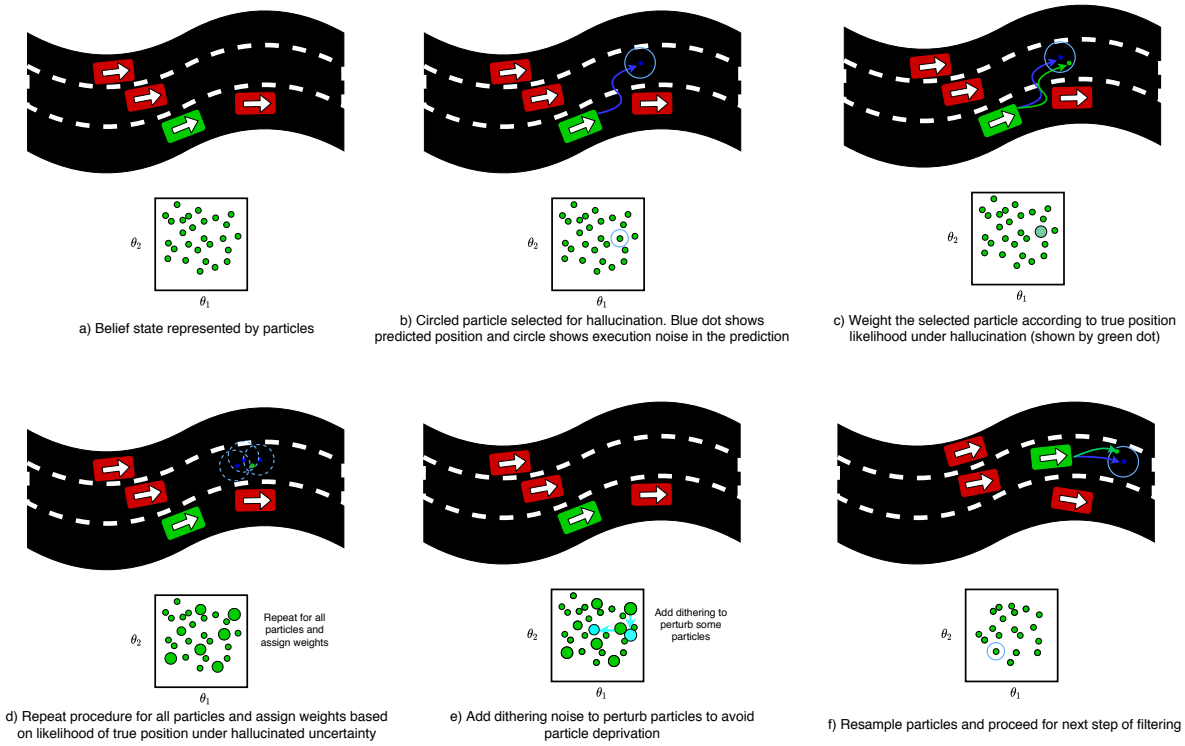

Figure 4.2: The particle filtering process to learn a distribution over the parameters of the underlying rule based driver model from ground truth demonstration data. The vehicle of interest (green) interacts with the surrounding vehicles (red). The blue trajectories show hallucinations carried out by different particles and the green trajectory shows the ground truth. The ground truth position likelihood under the distribution over hallucinated position is used to weight and resample the particles.

The hallucinated position is used as the mean of a bivariate Gaussian distribution whose covariance is governed by the stochasticity parameters. The mean position and the uncertainty are shown by the blue dot and circle respectively in panel b). Subsequently, the particle is weighted according the likelihood of the ground truth position under the bivariate Gaussian distribution. A similar procedure is carried out to assign weights to all the particles in the particle set resulting in a weighted particle set as shown in panel d). To counter the particle deprivation problem, a small amount of noise is added to the particle set. Finally, the vehicles are moved one step forward according to the ground truth trajectory, and the particle set is resampled according to the weighted set obtained in panel e).

At the end of the particle filtering process, every vehicle has an associated collection of particles that should best explain the observed driving behavior given in the demonstration trajectory. These particles represent possible setting of the parameters of the underlying rule-based models, i.e., the functions f and g, which provide interpretability to the models.

4.3 Intelligent Driver Model and Extensions

The Intelligent Driver Model (IDM) (Treiber et al., 2000) is a parametric rule-based car-following model that balances two forces, the desire to achieve free speed if there were no vehicle in front, and the need to maintain safe separation with the vehicle in front. The IDM is guaranteed to be collision free. The inputs to the model are the vehicle's current speed $v(t)$ at time t, relative speed $r(t)$ with respect to the leading vehicle, and distance headway $d(t)$. The model then outputs an acceleration according to

$$a_{\text{IDM}} = a_{\max}\left(1 - \left(\frac{v(t)}{v_{\text{des}}}\right)^4 - \left(\frac{d_{\text{des}}}{d(t)}\right)^2\right), \tag{4.4}$$

where the desired distance is

$$d_{\text{des}} = d_{\min} + \tau.v(t) - \frac{v(t).r(t)}{2\sqrt{a_{\max}.b_{\text{pref}}}}. \tag{4.5}$$

The model has several parameters that determine the acceleration output based on the scene information. Here, v_{des} refers to the free speed velocity, $d_{\min}$ refers to the minimum allowable separation between the ego and leader vehicle, τ refers to the minimum time separation allowable between ego and leader vehicle, $a_{\max}$ and b_{pref} refer to the limits on the acceleration and deceleration, respectively. Though the collision-free motion of a vehicle can be simulated by arbitrarily setting some parameter values, the driving behavior is not necessarily realistic. Therefore, in this work, we learn the parameters from real human driver demonstrations.

The literature contains various extensions of the original IDM. The Enhanced IDM incorporates a slight modification that prevents the model from "over-reacting"

when another vehicle cuts in front of it (Kesting et al., 2010). The Foresighted Driver Model modifies the output of the IDM based on factors like upcoming curvature in the road (Eggert et al., 2015). Liebner et al. (2012) incorporate a spatially varying velocity profile within the IDM to account for variation in different types of maneuvers through intersections. Hoermann et al. (2017) use a stochastic IDM model with fixed-variance additive Gaussian white noise. Schulz et al. (2018a) use a similar model that also incorporates context-dependent upper and lower bounds on acceleration (Schulz et al., 2018a,b).

Many approaches in the literature estimate IDM model parameters offline. Lefèvre et al. (2014) use constrained nonlinear optimization. Morton et al. (2016) use the Levenberg-Marquardt algorithm. Some approaches select the parameters heuristically (Schulz et al., 2018a,b). In fact, "recommended" parameter values have been published for the IDM (Treiber and Kesting, 2017).

Offline estimation is also used for selecting parameter values in black-box driver models. Lefèvre et al. (2014) use Expectation Maximization (EM) to train a Gaussian mixture model (GMM), and the Levenberg-Marquardt algorithm to train a neural network (NN). Morton et al. (2016) use gradient-based optimization to train various feedforward and recurrent neural network models. Kuefler et al. (2017) use Generative Adversarial Imitation Learning (GAIL) to train a recurrent neural network.

Some approaches estimate driver model parameters online. In Multi-Policy Decision-Making, the parameters of several hand-crafted control policies are estimated online with Bayesian Changepoint Estimation and Maximum-likelihood estimation (Galceran et al., 2017). Sadigh et al. (2018) use online active information gathering to estimate the parameters of a human driver's reward function.

Several online estimation approaches are used for IDM in particular. Monteil et al. (2015) use an Extended Kalman filter. Examples of particle filters used with IDM parameters include approximate online POMDP solvers (Sunberg et al., 2017) and fully probabilistic scene prediction algorithms (Hoermann et al., 2017). The online parameter estimation approach of Buyer et al. (2019) is similar to ours, although they use a different IDM extension and do not use their model for forward simulation of traffic scenes (Buyer et al., 2019). None of the above models explicitly estimate

"stochasticity" parameters for individual drivers.

4.4 Particle Filtering

We wish to maintain a distribution over model parameters for each driver in a given traffic scene. We assume that the parameters θ_i of driver i are stationary, i.e., human drivers do not change their latent driving behavior over the time horizons being considered in this work. While this assumption may not be entirely realistic (humans may have to change driving style given extreme situations), the relaxation to non-stationary parameters has been left to future work. fig. 4.3 shows a Bayesian network where the hidden state contains the parameters of the IDM and the observations are the position trace obtained from demonstration data.

Given a starting scene, each vehicle has an associated initial set of I particles contained in the set $\Theta = \{\theta_1, \theta_2, \ldots, \theta_I\}$ For all these vehicles, demonstration trajectories of length T are given. At every time-step, a random particle θ_i is selected from the set Θ and a new position x^i_{t+1} for the vehicle is sampled from the generative model defined by the IDM with parameters represented by the particle. We explicitly capture the interaction between vehicles by sampling the new position in the presence of other vehicles driven by a deterministic IDM. The particle is then weighted according to the likelihood of the true position x_{t+1} under the distribution given by the sampled position x^i_{t+1}. This likelihood is computed by querying the Gaussian

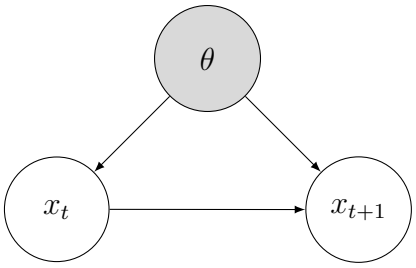

Figure 4.3: Bayesian network showing the driver modeling problem in the state estimation framework. θ represents the IDM parameters, x_t and x_{t+1} are the positions at timestep t and $t+1$, respectively. The objective is to learn the latent IDM parameters θ from data collected from a human driver.

probability density function with mean given by x_{t+1}^i and variance given by $\sigma_{\text{IDM}}\Delta t^2$ per eq. (4.8). Finally, particles are resampled according to the weights to yield a new set of particles. The particle filtering algorithm is given in algorithm 3.

Algorithm 3 IDM parameter estimation using particle filtering

Input: Expert trajectories of length T, Starting scene with K vehicles, Initial set of particle sets $\{\Theta_1, \Theta_2, \ldots, \Theta_K\}$
for $t \leftarrow 0, 1, \ldots, T$ {time-steps}
  for $k \leftarrow 1, 2, \ldots, K$ {vehicles}
    $x_{t+1}^k \leftarrow$ true position of the kth vehicle at $t+1$
    for $i \leftarrow 1, 2, \ldots, I_k$ {particles}
      $\theta_i \leftarrow$ random particle in Θ_k
      $x_{t+1}^{k,i} \sim \mathcal{N}(x_t^k + \frac{1}{2}a_{\text{IDM}_{\theta_i}}\Delta t^2, \sigma_{\text{IDM}_{\theta_i}}\Delta t^2)$ {Sampled next position of the kth vehicle using the ith particle - eq. (4.8)}
      $w_i \leftarrow O\big(x_{t+1}^k \mid x_{t+1}^{k,i}\big)$ {Probability density of true next position given sampled next position}
    $\Theta_k \leftarrow$ Obtain I_k samples from Θ_k according to $[w_1, w_2, \ldots, w_{I_k}]$ {Resampling}

Particle filtering is susceptible to the particle deprivation problem wherein particles converge to one region of the state space and there is no exploration of other regions. Dithering (Schön et al., 2011) helps to mitigate the particle deprivation problem wherein external noise is added to aid exploration of state space regions. We implement dithering by adding random noise to the top 20% particles ranked according to the corresponding likelihood.

Rather than estimating all model parameters, we estimate only the desired velocity (v_{des}) and the driver-dependent stochasticity (σ_{IDM}) as the parameters to be estimated. Therefore, the parameter space is two-dimensional, with $\theta = [v_{\text{des}}, \sigma_{\text{IDM}}]$.

At the end of the particle filtering process, every vehicle has an associated collection of particles that should best explain the observed driving behavior given in the demonstration trajectory. IDM parameters are then extracted from this collection of particles to drive the vehicles in simulation and collect metrics to assess the driving behavior.

4.5 Experiments on Highway Driving

We evaluate the performance of our model on demonstration data from two real world datasets, namely the Next-Generation Simulation (NGSIM) for US Highway 101 (Colyar and Halkias, 2007) which provides driving data collected at 10 Hz and the Highway Drone Dataset (HighD) (Krajewski et al., 2018) which provides driving data from German highways recorded at 25 Hz using a drone. We benchmark our approach against representative rule-based and black-box models as well as constant velocity and constant acceleration baselines.

Experiments are conducted on a set of thirty scenarios (fifteen scenarios randomly sampled from each dataset). Twenty vehicles in each scenario are randomly selected as target vehicles. For each scenario and each model, predicted trajectories are generated by forward simulation of this set of target vehicles over a 5 s time horizon, where the target vehicles are controlled by the driver model defined by the parameters estimated using particle filtering.

We use Root Mean Squared Error (RMSE) of the position and velocity to measure "closeness" of a predicted trajectory to the corresponding ground-truth trajectory. Figure 4.6 shows the RMSE over time for an example scenario with 20 vehicles over a 5 s duration from the NGSIM dataset.

While RMSE measures prediction accuracy at the level of individual vehicles by comparing the obtained trajectories against ground truth from the demonstration trajectories, we also wish to quantify how "safely" each model drives. To this end, we count the number of "undesirable events" (collision, going off the road, and hard braking) that occur in each scene prediction.

4.5.1 IDM with Stochasticty

To model human driving behavior, which is inherently stochastic (given the exact same scene, a human driver may not always take the same resulting action), we use the IDM with stochasticity (Treiber and Kesting, 2017). We assume that the output

acceleration is distributed according to

$$a \sim \mathcal{N}(a \mid a_{\text{IDM}}, \sigma_{\text{IDM}}), \tag{4.6}$$

where a_{IDM} and σ_{IDM} represent the mean and variance, respectively, of a Gaussian distribution. The mean a_{IDM} is computed with eq. (4.4), and σ_{IDM} is a new model parameter. Assuming the vehicle dynamics

$$x_{t+1} = x_t + \frac{1}{2} a \Delta t^2, \tag{4.7}$$

where x is the position and Δt is the unit-time, we obtain the new position distributed according to

$$x_{t+1} \sim \mathcal{N}(x_{t+1} \mid x_t + \frac{1}{2} a_{\text{IDM}} \Delta t^2, \sigma_{\text{IDM}} \Delta t^2). \tag{4.8}$$

4.5.2 Filtering

To estimate the parameters of the IDM using our filtering approach as per algorithm 3, the particles are initially sampled from a uniform distribution discretized into a grid with resolution of 0.5 m/s for the desired velocity parameter (v_{des}) and 0.1 for the stochasticity parameter (σ_{idm}). At the dithering stage (to avoid particle deprivation), we add noise sampled from a discrete uniform distribution with $v_{\text{des}} \in \{-0.5, 0, 0.5\}$ and $\sigma_{\text{IDM}} \in \{-0.1, 0, 0.1\}$. These values are chosen to preserve the discretization present in the initial sampling of particles. The time taken for filtering to converge in a 20 vehicle scenario over a 5 s duration was 30 s on an Intel Core i9-9900K eight-core processor.

To assess the convergence of the particle filtering approach, fig. 4.4 shows the root mean squared distance from the mean of the final particle distribution over the set of particles at every iteration. The particles converge as more demonstration data is shown to the filtering algorithm.

Figure 4.5 shows the mean particle after the filtering process for a subset of vehicles from both the NGSIM and the HighD datasets. The HighD vehicles have a higher desired velocity (v_{des}) parameter on average, reflecting the fact that vehicles drive

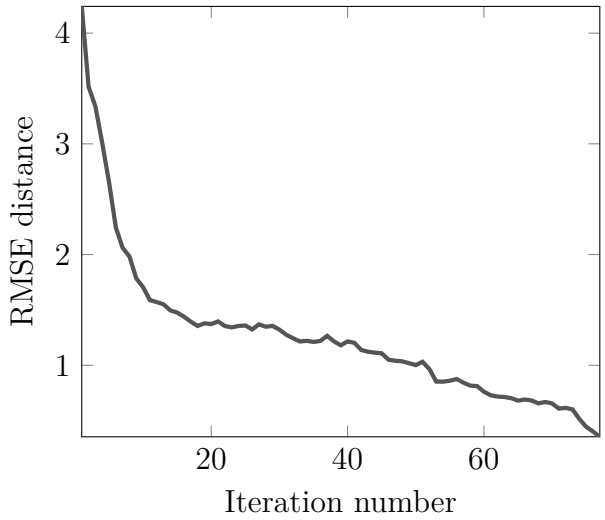

Figure 4.4: RMSE distance from final particle over particle set at every iteration averaged over all the vehicles. The particle set converges to the final particle with the progress of filtering.

faster on German highways.

4.5.3 Baseline Models

To benchmark the performance of our approach, we compare the driving behavior obtained by our model against that obtained by five other models. The first benchmark model is IDM with the "default" parameter values recommended in (Treiber and Kesting, 2017): $v_{\text{des}} = 30\,\text{m/s}$, $\tau = 1.0\,\text{s}$, $d_{\text{min}} = 2\,\text{m}$, $a_{\text{max}} = 3\,\text{m/s}^2$, and $b_{\text{pref}} = 2\,\text{m/s}^2$. Our second (also rule-based) benchmark model is the IDM with parameters obtained by offline estimation using non-linear least squares (Morton and Kochenderfer, 2017). The associated parameter values are $v_{\text{des}} = 17.837\,\text{m/s}$, $\tau = 0.918\,\text{s}$, $d_{\text{min}} = 5.249\,\text{m}$, $a_{\text{max}} = 0.758\,\text{m/s}^2$, and $b_{\text{pref}} = 3.811\,\text{m/s}^2$. Our third benchmark model is a recurrent network trained with Generative Adversarial Imitation Learning (GAIL) (Bhattacharyya et al., 2018). We also baseline our method against constant velocity (vehicles continue driving at the same speed that they start with at the beginning of the simulation) and constant acceleration (vehicles accelerating at $1\,\text{m/s}^2$) models.

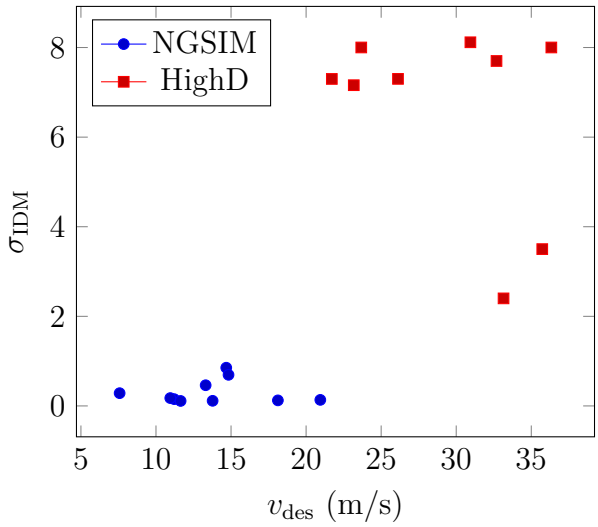

Figure 4.5: Mean particles from final distributions achieved after particle filtering for a set of 10 NGSIM and HighD vehicles observed over trajectories of 50 timesteps. HighD vehicles are faster on average.

4.5.4 Results

RMSE results for an example scenario with 20 vehicles over a 5 s duration from the NGSIM dataset are shown in fig. 4.6. We observe that our method provides driving trajectories that are closer to the ground truth as compared to those generated by IDM with default parameter values, and those generated by GAIL driven policies. We see that the RMSE in both position and velocity averaged over the set of vehicles is lowest for all timesteps using our driving model.

Further experiments on both NGSIM and HighD datasets are reported in table 4.1. These results are generated using 15 randomly sampled scenarios from both the HighD and NGSIM datasets. Every scenario is such that there is a set of 20 vehicles driving over a 5 s horizon which translates to 50 timesteps for NGSIM and 125 timesteps for HighD. We see that while our method outperforms other methods, it performs worse than the constant velocity baseline for the HighD dataset. One possible reason may be the default values for the parameters that govern the interaction between vehicles, i.e. minimum allowed separation d_{min} and minimum timegap τ. Including these parameters within the filtering process will allow finer grained driver modeling

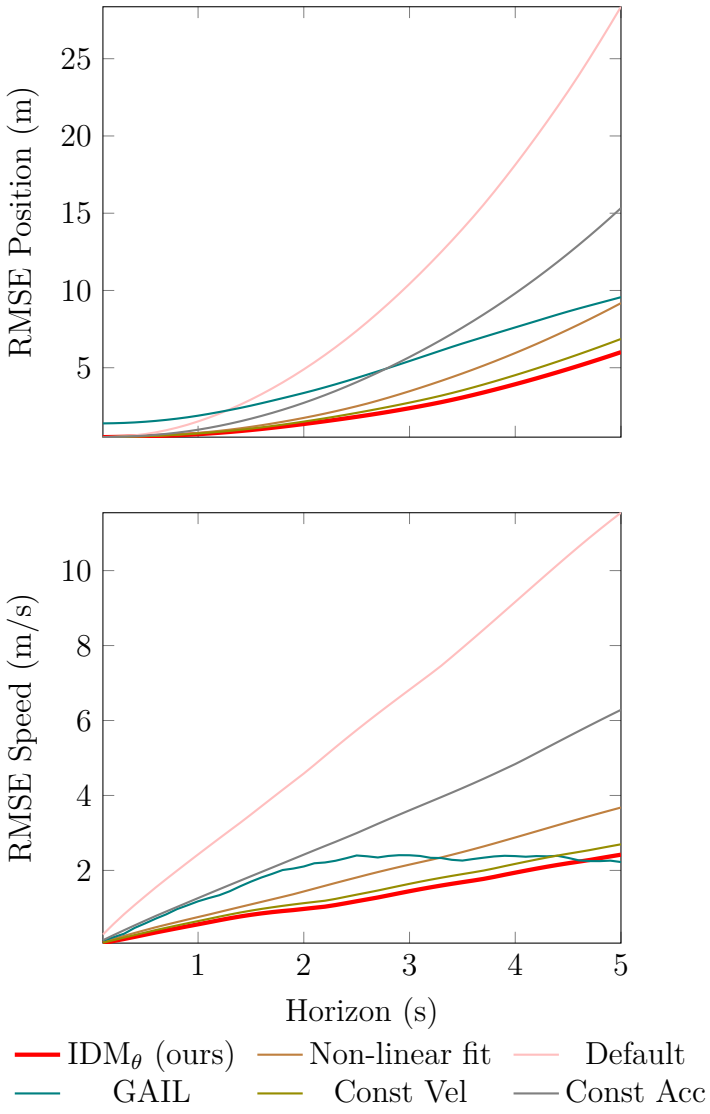

Figure 4.6: Root mean square error in position and velocity averaged over all cars to benchmark our model (IDM$_\theta$) against other driver models. Default refers to an IDM with parameters as set to default for motorways (Treiber and Kesting, 2017). Non-linear fit refers to an IDM with parameters estimated offline from data using non-linear least-squares fit (Morton and Kochenderfer, 2017). GAIL refers to a black box driver model trained using Generative Adversarial Imitation Learning (Bhattacharyya et al., 2018). Baseline models are constant acceleration and constant velocity driving models.

Table 4.1: Experiments over 15 randomly selected scenarios for both NGSIM and HighD each with 20 vehicles driving for a 5 s duration. The metrics column shows the RMSE values for position (Pos) and velocity (Vel) at the end of 5 s. Cumulative number of collisions (Colls) at the end of the horizon are also reported. Our model (IDM_θ) is compared against other models: non-linear least squares fit (LM Fit) (Morton and Kochenderfer, 2017), constant speed (Speed), constant acceleratino (Acc.), black-box baseline (GAIL) (Bhattacharyya et al., 2018) and heuristic parameters (default) (Treiber and Kesting, 2017).

		Models					
Metrics	**Dataset**	IDM_θ	Default	GAIL	Speed	Acc.	LM Fit
Pos	NGSIM	5.90	27.78	10.42	6.24	12.64	7.34
	HighD	8.02	18.30	13.63	2.42	11.01	35.13
Vel	NGSIM	2.12	10.72	3.52	2.22	5.03	2.69
	HighD	2.14	4.59	2.94	0.94	4.39	10.05
Colls	NGSIM	0	0	53	113	119	0
	HighD	0	0	15	0	27	0

and is an interesting direction for future work.

The cumulative number of undesirable driving instances for 20 vehicles over a 5 s duration in a congested traffic scenario from the NGSIM dataset is shown in fig. 4.7. The cumulative number of undesirable driving instances keep growing with time for the data-driven benchmark in fig. 4.7. This reflects the fact that GAIL does not provide guarantees on safety. As expected, the IDM based models including ours, and the two rule-based benchmarks do not show any collisions because the IDM is collision-free by default. The constant velocity and constant acceleration baselines also do not provide collision-free trajectories because they are not reacting to the vehicle in front of them but merely driving with constant velocity and acceleration, respectively.

Cumulative number of collisions for all vehicles over the duration of the trajectory are also reported in table 4.1. We observe that the constant velocity baseline suffers from no collisions in the HighD dataset. This is because the dataset is not as congested as the NGSIM dataset and hence vehicles start with sufficient distance headway and

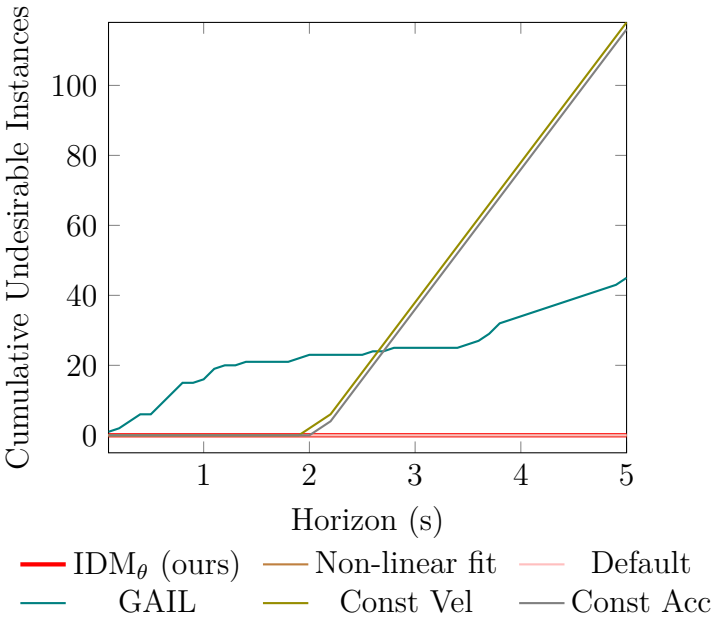

Figure 4.7: Cumulative number of undesirable instances summed over all vehicles over a 5 s time horizon using different driving models in a congested scenario from the NGSIM dataset. IDM based models, including ours, result in no collisions, off-the-road driving, or hard decelerations.

relative velocity to avoid collisions. However, the constant acceleration does result in some collisions whenever a faster vehicle starts out behind a slower vehicle. The data-driven benchmark also results in some collisions (fewer than NGSIM due to larger separation between vehicles). As expected, congested scenarios present a challenge for the benchmark models.

4.6 Experiments on Highway Merging

We demonstrate our proposed methodology in the case of modeling highway merging behavior. Merging is a complex problem involving negotiation between multiple drivers, and implicitly inferring the intent of these drivers. Prior approaches to modeling merging behavior have been in the context of planning. Bouton et al. (2019) demonstrate the use of hierarchical reinforcement learning to enable an ego vehicle to safely merge in dense traffic. Other approaches to merging include online planning (Hubmann et al., 2018; Schmerling et al., 2018; Ward et al., 2017), and game theoretic methods (Fisac et al., 2019; Sadigh et al., 2018), both of which suffer from the lack of ability to scale.

The Interaction Dataset (Zhan et al., 2019) contains interactive driving scenarios from different countries. For this work, we focus on a merging scenario. Figure 4.8 shows the map and the vehicles at one time snapshot in our driving simulation platform. There are two merge lanes from both directions. Further, the merge lanes allow lane changes into the main lanes. Our goal is to obtain a driver model that performs merging like humans do based on the demonstrations provided in the merging scenario. Further, we want to use these driver models to generate novel scenarios of interest.

4.6.1 Cooperative IDM and Baseline Models

We extend the Cooperative Intelligent Driver Model (CIDM) as originally proposed by Bouton et al. (2019). In addition to the IDM parameters, CIDM includes a cooperation parameter $c \in [0, 1]$ which controls the level of cooperation to the merging

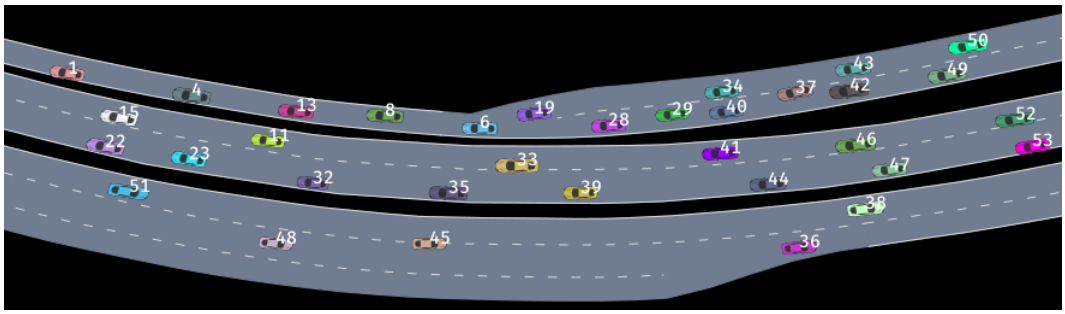

Figure 4.8: The Interaction dataset (Zhan et al., 2019) contains real-world driving demonstrations. In this case study, we model highway merging from demonstrations. This figure shows one time snapshot from an example scenario from the dataset. Here, vehicles 19, 34, 43 and 50 are attempting to merge into the main lane. Using particle filtering, we learn a distribution over the parameters governing the rule-based cooperative Intelligent Driver Model (Bouton et al., 2019) to best capture the demonstrated driving behavior.

vehicle. With $c = 1$, the driver slows down to yield to the merging vehicle and with $c = 0$, the driver completely ignores the merging vehicle until it traverses the merge point, after which it follows IDM. While CIDM was originally proposed to work with only one vehicle merging into a main lane, we extend it to work with multiple merging vehicles.

We baseline our models against three driver models. First, vanilla IDM that uses parameter values as obtained by heuristics (Treiber et al., 2000), representing a purely rule-based model. The second baseline model is CIDM, which has a cooperation parameter of 1 in addition to the parameter values for IDM. This represents the most cautious setting of the cooperative IDM (Bouton et al., 2019). The final baseline is LMIDM, which represents the addition of data processing to underlying rules. In LMIDM, we use non-linear least squares to estimate the parameters from data using the Levenberg-Marquardt (LM) algorithm (Moré, 1978).

4.6.2 Results

Figure 4.9 shows the imitation performance in terms of how closely the generated trajectories match the ground truth demonstration trajectories. The particle filtering

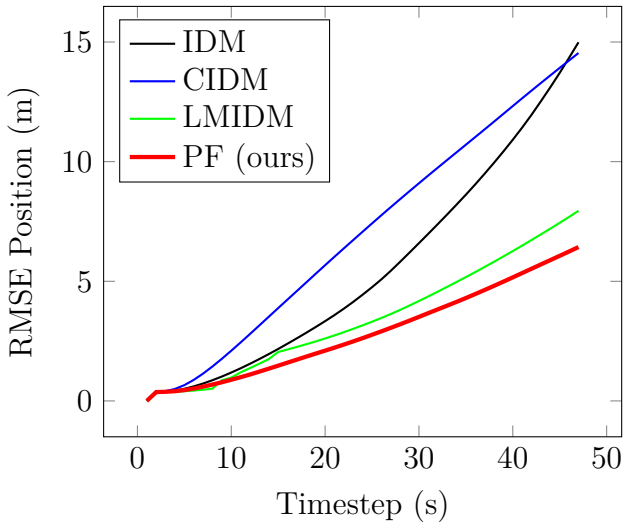

Figure 4.9: Root mean square error (RMSE) between ground truth trajectory and those generated using our driver models. Particle filtering based cooperative IDM shows the lowest RMSE values indicating the best imitation performance.

based approach to learn parameters of the cooperative-IDM shows the lowest RMSE value highlighting its ability to replicate ground truth trajectories.

Table 4.2 shows the number of collisions between vehicles arising out of the driver model simulations. Since IDM is a lane follower, and not designed to perform merges, it shows a high number of collisions. CIDM is extremely cautious and as expected,

Model	**Collision Fraction**
IDM	7.66
CIDM	0.00
LMIDM	1.28
PF (ours)	0.00

Table 4.2: Collision fraction observed in simulations carried out using different driver models. The vanilla IDM based models show collisions since they are unaware of merging. The particle filtering based and the most cooperative CIDM do not show any collisions.

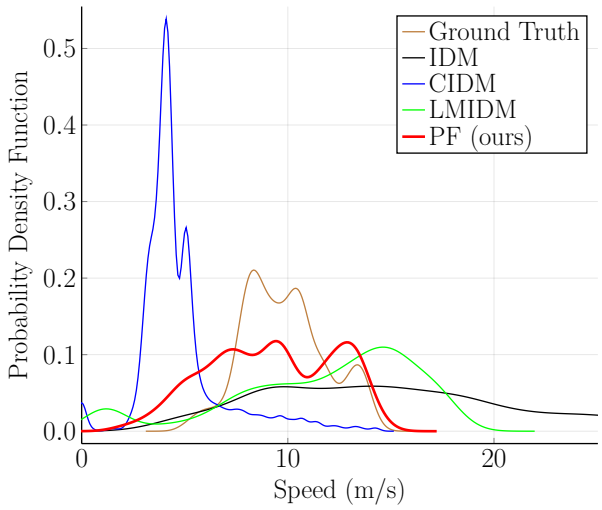

Figure 4.10: Speed distribution over all vehicles obtained from different driver models compared against ground truth demonstration. The particle filtering based models show the closest speed distribution to the ground truth. IDM results in a speed distribution that is not close to the demonstations due to parameters being selected using heurisitics. The maximum cooperation based CIDM shows very low speeds due to extreme caution. The LMIDM model is closer to the ground truth since it leverages the data.

therefore it does not cause vehicle collisions however suffers from poor RMSE performance as seen in Figure 4.9. The LMIDM based models result in some collisions and the particle filtering based models do not show any collisions. These results show the benefit of interpretable models. Had we seen collisions while using black-box models, we would not have been able to interpret the behavior leading to those collisions.

Figure 4.10 shows the speed distribution obtained using driving behavior generated by the different driver models, along with the speed distribution in the ground truth demonstrations. As expected, the speeds obtained by the CIDM tend toward 0 because of extreme caution. The particle filtering based driver model shows the closest speed distribution to ground truth, which shows the ability of these models generate good emergent driving behavior in addition to performing individual trajectory imitation as shown in figure 4.9.

To assess the scenario generation capability of our driver models, we conducted a 'driving Turing test'. We showed 21 human participants 10 different videos of both

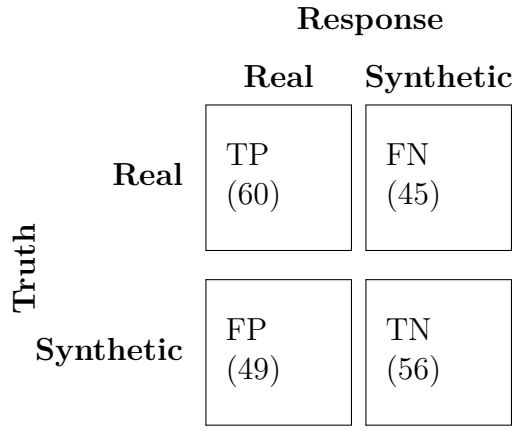

Figure 4.11: The confusion matrix obtained from the 'driving Turing test' results. Participants were shown videos of real and synthetic driving videos. Real is considered as positive and synthetic as negative for the purposes of the confusion matrix. The numbers in the specific squares represent the number of responses under the categories: true positive (TP), false negative (FN), false positive (FP), and true negative (TN).

replays from the dataset and synthetic trajectories generated using our driver models. They were asked to judge whether the video being shown was real or synthetic.

Figure 4.11 shows the confusion matrix obtained from this classification exercise. The accuracy was 55.71%, indicating that our driver models perform reasonably well in terms of generating realistic driving behavior. Figure 4.12 shows the responses in the form of a table with the columns being the scenario number, and the rows being the different human subjects. The black squares indicate misclassification. Almost half the squares are black, which confirms the ability of our driver models to generate realistic human behavior.

4.7 Discussion

This chapter demonstrated the hybrid rule-based and data-driven approach discussed in the previous chapter on the problem of driver modeling. A distribution over the

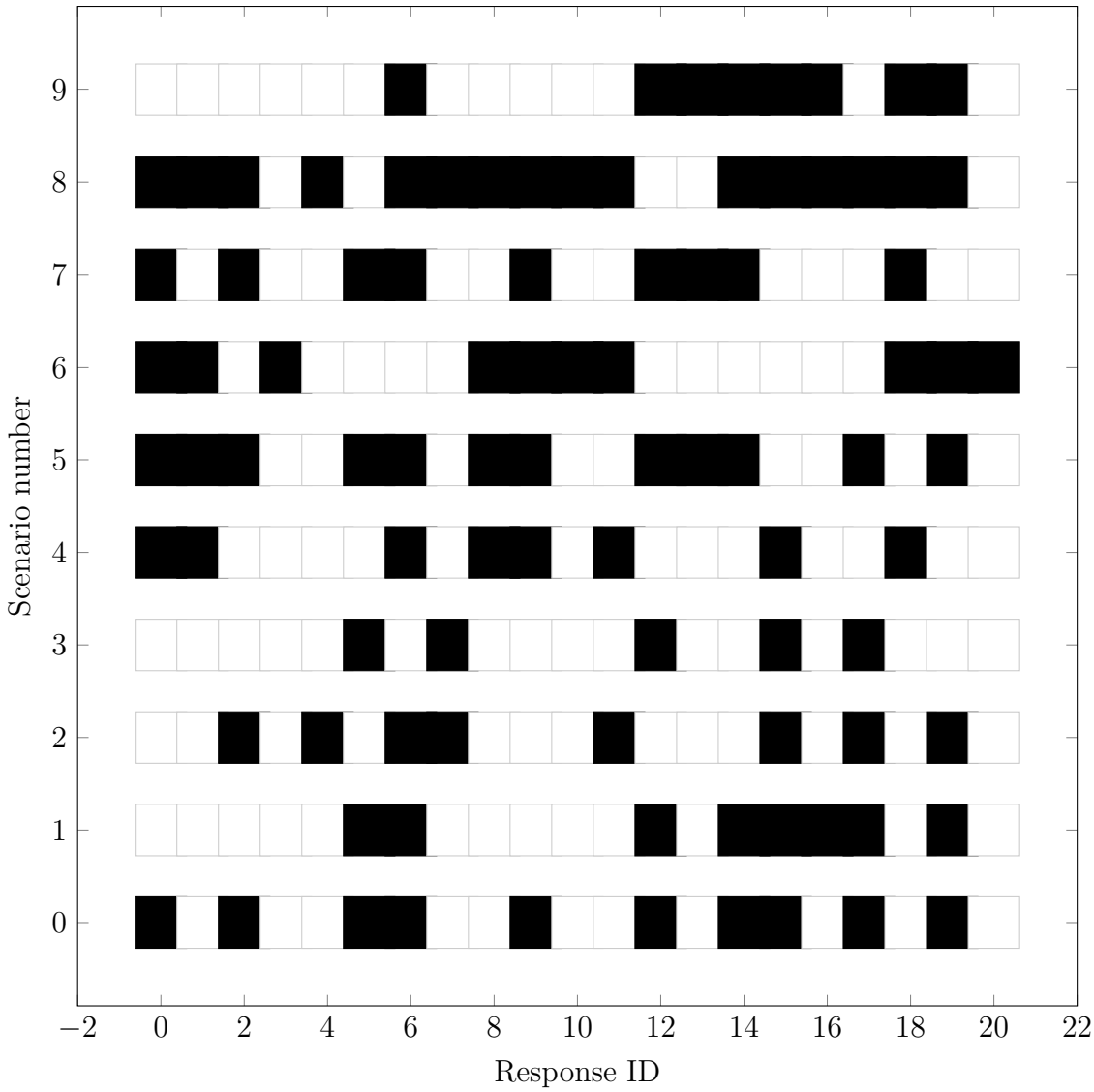

Figure 4.12: 'Driving Turing test' results based on survey responses gathered from 21 human respondents who were shown 10 videos that included both real and synthetic human driving trajectories. They were asked to identify whether they were seeing real driving or driving synthesized using our driver models. The black squares indicate misclassification.

parameters of the stochastic intelligent driver model was learnt in the case of highway driving. A distribution over the parameters of the stochastic cooperative intelligent driver model was learnt in the case of highway merging. Experiments were performed on three real world datasets namely, NGSIM, HighD, and Interaction to model driving behavior. The results showed that driver models were able to imitate the demonstrated driving trajectories as well as generate novel driving trajectories.

Chapter 5

Conclusions

This chapter summarizes the content and contributions of this thesis, and concludes with an outline for future research directions.

5.1 Summary

Autonomous agents need to understand human behavior to work with and around humans. Currently, most autonomous systems do not have such reasoning capabilities which forces them to operate in low-risk roles with minimal human interaction. However, this will need to change with the rising growth of automation in transportation, warehouses, and manufacturing.

Modeling human behavior is challenging because of stochasticity, multimodality, non-linearity, high dimensional state action spaces, latent intents, and interaction with other agents. This thesis presented data-driven techniques to model human behavior from demonstrated trajectories. The methods proposed in this thesis were demonstrated on the problem of driver modeling. Here, the goal was to learn driving policies by observing trajectories obtained from human drivers. The performance of the resulting driving policies was assessed by the closeness of generated rollout trajectories to the ground truth, and how realistic the resulting driving behavior was.

Driving can be viewed as a sequential decision making task under uncertainty. Hence it can be modeled using an MDP. However, the cost function of the driving

MDP is unknown making it challenging to solve using established methods. Imitation learning is a promising approach to solve MDPs where the cost function is unknown. Imitation learning uses demonstration trajectories gathered from experts solving the task to recover policies that perform at least as well as the experts.

Generative Adversarial Imitation Learning is a recent technique that enables scaling up imitation learning to real world problems with continuous state-action spaces. However, GAIL was proposed to learn policies for single agent tasks such as the cart-pole or mujoco problems. Using techniques from multi-agent reinforcement learning, this thesis presented parameter-sharing GAIL which enables GAIL to learn multi-agent policies. This thesis showed that PS-GAIL is better able to scale with increasing number of agents on the problem of highway driving. Further, PS-GAIL was shown to generate more realistic driving behavior as assessed by the emergent metrics of driving.

GAIL (and PS-GAIL) are general techniques that do not capture specific domain knowledge of the task being performed. To enable the designers of the learning agents to provide this domain knowledge, this thesis presented Reward Augmented Imitation Learning (RAIL). Using penalty functions, RAIL augments the learning signal to the learning agents by penalizing undesirable states. This helps guide the exploration into beneficial states. RAIL was shown to outperform PS-GAIL on experiments on highway driving.

For safety critical domains such as driving, interpretability of the resulting driving behavior is important. Black box models such as the imitation learning based models are not interpretable since they represent the driving policies using deep neural networks. Towards the goal of achieving interpretable policies, this thesis presented a hybrid rule-based and data-driven approach to modeling human behavior from demonstrations. The underlying rule-based models of human behavior were assumed to evolve the state of the system according to interpretable hidden parameters. The well established technique of particle filtering was used to infer distributions over these parameters online from demonstration data. The hybrid rule-based and data-driven technique was compared against heuristic rule-based models and black-box imitation learning based models. Rollout trajectories were generated by sampling parameters

from the obtained distribution and their closeness was assessed to the demonstration trajectories. In addition to better imitation performance, the hybrid rule-based and data-driven method demonstrated safe behavior without any collisions or off the road driving.

Models of human behavior can be used for building simulation platforms to replicate the intended environment of operation for autonomous agents. This can be used for safety validation in simulation. Towards this goal, the models developed in this thesis were used to generate trajectories for highway merging scenarios. Videos of the generated trajectories were then shown to human participants to assess their realism as compared to the ground truth trajectories. The results from this driving Turing test showed that the models developed in this thesis are able to generate reasonably realistic behavior on the scenario of merging.

5.2 Contributions

This thesis makes the following contributions:

- **Multi-agent Generative Adversarial Imitation Learning**: GAIL was originally proposed to work in situations where there is a singe learning agent for instance continuous control tasks in Mujoco (Todorov et al., 2012). However, it does not work well for multiple interacting agents as seen by experiments on driver modeling. This is because the training and test distributions are different thereby causing a covariate shift problem. To alleviate this problem, this thesis proposed a method based on parameter sharing to learn from demonstrations provided by multiple interacting agents (section 2.5). Experience gathered from multiple demonstration trajectories was batched together to inform the learning agent. Experiments on highway driving showed that this method scales better with increasing number of agents as compared to single agent GAIL, and this results in better imitation performance (section 3.6).

- **Reward augmented imitation learning**: The learning agent in GAIL does not explicitly understand the domain of learning. The learning problem is posed

in general terms where an underlying MDP is assumed. For structured domains such as driving which have governing rules, this thesis proposed a method to provide that domain knowledge to the learning agent via reward augmentation (section 3.6.1). Using experiments on highway driving, it was demonstrated that more accurate driving models were obtained through the use of shaping rewards that provided the learning agent with rules.

- **Hybrid rule-based and data-driven modeling**: Black-box model learning based models do not provide interpretability which hinders their deployment on safety-critical domains such as driving. This thesis proposed a method to learn a distribution over the parameters of underlying rule-based models (section 4.2). Using the well established technique of particle filtering, we showed that driver models can be learned that are inherently collision free due to baked in safety constraints, and also show greater fidelity to data as compared to previous rule-based model (section 4.5).

- **Scenario generation**: We used the models obtained from particle filtering to generate new scenarios in a merging case study (section 4.6). Using a driving Turing test where we showed human participants different videos of ground truth, and synthetic behavior, our scenario generation method was demonstrated to generate realistic driving trajectories.

5.3 Future Work

This thesis has drawn upon concepts from imitation learning, multi-agent systems, and system identification. Each of these fields contains a rich set of existing and emerging methods. Many opportunities exist for extending the ideas presented in this work. Some areas of further study are highlighted below.

5.3.1 Safety Validation in Simulation

This thesis focused on developing models of human behavior and demonstrated them on the problem of driver modeling. The models can be incorporated into any simulation platform as a collection of parameters for the driver models. This simulator can then be used to test autonomous driving planners by placing the autonomous vehicle in test scenarios (Corso et al., 2020).

5.3.2 Incorporating Learning into Planning

This thesis was focused on the problem of learning models of human motion from demonstrations. Models of human behavior can be used to inform decision making of autonomous agents. The learned human models can be incorporated into planning algorithms such as Monte Carlo Tree Search (Sunberg and Kochenderfer, 2020) where the hallucinated future states evolve according to the learned models of traffic participants.

5.3.3 Relaxing the Assumption of Cooperation

This thesis made the assumption of cooperative agents in the multi-agent learning process. This allowed the use of a policy with shared parameters across all the agents (Terry et al., 2020). However, in real world human motion scenarios, participants are not always cooperative and their behavior lies somewhere in the spectrum from cooperative to adversarial. This raises the question of learning multi-agent motion from demonstrations without the assumption of cooperation (Etesami and Straehle, 2020).

5.3.4 Accounting for Partial Observability

Partial observability in the inputs to the models developed in this thesis can arise out of two sources: sensor noise, and latent driver states. In this thesis, we assumed that there was no sensor noise. Future work could investigate the impact of imperfect sensor information on driver models. Further, the inherent intent and traits of the

drivers which were unmodeled in this work. Future work could incorporate modeling the intent of human agents (Brown et al., 2020).

Bibliography

Abbeel, P., D. Dolgov, A. Ng, and S. Thrun (2008). "Apprenticeship learning for motion planning with application to parking lot navigation". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (cit. on p. 6).

Abbeel, P. and A. Ng (2004). "Apprenticeship learning via inverse reinforcement learning". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 5, 6, 10, 12, 14, 46).

Alahi, A., K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese (2016). "Social LSTM: Human trajectory prediction in crowded spaces". In: *International Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 3).

Altché, F. and A. de La Fortelle (2017). "An LSTM network for highway trajectory prediction". In: *IEEE International Conference on Intelligent Transportation Systems (ITSC)* (cit. on pp. 3, 4, 5).

Argall, B., S. Chernova, M. Veloso, and B. Browning (2009). "A survey of robot learning from demonstration". In: *Robotics and Autonomous Systems* 57.5, pp. 469–483 (cit. on p. 9).

Arjovsky, M., S. Chintala, and L. Bottou (2017). "Wasserstein Generative Adversarial Networks". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 21, 36).

Armand, A., D. Filliat, and J. Ibanez-Guzman (2013). "Modelling stop intersection approaches using gaussian processes". In: *IEEE International Conference on Intelligent Transportation Systems (ITSC)* (cit. on p. 5).

Bai, H., S. Cai, N. Ye, D. Hsu, and W. S. Lee (2015). "Intention-aware online POMDP planning for autonomous driving in a crowd". In: *International Conference on Robotics and Automation (ICRA)* (cit. on p. 46).

Bansal, G., B. Nushi, E. Kamar, W. Lasecki, D. Weld, and E. Horvitz (2019). "Beyond accuracy: The role of mental models in human-AI team performance". In: *AAAI Conference on Human Computation and Crowdsourcing (HCOMP)* (cit. on p. 45).

Bansal, M., A. Krizhevsky, and A. Ogale (2018). "Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst". In: *arXiv preprint arXiv:1812.03079* (cit. on p. 4).

Bhattacharyya, R., D. Phillips, C. Liu, J. Gupta, K. Driggs-Campbell, and M. Kochenderfer (2019). "Simulating emergent properties of human driving behavior using reward augmented multi-agent imitation learning". In: *IEEE International Conference on Robotics and Automation (ICRA)* (cit. on pp. 8, 36).

Bhattacharyya, R., D. Phillips, B. Wulfe, J. Morton, A. Kuefler, and M. Kochenderfer (2018). "Multi-agent imitation learning for driving simulation". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (cit. on pp. 8, 22, 57, 59, 60).

Bhattacharyya, R., R. Senanayake, K. Brown, and M. Kochenderfer (2020a). "Online parameter estimation for human driver behavior prediction". In: *American Control Conference (ACC)* (cit. on p. 8).

Bhattacharyya, R., B. Wulfe, D. Phillips, A. Kuefler, J. Morton, R. Senanayake, and M. Kochenderfer (2020b). "Modeling human driving behavior through Generative Adversarial Imitation Learning". In: *arXiv preprint arXiv:2006.06412* (cit. on pp. 5, 45, 46).

Bloem, M. and N. Bambos (2014). "Infinite time horizon maximum causal entropy inverse reinforcement learning". In: *IEEE Conference on Decision and Control (CDC)* (cit. on p. 14).

Bouton, M., A. Nakhaei, K. Fujimura, and M. Kochenderfer (2019). "Cooperation-aware reinforcement learning for merging in dense traffic". In: *Intelligent Transportation Systems Conference (ITSC)* (cit. on pp. 62, 63).

Boyraz, P., M. Acar, and D. Kerr (2007). "Signal modelling and Hidden Markov Models for driving manoeuvre recognition and driver fault diagnosis in an urban road scenario". In: *Intelligent Vehicles Symposium (IV)* (cit. on p. 30).

Brown, K., K. Driggs-Campbell, and M. Kochenderfer (2020). "Modeling and prediction of human driver behavior: A survey". In: *arXiv preprint arXiv:2006.08832* (cit. on pp. 4, 45, 74).

Buyer, J., D. Waldenmayer, N. Sußmann, R. Zöllner, and J. Zöllner (2019). "Interaction-aware approach for online parameter estimation of a multi-lane Intelligent Driver Model". In: *IEEE Intelligent Transportation Systems Conference* (cit. on p. 52).

Chandra, R., T. Guan, S. Panuganti, T. Mittal, U. Bhattacharya, A. Bera, and D. Manocha (2020). "Forecasting Trajectory and Behavior of Road-Agents Using Spectral Clustering in Graph-LSTMs". In: *Robotics and Automation Letters (RA-L)* 5.3, pp. 4882–4890 (cit. on p. 46).

Cho, K., B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio (2014). "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Conference on Empirical Methods in Natural Language Processing (EMNLP)* (cit. on p. 30).

Colyar, J. and J. Halkias (2007). *US highway 101 dataset*. Tech. rep. FHWA-HRT-07-030. Federal Highway Administration (FHWA) (cit. on pp. 27, 55).

Corso, A., R. Moss, M. Koren, R. Lee, and M. Kochenderfer (2020). "A Survey of Algorithms for Black-Box Safety Validation". In: *arXiv preprint arXiv:2005.02979* (cit. on p. 73).

Deo, N., A. Rangesh, and M. Trivedi (2018). "How would surround vehicles move? a unified framework for maneuver classification and motion prediction". In: *IEEE Transactions on Intelligent Vehicles* 3.2, pp. 129–140 (cit. on p. 5).

Diallo, E., A. Sugiyama, and T. Sugawara (2017). "Learning to coordinate with deep reinforcement learning in doubles pong game". In: *International Conference on Machine Learning and Applications (ICMLA)* (cit. on p. 23).

Diehl, F., T. Brunner, M. T. Le, and A. Knoll (2019). "Graph neural networks for modelling traffic participant interaction". In: *IEEE Intelligent Vehicles Symposium (IV)* (cit. on p. 4).

Ding, W., J. Chen, and S. Shen (2019). "Predicting vehicle behaviors over an extended horizon using behavior interaction network". In: *IEEE International Conference on Robotics and Automation (ICRA)* (cit. on p. 3).

Dragan, A., K. Lee, and S. Srinivasa (2013). "Legibility and predictability of robot motion". In: *International Conference on Human-Robot Interaction (HRI)* (cit. on p. 46).

Driggs-Campbell, K., V. Shia, and R. Bajcsy (2015). "Improved driver modeling for human-in-the-loop vehicular control". In: *IEEE International Conference on Robotics and Automation (ICRA)* (cit. on p. 4).

Duan, Y., X. Chen, R. Houthooft, J. Schulman, and P. Abbeel (2016). "Benchmarking deep reinforcement learning for continuous control". In: *International Conference on Machine Learning (ICML)* (cit. on p. 34).

Eggert, J., F. Damerow, and S. Klingelschmitt (2015). "The foresighted driver model". In: *IEEE Intelligent Vehicles Symposium (IV)* (cit. on p. 52).

Etesami, J. and C.-N. Straehle (2020). "Non-cooperative Multi-agent Systems with Exploring Agents". In: *arXiv preprint arXiv:2005.12360* (cit. on p. 73).

Fisac, J., E. Bronstein, E. Stefansson, D. Sadigh, S. S. Sastry, and A. Dragan (2019). "Hierarchical game-theoretic planning for autonomous vehicles". In: *International Conference on Robotics and Automation (ICRA)* (cit. on pp. 5, 62).

Foerster, J., I. A. Assael, N. De Freitas, and S. Whiteson (2016). "Learning to communicate with deep multi-agent reinforcement learning". In: *Advances in neural information processing systems (NeurIPS)* (cit. on p. 23).

Galceran, E., A. Cunningham, R. Eustice, and E. Olson (2017). "Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction: Theory and experiment". In: *Autonomous Robots* 41.6, pp. 1367–1382 (cit. on p. 52).

Ghasemipour, S. K. S., R. Zemel, and S. Gu (2020). "A divergence minimization perspective on imitation learning methods". In: *Conference on Robot Learning (CoRL)* (cit. on p. 21).

Gipps, P. (1981). "A behavioural car-following model for computer simulation". In: *Transportation Research Part B: Methodological* 15.2, pp. 105–111 (cit. on p. 5).

Gombolay, M., R. Jensen, J. Stigile, T. Golen, N. Shah, S.-H. Son, and J. Shah (2018). "Human-machine collaborative optimization via apprenticeship scheduling". In: *Journal of Artificial Intelligence Research* 63, pp. 1–49 (cit. on p. 46).

González, D. S., M. Garzón, J. S. Dibangoye, and C. Laugier (2019). "Human-like decision-making for automated driving in highways". In: *IEEE Intelligent Transportation Systems Conference (ITSC)* (cit. on p. 5).

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014). "Generative adversarial nets". In: *Advances in Neural Information Processing Systems (NeurIPS)* (cit. on pp. 10, 18).

Gruver, N., J. Song, M. Kochenderfer, and S. Ermon (2020). "Multi-agent adversarial inverse reinforcement learning with latent variables". In: *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)* (cit. on p. 22).

Gulrajani, I., F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville (2017). "Improved training of Wasserstein GANs". In: *Neural Information Processing Systems (NeurIPS)* (cit. on p. 35).

Gupta, A., J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi (2018). "Social gan: Socially acceptable trajectories with generative adversarial networks". In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 3).

Gupta, J., M. Egorov, and M. Kochenderfer (2017). "Cooperative multi-agent control using deep reinforcement learning". In: *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)* (cit. on pp. 23, 34).

Gupta, J., K. Menda, Z. Manchester, and M. Kochenderfer (2020). "Structured mechanical models for robot learning and control". In: *Conference on Learning for Dynamics and Control (L4DC)* (cit. on p. 46).

Gweon, H. and R. Saxe (2013). "Developmental cognitive neuroscience of theory of mind". In: *Neural Circuit Development and Function in the Brain*, pp. 367–377 (cit. on p. 1).

Hadfield-Menell, D., S. Milli, P. Abbeel, S. J. Russell, and A. Dragan (2017). "Inverse reward design". In: *Advances in Neural Information Processing Systems (NeurIPS)* (cit. on p. 46).

Helbing, D., I. Farkas, and T. Vicsek (2000). "Simulating dynamical features of escape panic". In: *Nature* 407.6803, pp. 487–490 (cit. on p. 2).

Helbing, D. and P. Molnar (1995). "Social force model for pedestrian dynamics". In: *Physical Review E* 51.5, pp. 4282–4286 (cit. on pp. 2, 44, 46).

Ho, J. and S. Ermon (2016). "Generative adversarial imitation learning". In: *Advances in Neural Information Processing Systems (NeurIPS)* (cit. on pp. 10, 15, 17).

Ho, J., J. Gupta, and S. Ermon (2016). "Model-free imitation learning with policy optimization". In: *International Conference on Machine Learning (ICML)* (cit. on p. 10).

Hoermann, S., D. Stumper, and K. Dietmayer (2017). "Probabilistic long-Term prediction for autonomous vehicles". In: *IEEE Intelligent Vehicles Symposium (IV)* (cit. on p. 52).

Hongfei, J., J. Zhicai, and N. Anning (2003). "Develop a car-following model using data collected by five-wheel system". In: *Intelligent Transporation Systems Conference (ITSC)* (cit. on p. 30).

Hubmann, C., J. Schulz, G. Xu, D. Althoff, and C. Stiller (2018). "A belief state planner for interactive merge maneuvers in congested traffic". In: *International Conference on Intelligent Transportation Systems (ITSC)* (cit. on p. 62).

Isele, D. (2019). "Interactive decision making for autonomous vehicles in dense traffic". In: *IEEE Intelligent Transportation Systems Conference (ITSC)* (cit. on p. 5).

Ivanovic, B., K. Leung, E. Schmerling, and M. Pavone (2020). "Multimodal Deep Generative Models for Trajectory Prediction: A Conditional Variational Autoencoder Approach". In: *arXiv preprint arXiv:2008.03880* (cit. on p. 3).

Jain, A., A. R. Zamir, S. Savarese, and A. Saxena (2016). "Structural-rnn: Deep learning on spatio-temporal graphs". In: *International Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 3).

Kesting, A., M. Treiber, and D. Helbing (2007). "General lane-changing model MOBIL for car-following models". In: *Transportation Research Record* 1999.1, pp. 86–94 (cit. on pp. 7, 33, 47).

— (2010). "Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity". In: *Philosophical Transactions of the Royal Society A:*

Mathematical, Physical and Engineering Sciences 368.1928, pp. 4585–4605 (cit. on p. 52).

Khodayari, A., A. Ghaffari, R. Kazemi, and R. Braunstingl (2012). "A modified car-following model based on a neural network model of the human driver effects". In: *IEEE Transactions on Systems, Man, and Cybernetics* 42.6, pp. 1440–1449 (cit. on p. 30).

Kim, B., C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi (2017). "Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network". In: *IEEE International Conference on Intelligent Transportation Systems (ITSC)* (cit. on p. 5).

Kingma, D. P. and J. Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (cit. on pp. 24, 35).

Koopman, P. and M. Wagner (2016). "Challenges in autonomous vehicle testing and validation". In: *SAE International Journal of Transportation Safety* 4.1, pp. 15–24 (cit. on p. 26).

Krajewski, R., J. Bock, L. Kloeker, and L. Eckstein (2018). "The highD dataset: A drone dataset of naturalistic vehicle trajectories on German highways for validation of highly automated driving systems". In: *IEEE Intelligent Transportation Systems Conference (ITSC)* (cit. on p. 55).

Krajewski, R., T. Moers, A. Meister, and L. Eckstein (2019). "BézierVAE: Improved trajectory modeling using variational autoencoders for the safety validation of highly automated vehicles". In: *IEEE Intelligent Transportation Systems Conference (ITSC)* (cit. on p. 5).

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)* (cit. on p. 30).

Kuefler, A., J. Morton, T. Wheeler, and M. Kochenderfer (2017). "Imitating driver behavior with generative adversarial networks". In: *IEEE Intelligent Vehicles Symposium (IV)* (cit. on pp. 3, 5, 32, 52).

Lee, H., R. Grosse, R. Ranganath, and A. Ng (2009). "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations". In: *International Conference on Machine Learning (ICML)* (cit. on p. 30).

Lee, R., O. Mengshoel, A. Saksena, R. Gardner, D. Genin, J. Brush, and M. Kochenderfer (2018). "Differential adaptive stress testing of airborne collision avoidance systems". In: *AIAA Modeling and Simulation Conference* (cit. on p. 46).

Lefèvre, S., C. Sun, R. Bajcsy, and C. Laugier (2014). "Comparison of parametric and non-parametric approaches for vehicle speed prediction". In: *American Control Conference (ACC)* (cit. on pp. 30, 32, 52).

Levine, S. and V. Koltun (2012). "Continuous inverse optimal control with locally optimal examples". In: *International Conference on Machine Learning (ICML)* (cit. on p. 5).

Li, Y., J. Song, and S. Ermon (2017). "Infogail: Interpretable imitation learning from visual demonstrations". In: *Advances in Neural Information Processing Systems (NeurIPS)* (cit. on pp. 35, 36).

Liebner, M., M. Baumann, F. Klanner, and C. Stiller (2012). "Driver intent inference at urban intersections using the Intelligent Driver Model". In: *IEEE Intelligent Vehicles Symposium (IV)* (cit. on p. 52).

Littman, M. (1994). "Markov games as a framework for multi-agent reinforcement learning". In: *Machine Learning Proceedings*. Elsevier, pp. 157–163 (cit. on p. 22).

Liu, J. S. and R. Chen (1998). "Sequential Monte Carlo methods for dynamic systems". In: *Journal of the American Statistical Association* 93.443, pp. 1032–1044 (cit. on p. 49).

Liu, Q., B. Lathrop, and V. Butakov (2014). "Vehicle lateral position prediction: A small step towards a comprehensive risk assessment system". In: *Intelligent Transportation Systems Conference (ITSC)* (cit. on p. 30).

Luo, W., B. Yang, and R. Urtasun (2018). "Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net". In: *Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 4).

Maurer, M., C. Gerdes, B. Lenz, and H. Winner (2016). *Autonomous driving: technical, legal and social aspects*. Springer Nature (cit. on p. 4).

Menda, K., J. de Becdelièvre, J. K. Gupta, I. Kroo, M. J. Kochenderfer, and Z. Manchester (2020). "Scalable Identification of Partially Observed Systems with Certainty-Equivalent EM". In: *International Conference on Machine Learning (ICML)* (cit. on p. 46).

Monteil, J., N. OHara, V. Cahill, and M. Bouroche (2015). "Real-time estimation of drivers' behaviour". In: *IEEE Intelligent Transportation Systems Conference (ITSC)* (cit. on p. 52).

Moré, J. J. (1978). "The Levenberg-Marquardt algorithm: implementation and theory". In: *Numerical Analysis*. Springer, pp. 105–116 (cit. on p. 63).

Morton, J. and M. Kochenderfer (2017). "Simultaneous policy learning and latent state inference for imitating driver behavior". In: *International Conference on Intelligent Transportation Systems (ITSC)* (cit. on pp. 57, 59, 60).

Morton, J., T. Wheeler, and M. Kochenderfer (2016). "Analysis of recurrent neural networks for probabilistic modeling of driver behavior". In: *Transactions on Intelligent Transportation Systems* 18.5, pp. 1289–1298 (cit. on pp. 5, 30, 52).

— (2018). "Closed-loop policies for operational tests of safety-critical systems". In: *IEEE Transactions on Intelligent Transporation Systems*, pp. 317–328 (cit. on p. 26).

Ng, A. and S. Russell (2000). "Algorithms for inverse reinforcement learning". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 6, 10).

Nguyen, T., N. Nguyen, and S. Nahavandi (2020). "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications". In: *IEEE Transactions on Cybernetics*, pp. 3826–3839 (cit. on p. 23).

Nikolaidis, S., S. Nath, A. Procaccia, and S. Srinivasa (2017). "Game-theoretic modeling of human adaptation in human-robot collaboration". In: *International Conference on Human-Robot Interaction (HRI)* (cit. on p. 3).

Okuda, H., K. Harada, T. Suzuki, S. Saigo, and S. Inoue (2017). "Design of automated merging control by minimizing decision entropy of drivers on main lane". In: *IEEE Intelligent Vehicles Symposium (IV)* (cit. on p. 5).

Palmer, G., K. Tuyls, D. Bloembergen, and R. Savani (2018). "Lenient Multi-Agent Deep Reinforcement Learning". In: *International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)* (cit. on p. 23).

Panwai, S. and H. Dia (2007). "Neural Agent Car-Following Models". In: *Transactions on Intelligent Transportation Systems* 8.1, pp. 60–70 (cit. on p. 30).

Park, S., B. Kim, C. Kang, C. Chung, and J. Choi (2018). "Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture". In: *IEEE Intelligent Vehicles Symposium (IV)* (cit. on p. 3).

Pearson, R. and M. Pottmann (2000). "Gray-box identification of block-oriented non-linear models". In: *Journal of Process Control* 10.4, pp. 301–315 (cit. on p. 46).

Pentland, A. and A. Liu (1999). "Modeling and prediction of human behavior". In: *Neural computation* 11.1, pp. 229–242 (cit. on p. 45).

Pomerleau, D. (1989). "Alvinn: An autonomous land vehicle in a neural network". In: *Advances in Neural Information Processing Systems (NeurIPS)* (cit. on pp. 6, 9).

Pruekprasert, S., X. Zhang, J. Dubut, C. Huang, and M. Kishida (2019). "Decision making for autonomous vehicles at unsignalized intersection in presence of malicious vehicles". In: *IEEE Intelligent Transportation Systems Conference (ITSC)* (cit. on p. 5).

Reddy, S., A. Dragan, and S. Levine (2018). "Where do you think you're going?: Inferring beliefs about dynamics from behavior". In: *Advances in Neural Information Processing Systems (NeurIPS)* (cit. on p. 46).

Ross, S. and D. Bagnell (2010). "Efficient reductions for imitation learning". In: *International Conference on Artificial Intelligence and Statistics (AISTATS)* (cit. on p. 9).

Ross, S., G. Gordon, and D. Bagnell (2011). "A reduction of imitation learning and structured prediction to no-regret online learning". In: *International Conference on Artificial Intelligence and Statistics (AISTATS)* (cit. on pp. 9, 10, 12, 46).

Rudenko, A., L. Palmieri, and K. Arras (2018). "Joint long-term prediction of human motion using a planning-based social force approach". In: *IEEE International Conference on Robotics and Automation (ICRA)* (cit. on p. 2).

Sadigh, D., N. Landolfi, S. Sastry, S. Seshia, and A. Dragan (2018). "Planning for cars that coordinate with people: leveraging effects on human actions for planning and active information gathering over human internal state". In: *Autonomous Robots* 42.7, pp. 1405–1426 (cit. on pp. 52, 62).

Schaal, S. (1999). "Is imitation learning the route to humanoid robots?" In: *Trends in cognitive sciences* 3.6, pp. 233–242 (cit. on pp. 9, 12).

Schmerling, E., K. Leung, W. Vollprecht, and M. Pavone (2018). "Multimodal probabilistic model-based planning for human-robot interaction". In: *International Conference on Robotics and Automation (ICRA)* (cit. on p. 62).

Schön, T., F. Gustafsson, and R. Karlsson (2011). "The particle filter in practice". In: *The Oxford Handbook of Nonlinear Filtering*. Oxford University Press (cit. on p. 54).

Schulman, J., S. Levine, P. Abbeel, M. Jordan, and P. Moritz (2015). "Trust region policy optimization". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 10, 18, 24).

Schulz, J., C. Hubmann, J. Löchner, and D. Burschka (2018a). "Interaction-aware probabilistic behavior prediction in urban environments". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (cit. on p. 52).

— (2018b). "Multiple model unscented kalman filtering in dynamic bayesian networks for intention estimation and trajectory prediction". In: *IEEE International Conference on Intelligent Transportation Systems (ITSC)* (cit. on p. 52).

Schwarz, G. (1978). "Estimating the dimension of a model". In: *The Annals of Statistics* 6.2, pp. 461–464 (cit. on p. 32).

Shimodaira, H. (2000). "Improving predictive inference under covariate shift by weighting the log-likelihood function". In: *Journal of Statistical Planning and Inference* 90.2, pp. 227–244 (cit. on p. 12).

Song, J., H. Ren, D. Sadigh, and S. Ermon (2018). "Multi-agent generative adversarial imitation learning". In: *Advances in Neural Information Processing Systems (NeurIPS)* (cit. on pp. 22, 24).

Sukhbaatar, S., A. Szlam, and R. Fergus (2016). "Learning multiagent communication with backpropagation". In: *Advances in neural information processing systems (NeurIPS)* (cit. on p. 23).

Sunberg, Z., C. Ho, and M. Kochenderfer (2017). "The value of inferring the internal state of traffic participants for autonomous freeway driving". In: *American Control Conference* (cit. on p. 52).

Sunberg, Z. and M. Kochenderfer (2020). "Improving automated driving through planning with human internal states". In: *arXiv preprint arXiv:2005.14549* (cit. on p. 73).

Syed, U. and R. Schapire (2008). "A game-theoretic approach to apprenticeship learning". In: *Advances in Neural Information Processing Systems (NeurIPS)* (cit. on pp. 10, 14).

Terry, J., N. Grammel, A. Hari, L. Santos, B. Black, and D. Manocha (2020). "Parameter Sharing is Surprisingly Useful for Multi-Agent Deep Reinforcement Learning". In: *arXiv preprint 2005.13625* (cit. on pp. 23, 73).

Thrun, S. (2002). "Particle filters in robotics". In: *Uncertainty in Artificial Intelligence (UAI)* (cit. on p. 49).

Todorov, E., T. Erez, and Y. Tassa (2012). "Mujoco: A physics engine for model-based control". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (cit. on pp. 6, 71).

Treiber, M., A. Hennecke, and D. Helbing (2000). "Congested traffic states in empirical observations and microscopic simulations". In: *Physical Review E* 62.2, pp. 1805–1852 (cit. on pp. 2, 7, 32, 44, 47, 51, 63).

Treiber, M. and A. Kesting (2017). "The intelligent driver model with stochasticity-new insights into traffic flow oscillations". In: *Transportation Research Procedia* 23, pp. 174–187 (cit. on pp. 52, 55, 57, 59, 60).

Van Den Berg, J., S. Patil, J. Sewall, D. Manocha, and M. Lin (2008). "Interactive navigation of multiple agents in crowded environments". In: *Symposium on Interactive 3D Graphics and Games* (cit. on p. 3).

Vemula, A., K. Muelling, and J. Oh (2018). "Social attention: Modeling attention in human crowds". In: *IEEE International Conference on Robotics and Automation (ICRA)* (cit. on p. 3).

Völz, B., K. Behrendt, H. Mielenz, I. Gilitschenski, R. Siegwart, and J. Nieto (2016). "A data-driven approach for pedestrian intention estimation". In: *International Conference on Intelligent Transportation Systems (ITSC)* (cit. on p. 46).

Wang, M., Z. Wang, J. Talbot, C. Gerdes, and M. Schwager (2019). "Game theoretic planning for self-driving cars in competitive scenarios". In: *Robotics: Science and Systems (RSS)* (cit. on p. 3).

Wang, Z., K. Mülling, M. P. Deisenroth, H. Ben Amor, D. Vogt, B. Schölkopf, and J. Peters (2013). "Probabilistic movement modeling for intention inference in human–robot interaction". In: *The International Journal of Robotics Research (IJRR)* 32.7, pp. 841–858 (cit. on p. 46).

Ward, E., N. Evestedt, D. Axehill, and J. Folkesson (2017). "Probabilistic model for interaction aware planning in merge scenarios". In: *Transactions on Intelligent Vehicles* 2.2, pp. 133–146 (cit. on p. 62).

Wheeler, T., M. Kochenderfer, and P. Robbel (2015). "Initial scene configurations for highway traffic propagation". In: *International Conference on Intelligent Transportation Systems (ITSC)* (cit. on p. 5).

Wierstra, D., A. Förster, J. Peters, and J. Schmidhuber (2010). "Recurrent policy gradients". In: *Logic Journal of IGPL* 18.5, pp. 620–634 (cit. on p. 30).

Wiest, J., M. Höffken, U. Kreßel, and K. Dietmayer (2012). "Probabilistic trajectory prediction with gaussian mixture models". In: *IEEE Intelligent Vehicles Symposium (IV)* (cit. on p. 5).

Wu, J., J. Ruenz, and M. Althoff (2018). "Probabilistic map-based pedestrian motion prediction taking traffic participants into consideration". In: *IEEE Intelligent Vehicles Symposium (IV)* (cit. on p. 3).

Xiao, H., M. Herman, J. Wagner, S. Ziesche, J. Etesami, and T. H. Linh (2019). "Wasserstein adversarial imitation learning". In: *arXiv preprint arXiv:1906.08113* (cit. on p. 21).

Yoon, S. and D. Kum (2016). "The multilayer perceptron approach to lateral motion prediction of surrounding vehicles for autonomous vehicles". In: *IEEE Intelligent Vehicles Symposium (IV)* (cit. on p. 5).

Yu, L., J. Song, and S. Ermon (2019). "Multi-Agent Adversarial Inverse Reinforcement Learning". In: *International Conference on Machine Learning (ICML)* (cit. on p. 22).

Zhan, W., L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kümmerle, H. Königshof, C. Stiller, A. de La Fortelle, and M. Tomizuka (2019). "INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps". In: *arXiv preprint arXiv:1910.03088* (cit. on pp. 62, 63).

Ziebart, B., A. Bagnell, and A. Dey (2010). "Modeling interaction via the principle of maximum causal entropy". In: *International Conference on Machine Learning (ICML)* (cit. on p. 14).

Ziebart, B., A. Maas, A. Bagnell, and A. Dey (2008). "Maximum entropy inverse reinforcement learning". In: *AAAI Conference on Artificial Intelligence (AAAI)* (cit. on p. 14).